```
/*
########################################################################
#########################
# Rocket League (220224.66435.3685966/5/2024) SDK
# Generated with the UE3SDKGenerator v2.2.7
#
========================================================================
=========== #
# File: Engine_classes.hpp
#
========================================================================
=========== #
# Credits: TheFeckless, ItsBranK
# Links: www.github.com/itsbrank/UE3SDKGenerator, www.twitter.com/itsbrank
########################################################################
#########################
*/
#pragma once

#ifdef _MSC_VER
#pragma pack(push, 0x8)
#endif

/*
#
========================================================================
=========== #
# Constants
#
========================================================================
=========== #
*/

#define CONST_MINFLOORZ                                0.7
#define CONST_ACTORMAXSTEPHEIGHT                       35.0
#define CONST_RBSTATE_LOCATIONSCALE                    100.0
#define CONST_RBSTATE_QUATBITS                         18
#define CONST_RBSTATE_LINVELSCALE                      100.0
#define CONST_RBSTATE_ANGVELSCALE                      10000.0
#define CONST_RB_None                                  0x00
#define CONST_RB_NeedsUpdate                           0x01
#define CONST_RB_Sleeping                              0x02
#define CONST_REP_RBLOCATION_ERROR_TOLERANCE_SQ        16.0f
#define CONST_TRACEFLAG_ForceController                16
#define CONST_TRACEFLAG_Blocking                       8
#define CONST_TRACEFLAG_SkipMovers                     4
#define CONST_TRACEFLAG_PhysicsVolumes                 2
#define CONST_TRACEFLAG_Bullet                         1
#define CONST_SDPG_NumBits                             3
#define CONST_BLOCKEDPATHCOST                          10000000
#define CONST_LATENT_MOVETOWARD                        503
#define CONST_INFINITE_PATH_COST                       10000000
#define CONST_MAX_ACTIVE_CAMERA_ANIMS                  8
#define CONST_GET_SAVE_SLOT_INVALID                    -1
```

```
#define CONST_GET_SAVE_SLOT_ERROR                         -2
#define CONST_COMMON_DATA_SAVE_SLOT_INDEX                 -1
#define CONST_SAVE_SYSTEM_VERSION_KEY                     "CloudSaveSystemVersion"
#define CONST_SAVE_DATA_BLOB_NAME_KEY                     "DataBlobName"
#define CONST_DATA_STORE_ID_KEY                           "DataStoreID"
#define CONST_NUM_SAVE_SLOTS_KEY                          "NumSaveSlots"
#define CONST_DEFAULT_SIZE_X                              1024
#define CONST_DEFAULT_SIZE_Y                              768
#define CONST_MAX_SUPPORTED_GAMEPADS                      4
#define CONST_MaxHistory                                  16
#define CONST_COVERLINK_DangerDist                        1536.f
#define CONST_COVERLINK_EdgeExposureDot                   0.85f
#define CONST_COVERLINK_EdgeCheckDot                      0.25f
#define CONST_COVERLINK_ExposureDot                       0.4f
#define CONST_EpochYear                                   1970
#define CONST_SecondsInMinute                             60
#define CONST_SecondsInHour                               3600
#define CONST_SecondsInDay                                86400
#define CONST_SecondsInMonth                              2629743
#define CONST_SecondsInYear                               31556926
#define CONST_NULLCHARACTER                               127
#define CONST_FSM_DEFAULTRECYCLETIME                      0.2
#define CONST_HeaderFlags_NoEventStrings                  1
#define CONST_GAMEEVENT_MATCH_STARTED                     0
#define CONST_GAMEEVENT_MATCH_ENDED                       1
#define CONST_GAMEEVENT_ROUND_STARTED                     2
#define CONST_GAMEEVENT_ROUND_ENDED                       3
#define CONST_GAMEEVENT_GAME_CLASS                        6
#define CONST_GAMEEVENT_GAME_OPTION_URL                   7
#define CONST_GAMEEVENT_GAME_MAPNAME                      8
#define CONST_GAMEEVENT_MEMORYUSAGE_POLL                  35
#define CONST_GAMEEVENT_FRAMERATE_POLL                    36
#define CONST_GAMEEVENT_NETWORKUSAGEIN_POLL               37
#define CONST_GAMEEVENT_NETWORKUSAGEOUT_POLL              38
#define CONST_GAMEEVENT_PING_POLL                         39
#define CONST_GAMEEVENT_RENDERTHREAD_POLL                 40
#define CONST_GAMEEVENT_GAMETHREAD_POLL                   41
#define CONST_GAMEEVENT_GPUFRAMETIME_POLL                 42
#define CONST_GAMEEVENT_FRAMETIME_POLL                    43
#define CONST_GAMEEVENT_TEAM_CREATED                      50
#define CONST_GAMEEVENT_TEAM_GAME_SCORE                   51
#define CONST_GAMEEVENT_TEAM_MATCH_WON                    4
#define CONST_GAMEEVENT_TEAM_ROUND_WON                    5
#define CONST_GAMEEVENT_TEAM_ROUND_STALEMATE              52
#define CONST_GAMEEVENT_PLAYER_LOGIN                      100
#define CONST_GAMEEVENT_PLAYER_LOGOUT                     101
#define CONST_GAMEEVENT_PLAYER_SPAWN                      102
#define CONST_GAMEEVENT_PLAYER_MATCH_WON                  103
#define CONST_GAMEEVENT_PLAYER_KILL                       104
#define CONST_GAMEEVENT_PLAYER_LOCATION_POLL              105
#define CONST_GAMEEVENT_PLAYER_TEAMCHANGE                 106
#define CONST_GAMEEVENT_PLAYER_KILL_STREAK                107
#define CONST_GAMEEVENT_PLAYER_DEATH                      108
#define CONST_GAMEEVENT_PLAYER_ROUND_WON                  109
```

```
#define CONST_GAMEEVENT_PLAYER_ROUND_STALEMATE              110
#define CONST_GAMEEVENT_WEAPON_DAMAGE                        150
#define CONST_GAMEEVENT_WEAPON_DAMAGE_MELEE                  151
#define CONST_GAMEEVENT_WEAPON_FIRED                         152
#define CONST_GAMEEVENT_PLAYER_KILL_NORMAL                   200
#define CONST_GAMEEVENT_GENERIC_PARAM_LIST_START             300
#define CONST_GAMEEVENT_GENERIC_PARAM_LIST_END               400
#define CONST_GAMEEVENT_GAME_SPECIFIC                        1000
#define CONST_GAMEEVENT_MAX_EVENTID                          0x0000FFFF
#define CONST_SCENEFILTER_None                               0x00000000
#define CONST_SCENEFILTER_IncludeTransient                   0x00000001
#define CONST_SCENEFILTER_InputProcessorOnly                 0x00000002
#define CONST_SCENEFILTER_PausersOnly                        0x00000004
#define CONST_SCENEFILTER_PrimitiveUsersOnly                 0x00000008
#define CONST_SCENEFILTER_UsesPostProcessing                 0x00000010
#define CONST_SCENEFILTER_ReceivesFocus                      0x00000020
#define CONST_SCENEFILTER_Any                                0xFFFFFFFF
#define CONST_MAX_INSTANCES_PER_CLASS                        5
#define CONST_MAX_BOOKMARK_NUMBER                            10
#define CONST_MAXCLIENTUPDATEINTERVAL                        0.25
#define CONST_CLIENTADJUSTUPDATECOST                         180.0
#define CONST_MAXNEARZEROVELOCITYSQUARED                     9.0
#define CONST_MAXPOSITIONERRORSQUARED                        3.0
#define CONST_MAX_AIGROUP_NUMBER                             10
#define CONST_LINECHECK_GRANULARITY                          768.f
#define CONST_NUM_PATHFINDING_PARAMS                         9
#define CONST_NumBreadCrumbs                                 10
#define CONST_PROCBUILDING_VERSION                           1
#define CONST_ROOF_MINZ                                      0.7

/*
#
=============================================================================
=========== #
# Enums
#
=============================================================================
=========== #
*/

// Enum Engine._Types_Engine.EGameClipsAvailability
enum class EGameClipsAvailability : uint8_t
{
GameClipsAvailability_Pending                    = 0,
GameClipsAvailability_NotAvailable               = 1,
GameClipsAvailability_Available                  = 2,
GameClipsAvailability_END                        = 3
};

// Enum Engine._Types_Engine.EGameClipsRecording
enum class EGameClipsRecording : uint8_t
{
GameClipsRecording_Pending                       = 0,
GameClipsRecording_NotRecording                  = 1,
GameClipsRecording_Recording           = 2,
GameClipsRecording_END               = 3
};

// Enum Engine._Types_Engine.EGameClipsConnectionStatus
enum class EGameClipsConnectionStatus : uint8_t
{
GameClipsConnectionStatus_Pending         = 0,
GameClipsConnectionStatus_NotConnected       = 1,
GameClipsConnectionStatus_Connected        = 2,
GameClipsConnectionStatus_END            = 3
};

// Enum Engine._Types_Engine.EGameClipsConnection
enum class EGameClipsConnection : uint8_t
{
GameClipsConnection_Epic             = 0,
GameClipsConnection_Postparty           = 1,
GameClipsConnection_END               = 2
};

// Enum Engine._Types_Engine.EGameClipsClipStatus
enum class EGameClipsClipStatus : uint8_t
{
GameClipsClipStatus_None             = 0,
GameClipsClipStatus_Creating          = 1,
GameClipsClipStatus_Uploading          = 2,
GameClipsClipStatus_Completed           = 3,
GameClipsClipStatus_Failed           = 4,
GameClipsClipStatus_END              = 5
};

// Enum Engine._Types_Engine.EGameClipsStateChangeType
enum class EGameClipsStateChangeType : uint8_t
{
GameClipsStateChangeType_Availability     = 0,
GameClipsStateChangeType_Recording         = 1,
GameClipsStateChangeType_ClipStatus        = 2,
GameClipsStateChangeType_UserStatus        = 3,
GameClipsStateChangeType_END             = 4
};

// Enum Engine._Types_Engine.EGameClipsMaskStatus
enum class EGameClipsMaskStatus : uint8_t
{
GameClipsMaskStatus_Show               = 0,
GameClipsMaskStatus_Hide              = 1,
GameClipsMaskStatus_END               = 2
};

// Enum Engine.Actor.EActorMetricsType
enum class EActorMetricsType : uint8_t
{
METRICS_VERTS                    = 0,
METRICS_TRIS                               = 1,
METRICS_SECTIONS                               = 2,
METRICS_END                                = 3
};

// Enum Engine.Actor.EMoveDir
enum class EMoveDir : uint8_t
{
MD_Stationary                            = 0,
MD_Forward                               = 1,
MD_Backward                               = 2,
MD_Left                              = 3,
MD_Right                              = 4,
MD_Up                                = 5,
MD_Down                                = 6,
MD_END                                = 7
};

// Enum Engine.Actor.EPhysics
enum class EPhysics : uint8_t
{
PHYS_None                                = 0,
PHYS_Walking                              = 1,
PHYS_Falling                             = 2,
PHYS_Swimming                              = 3,
PHYS_Flying                              = 4,
PHYS_Rotating                             = 5,
PHYS_Projectile                            = 6,
PHYS_Interpolating                          = 7,
PHYS_Spider                             = 8,
PHYS_RigidBody                            = 9,
PHYS_SoftBody                             = 10,
PHYS_NavMeshWalking                          = 11,
PHYS_Unused                              = 12,
PHYS_Custom                              = 13,
PHYS_END                                = 14
};

// Enum Engine.Actor.EForceMode
enum class EForceMode : uint8_t
{
ForceMode_Force                            = 0,
ForceMode_Impulse                            = 1,
ForceMode_Velocity                          = 2,
ForceMode_SmoothImpulse                         = 3,
ForceMode_SmoothVelocity                        = 4,
ForceMode_Acceleration                        = 5,
ForceMode_END                               = 6
};

// Enum Engine.Actor.ECollisionType
enum class ECollisionType : uint8_t
{
COLLIDE_CustomDefault                         = 0,
COLLIDE_NoCollision                    = 1,
COLLIDE_BlockAll                       = 2,
COLLIDE_BlockWeapons                   = 3,
COLLIDE_TouchAll                       = 4,
COLLIDE_TouchWeapons                   = 5,
COLLIDE_BlockAllButWeapons             = 6,
COLLIDE_TouchAllButWeapons             = 7,
COLLIDE_BlockWeaponsKickable           = 8,
COLLIDE_END                            = 9
};

// Enum Engine.Actor.ETravelType
enum class ETravelType : uint8_t
{
TRAVEL_Absolute                        = 0,
TRAVEL_Partial                         = 1,
TRAVEL_Relative                        = 2,
TRAVEL_END                             = 3
};

// Enum Engine.Actor.EDoubleClickDir
enum class EDoubleClickDir : uint8_t
{
DCLICK_None                            = 0,
DCLICK_Left                            = 1,
DCLICK_Right                           = 2,
DCLICK_Forward                         = 3,
DCLICK_Back                            = 4,
DCLICK_Active                          = 5,
DCLICK_Done                            = 6,
DCLICK_END                             = 7
};

// Enum Engine.Actor.ENetRole
enum class ENetRole : uint8_t
{
ROLE_None                              = 0,
ROLE_SimulatedProxy                    = 1,
ROLE_AutonomousProxy                   = 2,
ROLE_Authority                         = 3,
ROLE_END                               = 4
};

// Enum Engine.PrimitiveComponent.GJKResult
enum class EGJKResult : uint8_t
{
GJK_Intersect                          = 0,
GJK_NoIntersection                     = 1,
GJK_Fail                               = 2,
GJK_END                                = 3
};

// Enum Engine.Scene.EDetailMode
enum class EDetailMode : uint8_t
```

```
{
DM_Low                              = 0,
DM_Medium                             = 1,
DM_High                             = 2,
DM_END                              = 3
};

// Enum Engine.Scene.ESceneDepthPriorityGroup
enum class ESceneDepthPriorityGroup : uint8_t
{
SDPG_UnrealEdBackground                     = 0,
SDPG_World                                 = 1,
SDPG_Foreground                              = 2,
SDPG_UnrealEdForeground                     = 3,
SDPG_PostProcess                             = 4,
SDPG_END                                    = 5
};

// Enum Engine.PrimitiveComponent.ERadialImpulseFalloff
enum class ERadialImpulseFalloff : uint8_t
{
RIF_Constant                               = 0,
RIF_Linear                                = 1,
RIF_END                                   = 2
};

// Enum Engine.PrimitiveComponent.ERBCollisionChannel
enum class ERBCollisionChannel : uint8_t
{
RBCC_Default                                = 0,
RBCC_Nothing                                 = 1,
RBCC_Pawn                                   = 2,
RBCC_Vehicle                                = 3,
RBCC_Water                                  = 4,
RBCC_GameplayPhysics                          = 5,
RBCC_EffectPhysics                           = 6,
RBCC_Ball                                  = 7,
RBCC_VehicleBlocker                           = 8,
RBCC_BallBlocker                             = 9,
RBCC_Untitled4                              = 10,
RBCC_Cloth                                = 11,
RBCC_FluidDrain                              = 12,
RBCC_SoftBody                                = 13,
RBCC_FracturedMeshPart                        = 14,
RBCC_BlockingVolume                          = 15,
RBCC_DeadPawn                               = 16,
RBCC_Clothing                               = 17,
RBCC_ClothingCollision                        = 18,
RBCC_END                                    = 19
};

// Enum Engine.Pylon.ENavMeshEdgeType
enum class ENavMeshEdgeType : uint8_t
{
```

```
NAVEDGE_Normal                              = 0,
NAVEDGE_Mantle                              = 1,
NAVEDGE_Coverslip                           = 2,
NAVEDGE_SwatTurn                            = 3,
NAVEDGE_DropDown                            = 4,
NAVEDGE_PathObject                          = 5,
NAVEDGE_BackRefDummy                        = 6,
NAVEDGE_Jump                                = 7,
NAVEDGE_END                                 = 8
};

// Enum Engine.AkBank.EBankLoadStatus
enum class EBankLoadStatus : uint8_t
{
BLS_Unloaded                                = 0,
BLS_Pending                                 = 1,
BLS_Loaded                                  = 2,
BLS_TimedOut                                = 3,
BLS_END                                     = 4
};

// Enum Engine.AmbientOcclusionEffect.EAmbientOcclusionQuality
enum class EAmbientOcclusionQuality : uint8_t
{
AO_High                                     = 0,
AO_Medium                                   = 1,
AO_Low                                      = 2,
AO_END                                      = 3
};

// Enum Engine.Brush.ECsgOper
enum class ECsgOper : uint8_t
{
CSG_Active                                  = 0,
CSG_Add                                     = 1,
CSG_Subtract                                = 2,
CSG_Intersect                               = 3,
CSG_Deintersect                             = 4,
CSG_END                                     = 5
};

// Enum Engine.ReverbVolume.ReverbPreset
enum class EReverbPreset : uint8_t
{
REVERB_Default                              = 0,
REVERB_Bathroom                             = 1,
REVERB_StoneRoom                            = 2,
REVERB_Auditorium                           = 3,
REVERB_ConcertHall                          = 4,
REVERB_Cave                                 = 5,
REVERB_Hallway                              = 6,
REVERB_StoneCorridor                        = 7,
REVERB_Alley                                = 8,
REVERB_Forest                               = 9,
REVERB_City                              = 10,
REVERB_Mountains                         = 11,
REVERB_Quarry                            = 12,
REVERB_Plain                             = 13,
REVERB_ParkingLot                        = 14,
REVERB_SewerPipe                         = 15,
REVERB_Underwater                        = 16,
REVERB_SmallRoom                         = 17,
REVERB_MediumRoom                        = 18,
REVERB_LargeRoom                         = 19,
REVERB_MediumHall                        = 20,
REVERB_LargeHall                         = 21,
REVERB_Plate                             = 22,
REVERB_END                               = 23
};

// Enum Engine.EngineTypes.EInputPlatformType
enum class EInputPlatformType : uint8_t
{
IPT_PC                                   = 0,
IPT                                      = 1,
IPT_PS3                                  = 2,
IPT_PS4                                  = 3,
IPT_XBOX_ONE                             = 4,
IPT_NNX                                  = 5,
IPT_NNX_SINGLE                           = 6,
IPT_END                                  = 7
};

// Enum Engine.EngineTypes.EPathFindingError
enum class EPathFindingError : uint8_t
{
PATHERROR_STARTPOLYNOTFOUND              = 0,
PATHERROR_GOALPOLYNOTFOUND               = 1,
PATHERROR_ANCHORPYLONNOTFOUND            = 2,
PATHERROR_NOPATHFOUND                    = 3,
PATHERROR_COMPUTEVALIDFINALDEST_FAIL     = 4,
PATHERROR_GETNEXTMOVELOCATION_FAIL       = 5,
PATHERROR_MOVETIMEOUT                    = 6,
PATHERROR_END                            = 7
};

// Enum Engine.AudioDevice.EDebugState
enum class EDebugState : uint8_t
{
DEBUGSTATE_None                          = 0,
DEBUGSTATE_IsolateDryAudio               = 1,
DEBUGSTATE_IsolateReverb                 = 2,
DEBUGSTATE_TestLPF                       = 3,
DEBUGSTATE_TestStereoBleed               = 4,
DEBUGSTATE_TestLFEBleed                  = 5,
DEBUGSTATE_DisableLPF                    = 6,
DEBUGSTATE_DisableRadio                  = 7,
DEBUGSTATE_END                           = 8
```

```
};

// Enum Engine.AudioDevice.ESoundClassName
enum class ESoundClassName : uint8_t
{
Master                                    = 0,
ESoundClassName_END                               = 1
};

// Enum Engine.AudioDevice.ETTSSpeaker
enum class ETTSSpeaker : uint8_t
{
TTSSPEAKER_Paul                           = 0,
TTSSPEAKER_Harry                          = 1,
TTSSPEAKER_Frank                          = 2,
TTSSPEAKER_Dennis                          = 3,
TTSSPEAKER_Kit                           = 4,
TTSSPEAKER_Betty                          = 5,
TTSSPEAKER_Ursula                          = 6,
TTSSPEAKER_Rita                           = 7,
TTSSPEAKER_Wendy                          = 8,
TTSSPEAKER_END                            = 9
};

// Enum Engine.SoundNodeAttenuation.ESoundDistanceCalc
enum class ESoundDistanceCalc : uint8_t
{
SOUNDDISTANCE_Normal                          = 0,
SOUNDDISTANCE_InfiniteXYPlane                    = 1,
SOUNDDISTANCE_InfiniteXZPlane                    = 2,
SOUNDDISTANCE_InfiniteYZPlane                    = 3,
SOUNDDISTANCE_END                             = 4
};

// Enum Engine.SoundNodeAttenuation.SoundDistanceModel
enum class ESoundDistanceModel : uint8_t
{
ATTENUATION_Linear                          = 0,
ATTENUATION_Logarithmic                       = 1,
ATTENUATION_Inverse                         = 2,
ATTENUATION_LogReverse                       = 3,
ATTENUATION_NaturalSound                      = 4,
ATTENUATION_END                            = 5
};

// Enum Engine.PlatformInterfaceBase.EPlatformInterfaceDataType
enum class EPlatformInterfaceDataType : uint8_t
{
PIDT_None                              = 0,
PIDT_Int                            = 1,
PIDT_Float                            = 2,
PIDT_String                            = 3,
PIDT_Object                            = 4,
PIDT_Custom                             = 5,
PIDT_QWord                                 = 6,
PIDT_END                                 = 7
};

// Enum Engine.AnimSequence.AnimationCompressionFormat
enum class EAnimationCompressionFormat : uint8_t
{
ACF_None                                  = 0,
ACF_Float96NoW                                = 1,
ACF_Fixed48NoW                                 = 2,
ACF_IntervalFixed32NoW                            = 3,
ACF_Fixed32NoW                                 = 4,
ACF_Float32NoW                                = 5,
ACF_Identity                              = 6,
ACF_END                                  = 7
};

// Enum Engine.AnimSequence.AnimationKeyFormat
enum class EAnimationKeyFormat : uint8_t
{
AKF_ConstantKeyLerp                              = 0,
AKF_VariableKeyLerp                             = 1,
AKF_PerTrackCompression                             = 2,
AKF_END                                  = 3
};

// Enum Engine.AnimNode.ESliderType
enum class ESliderType : uint8_t
{
ST_1D                                  = 0,
ST_2D                                  = 1,
ST_END                                  = 2
};

// Enum Engine.AnimNode_MultiBlendPerBone.EWeightCheck
enum class EWeightCheck : uint8_t
{
EWC_AnimNodeSlotNotPlaying                         = 0,
EWC_END                                    = 1
};

// Enum Engine.AnimNode_MultiBlendPerBone.EBlendType
enum class EBlendType : uint8_t
{
EBT_ParentBoneSpace                              = 0,
EBT_MeshSpace                                 = 1,
EBT_END                                  = 2
};

// Enum Engine.AnimNodeAimOffset.EAnimAimDir
enum class EAnimAimDir : uint8_t
{
ANIMAIM_LEFTUP                                  = 0,
ANIMAIM_CENTERUP                                  = 1,
ANIMAIM_RIGHTUP                              = 2,
ANIMAIM_LEFTCENTER                             = 3,
ANIMAIM_CENTERCENTER                            = 4,
ANIMAIM_RIGHTCENTER                            = 5,
ANIMAIM_LEFTDOWN                              = 6,
ANIMAIM_CENTERDOWN                              = 7,
ANIMAIM_RIGHTDOWN                              = 8,
ANIMAIM_END                                = 9
};

// Enum Engine.AnimNodeAimOffset.EAimID
enum class EAimID : uint8_t
{
EAID_LeftUp                              = 0,
EAID_LeftDown                              = 1,
EAID_RightUp                              = 2,
EAID_RightDown                              = 3,
EAID_ZeroUp                              = 4,
EAID_ZeroDown                              = 5,
EAID_ZeroLeft                              = 6,
EAID_ZeroRight                              = 7,
EAID_CellLU                              = 8,
EAID_CellCU                              = 9,
EAID_CellRU                              = 10,
EAID_CellLC                              = 11,
EAID_CellCC                              = 12,
EAID_CellRC                              = 13,
EAID_CellLD                              = 14,
EAID_CellCD                              = 15,
EAID_CellRD                              = 16,
EAID_END                                = 17
};

// Enum Engine.AnimNodeBlendByBase.EBaseBlendType
enum class EBaseBlendType : uint8_t
{
BBT_ByActorTag                              = 0,
BBT_ByActorClass                              = 1,
BBT_END                                = 2
};

// Enum Engine.AnimNodeSequence.ERootRotationOption
enum class ERootRotationOption : uint8_t
{
RRO_Default                              = 0,
RRO_Discard                              = 1,
RRO_Extract                              = 2,
RRO_END                                = 3
};

// Enum Engine.AnimNodeSequence.ERootBoneAxis
enum class ERootBoneAxis : uint8_t
{
RBA_Default                              = 0,
RBA_Discard                       = 1,
RBA_Translate                      = 2,
RBA_END                          = 3
};

// Enum Engine.SkeletalMeshComponent.EPhysBodyOp
enum class EPhysBodyOp : uint8_t
{
PBO_None                          = 0,
PBO_Term                         = 1,
PBO_Disable                       = 2,
PBO_END                          = 3
};

// Enum Engine.SkeletalMeshComponent.EBoneVisibilityStatus
enum class EBoneVisibilityStatus : uint8_t
{
BVS_HiddenByParent                     = 0,
BVS_Visible                      = 1,
BVS_ExplicitlyHidden                 = 2,
BVS_END                          = 3
};

// Enum Engine.SkeletalMeshComponent.EFaceFXRegOp
enum class EFaceFXRegOp : uint8_t
{
FXRO_Add                         = 0,
FXRO_Multiply                      = 1,
FXRO_Replace                       = 2,
FXRO_END                          = 3
};

// Enum Engine.SkeletalMeshComponent.EFaceFXBlendMode
enum class EFaceFXBlendMode : uint8_t
{
FXBM_Overwrite                      = 0,
FXBM_Additive                      = 1,
FXBM_END                          = 2
};

// Enum Engine.SkeletalMeshComponent.EInstanceWeightUsage
enum class EInstanceWeightUsage : uint8_t
{
IWU_PartialSwap                     = 0,
IWU_FullSwap                       = 1,
IWU_END                          = 2
};

// Enum Engine.SkeletalMeshComponent.EAnimRotationOnly
enum class EAnimRotationOnly : uint8_t
{
EARO_AnimSet                        = 0,
EARO_ForceEnabled                     = 1,
EARO_ForceDisabled                    = 2,
EARO_END                                  = 3
};

// Enum Engine.SkeletalMeshComponent.ERootMotionRotationMode
enum class ERootMotionRotationMode : uint8_t
{
RMRM_Ignore                               = 0,
RMRM_RotateActor                              = 1,
RMRM_END                                  = 2
};

// Enum Engine.SkeletalMeshComponent.ERootMotionMode
enum class ERootMotionMode : uint8_t
{
RMM_Translate                             = 0,
RMM_Velocity                             = 1,
RMM_Ignore                               = 2,
RMM_Accel                                = 3,
RMM_Relative                             = 4,
RMM_END                                  = 5
};

// Enum Engine.SkeletalMeshComponent.EMaxDistanceScaleMode
enum class EMaxDistanceScaleMode : uint8_t
{
MDSM_Multiply                             = 0,
MDSM_Substract                              = 1,
MDSM_END                                  = 2
};

// Enum Engine.EngineTypes.EBlendMode
enum class EBlendMode : uint8_t
{
BLEND_Opaque                               = 0,
BLEND_Masked                               = 1,
BLEND_Translucent                            = 2,
BLEND_Additive                            = 3,
BLEND_Modulate                              = 4,
BLEND_ModulateAndAdd                            = 5,
BLEND_SoftMasked                             = 6,
BLEND_AlphaComposite                            = 7,
BLEND_DitheredTranslucent                         = 8,
BLEND_END                                 = 9
};

// Enum Engine.EngineTypes.EMaterialLightingModel
enum class EMaterialLightingModel : uint8_t
{
MLM_Phong                                 = 0,
MLM_NonDirectional                              = 1,
MLM_Unlit                              = 2,
MLM_SHPRT                                 = 3,
MLM_Custom                                = 4,
MLM_Anisotropic                              = 5,
MLM_END                                        = 6
};

// Enum Engine.EngineTypes.EMaterialTessellationMode
enum class EMaterialTessellationMode : uint8_t
{
MTM_NoTessellation                             = 0,
MTM_FlatTessellation                           = 1,
MTM_PNTriangles                                = 2,
MTM_END                                        = 3
};

// Enum Engine.EngineTypes.EMobileValueSource
enum class EMobileValueSource : uint8_t
{
MVS_Constant                                   = 0,
MVS_VertexColorRed                             = 1,
MVS_VertexColorGreen                           = 2,
MVS_VertexColorBlue                            = 3,
MVS_VertexColorAlpha                           = 4,
MVS_BaseTextureRed                             = 5,
MVS_BaseTextureGreen                           = 6,
MVS_BaseTextureBlue                            = 7,
MVS_BaseTextureAlpha                           = 8,
MVS_MaskTextureRed                             = 9,
MVS_MaskTextureGreen                           = 10,
MVS_MaskTextureBlue                            = 11,
MVS_MaskTextureAlpha                           = 12,
MVS_NormalTextureAlpha                         = 13,
MVS_EmissiveTextureRed                         = 14,
MVS_EmissiveTextureGreen                       = 15,
MVS_EmissiveTextureBlue                        = 16,
MVS_EmissiveTextureAlpha                       = 17,
MVS_END                                        = 18
};

// Enum Engine.EngineTypes.EMobileTextureBlendFactorSource
enum class EMobileTextureBlendFactorSource : uint8_t
{
MTBFS_VertexColor                              = 0,
MTBFS_MaskTexture                              = 1,
MTBFS_END                                      = 2
};

// Enum Engine.EngineTypes.EMobileTexCoordsSource
enum class EMobileTexCoordsSource : uint8_t
{
MTCS_TexCoords0                                = 0,
MTCS_TexCoords1                                = 1,
MTCS_TexCoords2                                = 2,
MTCS_TexCoords3                                = 3,
MTCS_END                                       = 4
};
```

```
// Enum Engine.EngineTypes.EMobileAlphaValueSource
enum class EMobileAlphaValueSource : uint8_t
{
MAVS_DiffuseTextureAlpha                    = 0,
MAVS_MaskTextureRed                         = 1,
MAVS_MaskTextureGreen                       = 2,
MAVS_MaskTextureBlue                        = 3,
MAVS_END                            = 4
};

// Enum Engine.EngineTypes.EMobileColorMultiplySource
enum class EMobileColorMultiplySource : uint8_t
{
MCMS_None                            = 0,
MCMS_BaseTextureRed                         = 1,
MCMS_BaseTextureGreen                       = 2,
MCMS_BaseTextureBlue                        = 3,
MCMS_BaseTextureAlpha                       = 4,
MCMS_MaskTextureRed                         = 5,
MCMS_MaskTextureGreen                       = 6,
MCMS_MaskTextureBlue                        = 7,
MCMS_MaskTextureAlpha                       = 8,
MCMS_END                            = 9
};

// Enum Engine.EngineTypes.EMobileEmissiveColorSource
enum class EMobileEmissiveColorSource : uint8_t
{
MECS_EmissiveTexture                      = 0,
MECS_BaseTexture                        = 1,
MECS_Constant                         = 2,
MECS_END                            = 3
};

// Enum Engine.EngineTypes.EMobileEnvironmentBlendMode
enum class EMobileEnvironmentBlendMode : uint8_t
{
MEBM_Add                            = 0,
MEBM_Lerp                           = 1,
MEBM_END                            = 2
};

// Enum Engine.EngineTypes.EMobileSpecularMask
enum class EMobileSpecularMask : uint8_t
{
MSM_Constant                          = 0,
MSM_Luminance                           = 1,
MSM_DiffuseRed                          = 2,
MSM_DiffuseGreen                          = 3,
MSM_DiffuseBlue                         = 4,
MSM_DiffuseAlpha                          = 5,
MSM_MaskTextureRGB                          = 6,
MSM_MaskTextureRed                         = 7,
MSM_MaskTextureGreen                        = 8,
MSM_MaskTextureBlue                          = 9,
MSM_MaskTextureAlpha                         = 10,
MSM_END                                  = 11
};

// Enum Engine.EngineTypes.EMobileAmbientOcclusionSource
enum class EMobileAmbientOcclusionSource : uint8_t
{
MAOS_Disabled                            = 0,
MAOS_VertexColorRed                          = 1,
MAOS_VertexColorGreen                         = 2,
MAOS_VertexColorBlue                         = 3,
MAOS_VertexColorAlpha                        = 4,
MAOS_END                                = 5
};

// Enum Engine.EngineTypes.ELightingBuildQuality
enum class ELightingBuildQuality : uint8_t
{
Quality_Preview                          = 0,
Quality_Medium                            = 1,
Quality_High                            = 2,
Quality_Production                          = 3,
Quality_NoGlobalIllumination                    = 4,
Quality_END                              = 5
};

// Enum Engine.Pawn.EPathSearchType
enum class EPathSearchType : uint8_t
{
PST_Default                              = 0,
PST_Breadth                               = 1,
PST_NewBestPathTo                            = 2,
PST_Constraint                             = 3,
PST_END                                 = 4
};

// Enum
Engine.DynamicLightEnvironmentComponent.EDynamicLightEnvironmentBoundsMethod
enum class EDynamicLightEnvironmentBoundsMethod : uint8_t
{
DLEB_OwnerComponents                         = 0,
DLEB_ManualOverride                         = 1,
DLEB_ActiveComponents                         = 2,
DLEB_END                                 = 3
};

// Enum Engine.ApexDestructibleAsset.EImpactDamageOverride
enum class EImpactDamageOverride : uint8_t
{
IDO_None                                = 0,
IDO_On                               = 1,
IDO_Off                              = 2,
IDO_END                                = 3
```

```
};

// Enum Engine.ApexDestructibleDamageParameters.EDamageParameterOverrideMode
enum class EDamageParameterOverrideMode : uint8_t
{
DPOM_Absolute                                = 0,
DPOM_Multiplier                              = 1,
DPOM_END                                     = 2
};

// Enum Engine.Camera.ECameraAnimPlaySpace
enum class ECameraAnimPlaySpace : uint8_t
{
CAPS_CameraLocal                             = 0,
CAPS_World                                   = 1,
CAPS_UserDefined                             = 2,
CAPS_END                                     = 3
};

// Enum Engine.Camera.EViewTargetBlendFunction
enum class EViewTargetBlendFunction : uint8_t
{
VTBlend_Linear                               = 0,
VTBlend_Cubic                                = 1,
VTBlend_EaseIn                               = 2,
VTBlend_EaseOut                              = 3,
VTBlend_EaseInOut                            = 4,
VTBlend_MidwayStep                           = 5,
VTBlend_END                                  = 6
};

// Enum Engine.DOFEffect.EFocusType
enum class EFocusType : uint8_t
{
FOCUS_Distance                               = 0,
FOCUS_Position                               = 1,
FOCUS_END                                    = 2
};

// Enum Engine.CameraShake.EInitialOscillatorOffset
enum class EInitialOscillatorOffset : uint8_t
{
EOO_OffsetRandom                             = 0,
EOO_OffsetZero                               = 1,
EOO_END                                      = 2
};

// Enum Engine.Canvas.ECanvasBlendMode
enum class ECanvasBlendMode : uint8_t
{
BLEND_CANVAS_Opaque                          = 0,
BLEND_CANVAS_Masked                          = 1,
BLEND_CANVAS_Translucent                     = 2,
BLEND_CANVAS_Additive                        = 3,
BLEND_CANVAS_Modulate                    = 4,
BLEND_CANVAS_ModulateAndAdd                 = 5,
BLEND_CANVAS_SoftMasked                  = 6,
BLEND_CANVAS_AlphaComposite                 = 7,
BLEND_CANVAS_DitheredTranslucent             = 8,
BLEND_CANVAS_AlphaOnly                   = 9,
BLEND_CANVAS_END                      = 10
};

// Enum Engine.Texture.TextureCompressionSettings
enum class ETextureCompressionSettings : uint8_t
{
TC_Default                        = 0,
TC_Normalmap                        = 1,
TC_Displacementmap                      = 2,
TC_NormalmapAlpha                       = 3,
TC_Grayscale                        = 4,
TC_HighDynamicRange                      = 5,
TC_OneBitAlpha                       = 6,
TC_NormalmapUncompressed                   = 7,
TC_NormalmapBC5                       = 8,
TC_NormalmapSwizzle                     = 9,
TC_OneBitMonochrome                     = 10,
TC_SimpleLightmapModification                = 11,
TC_VectorDisplacementmap                  = 12,
TC_BC7                          = 13,
TC_END                          = 14
};

// Enum Engine.Texture.EPixelFormat
enum class EPixelFormat : uint8_t
{
PF_Unknown                        = 0,
PF_A32B32G32R32F                      = 1,
PF_A8R8G8B8                        = 2,
PF_G8                          = 3,
PF_G16                          = 4,
PF_DXT1                          = 5,
PF_DXT3                          = 6,
PF_DXT5                          = 7,
PF_UYVY                          = 8,
PF_FloatRGB                        = 9,
PF_FloatRGBA                        = 10,
PF_DepthStencil                      = 11,
PF_ShadowDepth                        = 12,
PF_FilteredShadowDepth                    = 13,
PF_R32F                          = 14,
PF_G16R16                         = 15,
PF_G16R16F                         = 16,
PF_G16R16F_FILTER                      = 17,
PF_G32R32F                         = 18,
PF_A2B10G10R10                        = 19,
PF_A2R10G10B10                        = 20,
PF_A16B16G16R16                       = 21,
PF_D24                            = 22,
PF_R16F                            = 23,
PF_R16F_FILTER                         = 24,
PF_BC5                            = 25,
PF_V8U8                            = 26,
PF_A1                            = 27,
PF_FloatR11G11B10                        = 28,
PF_A4R4G4B4                           = 29,
PF_R5G6B5                            = 30,
PF_G8R8                            = 31,
PF_R32_UINT                          = 32,
PF_ASTC_4x4                           = 33,
PF_ASTC_6x6                           = 34,
PF_ASTC_8x8                           = 35,
PF_ASTC_10x10                          = 36,
PF_ASTC_12x12                          = 37,
PF_BC7                            = 38,
PF_COUNT                            = 39,
PF_END                            = 40
};

// Enum Engine.Texture.TextureFilter
enum class ETextureFilter : uint8_t
{
TF_Nearest                          = 0,
TF_Linear                          = 1,
TF_END                            = 2
};

// Enum Engine.Texture.TextureAddress
enum class ETextureAddress : uint8_t
{
TA_Wrap                            = 0,
TA_Clamp                            = 1,
TA_Mirror                          = 2,
TA_END                             = 3
};

// Enum Engine.Texture.TextureGroup
enum class ETextureGroup : uint8_t
{
TEXTUREGROUP_World                          = 0,
TEXTUREGROUP_WorldNormalMap                        = 1,
TEXTUREGROUP_WorldSpecular                      = 2,
TEXTUREGROUP_Character                        = 3,
TEXTUREGROUP_CharacterNormalMap                     = 4,
TEXTUREGROUP_CharacterSpecular                    = 5,
TEXTUREGROUP_Weapon                           = 6,
TEXTUREGROUP_WeaponNormalMap                        = 7,
TEXTUREGROUP_WeaponSpecular                     = 8,
TEXTUREGROUP_Vehicle                        = 9,
TEXTUREGROUP_VehicleNormalMap                      = 10,
TEXTUREGROUP_VehicleSpecular                    = 11,
TEXTUREGROUP_Cinematic                        = 12,
TEXTUREGROUP_Effects                              = 13,
TEXTUREGROUP_EffectsNotFiltered                   = 14,
TEXTUREGROUP_Skybox                               = 15,
TEXTUREGROUP_UI                                   = 16,
TEXTUREGROUP_Lightmap                             = 17,
TEXTUREGROUP_RenderTarget                         = 18,
TEXTUREGROUP_MobileFlattened                      = 19,
TEXTUREGROUP_ProcBuilding_Face                    = 20,
TEXTUREGROUP_ProcBuilding_LightMap                = 21,
TEXTUREGROUP_Shadowmap                            = 22,
TEXTUREGROUP_ColorLookupTable                     = 23,
TEXTUREGROUP_Terrain_Heightmap                    = 24,
TEXTUREGROUP_Terrain_Weightmap                    = 25,
TEXTUREGROUP_ImageBasedReflection                 = 26,
TEXTUREGROUP_Bokeh                                = 27,
TEXTUREGROUP_END                                  = 28
};

// Enum Engine.Texture.TextureMipGenSettings
enum class ETextureMipGenSettings : uint8_t
{
TMGS_FromTextureGroup                             = 0,
TMGS_SimpleAverage                                = 1,
TMGS_Sharpen0                                     = 2,
TMGS_Sharpen1                                     = 3,
TMGS_Sharpen2                                     = 4,
TMGS_Sharpen3                                     = 5,
TMGS_Sharpen4                                     = 6,
TMGS_Sharpen5                                     = 7,
TMGS_Sharpen6                                     = 8,
TMGS_Sharpen7                                     = 9,
TMGS_Sharpen8                                     = 10,
TMGS_Sharpen9                                     = 11,
TMGS_Sharpen10                                    = 12,
TMGS_NoMipmaps                                    = 13,
TMGS_LeaveExistingMips                            = 14,
TMGS_Blur1                                        = 15,
TMGS_Blur2                                        = 16,
TMGS_Blur3                                        = 17,
TMGS_Blur4                                        = 18,
TMGS_Blur5                                        = 19,
TMGS_END                                          = 20
};

// Enum Engine.Texture.ETextureMipCount
enum class ETextureMipCount : uint8_t
{
TMC_ResidentMips                                  = 0,
TMC_AllMips                                       = 1,
TMC_AllMipsBiased                                 = 2,
TMC_END                                           = 3
};

// Enum Engine.CloudSaveSystem.SaveDataVersionSupport
```

```
enum class ESaveDataVersionSupport : uint8_t
{
SaveDataVersionSupportLessThenEqual             = 0,
SaveDataVersionSupportEqual                    = 1,
SaveDataVersionSupportAny                      = 2,
SaveDataVersionSupport_END                     = 3
};

// Enum Engine.CloudSaveSystem.SaveSlotOperationEnum
enum class ESaveSlotOperationEnum : uint8_t
{
SSO_SET                                 = 0,
SSO_GET                                 = 1,
SSO_DELETE                                 = 2,
SSO_END                                 = 3
};

// Enum Engine.CloudStorageBase.ECloudStorageDelegate
enum class ECloudStorageDelegate : uint8_t
{
CSD_KeyValueReadComplete                       = 0,
CSD_KeyValueWriteComplete                      = 1,
CSD_ValueChanged                               = 2,
CSD_DocumentQueryComplete                        = 3,
CSD_DocumentReadComplete                         = 4,
CSD_DocumentWriteComplete                        = 5,
CSD_DocumentConflictDetected                     = 6,
CSD_END                                 = 7
};

// Enum Engine.Interaction.ETouchType
enum class ETouchType : uint8_t
{
Touch_Began                                = 0,
Touch_Moved                                 = 1,
Touch_Stationary                             = 2,
Touch_Ended                                = 3,
Touch_Cancelled                              = 4,
Touch_END                                 = 5
};

// Enum Engine.CoverGroup.ECoverGroupFillAction
enum class ECoverGroupFillAction : uint8_t
{
CGFA_Overwrite                               = 0,
CGFA_Add                                  = 1,
CGFA_Remove                                  = 2,
CGFA_Clear                                  = 3,
CGFA_Cylinder                                = 4,
CGFA_END                                  = 5
};

// Enum Engine.CoverLink.ECoverLocationDescription
enum class ECoverLocationDescription : uint8_t
```

```
{
CoverDesc_None                          = 0,
CoverDesc_InWindow                      = 1,
CoverDesc_InDoorway                     = 2,
CoverDesc_BehindCar                     = 3,
CoverDesc_BehindTruck                   = 4,
CoverDesc_OnTruck                       = 5,
CoverDesc_BehindBarrier                 = 6,
CoverDesc_BehindColumn                  = 7,
CoverDesc_BehindCrate                   = 8,
CoverDesc_BehindWall                    = 9,
CoverDesc_BehindStatue                  = 10,
CoverDesc_BehindSandbags                = 11,
CoverDesc_END                           = 12
};

// Enum Engine.CoverLink.ECoverType
enum class ECoverType : uint8_t
{
CT_None                                 = 0,
CT_Standing                             = 1,
CT_MidLevel                             = 2,
CT_END                                  = 3
};

// Enum Engine.CoverLink.ECoverAction
enum class ECoverAction : uint8_t
{
CA_Default                              = 0,
CA_BlindLeft                            = 1,
CA_BlindRight                           = 2,
CA_LeanLeft                             = 3,
CA_LeanRight                            = 4,
CA_PopUp                                = 5,
CA_BlindUp                              = 6,
CA_PeekLeft                             = 7,
CA_PeekRight                            = 8,
CA_PeekUp                               = 9,
CA_END                                  = 10
};

// Enum Engine.CoverLink.ECoverDirection
enum class ECoverDirection : uint8_t
{
CD_Default                              = 0,
CD_Left                                 = 1,
CD_Right                                = 2,
CD_Up                                   = 3,
CD_END                                  = 4
};

// Enum Engine.CoverLink.EFireLinkID
enum class EFireLinkID : uint8_t
{
```

```
FLI_FireLink                                  = 0,
FLI_RejectedFireLink                                  = 1,
FLI_END                                       = 2
};

// Enum Engine.StaticMeshComponent.ELightmapModificationFunction
enum class ELightmapModificationFunction : uint8_t
{
MLMF_Modulate                                 = 0,
MLMF_ModulateAlpha                                 = 1,
MLMF_END                                      = 2
};

// Enum Engine.DateTime.ETimeZone
enum class ETimeZone : uint8_t
{
TZ_UTC                                        = 0,
TZ_Local                                      = 1,
TZ_END                                        = 2
};

// Enum Engine.DecalComponent.EFilterMode
enum class EFilterMode : uint8_t
{
FM_None                                       = 0,
FM_Ignore                                     = 1,
FM_Affect                                     = 2,
FM_END                                        = 3
};

// Enum Engine.DecalComponent.EDecalTransform
enum class EDecalTransform : uint8_t
{
DecalTransform_OwnerAbsolute                  = 0,
DecalTransform_OwnerRelative                  = 1,
DecalTransform_SpawnRelative                  = 2,
DecalTransform_END                            = 3
};

// Enum Engine.MaterialInterface.EMaterialUsage
enum class EMaterialUsage : uint8_t
{
MATUSAGE_SkeletalMesh                         = 0,
MATUSAGE_FracturedMeshes                         = 1,
MATUSAGE_ParticleSprites                      = 2,
MATUSAGE_BeamTrails                           = 3,
MATUSAGE_ParticleSubUV                        = 4,
MATUSAGE_SpeedTree                            = 5,
MATUSAGE_StaticLighting                       = 6,
MATUSAGE_GammaCorrection                         = 7,
MATUSAGE_LensFlare                            = 8,
MATUSAGE_InstancedMeshParticles                  = 9,
MATUSAGE_FluidSurface                         = 10,
MATUSAGE_Decals                               = 11,
MATUSAGE_MaterialEffect                          = 12,
MATUSAGE_MorphTargets                            = 13,
MATUSAGE_FogVolumes                              = 14,
MATUSAGE_RadialBlur                              = 15,
MATUSAGE_InstancedMeshes                         = 16,
MATUSAGE_SplineMesh                              = 17,
MATUSAGE_ScreenDoorFade                          = 18,
MATUSAGE_APEXMesh                                = 19,
MATUSAGE_Terrain                                 = 20,
MATUSAGE_Landscape                               = 21,
MATUSAGE_MobileLandscape                         = 22,
MATUSAGE_END                                     = 23
};

// Enum Engine.LightComponent.EShadowFilterQuality
enum class EShadowFilterQuality : uint8_t
{
SFQ_Low                                = 0,
SFQ_Medium                             = 1,
SFQ_High                               = 2,
SFQ_END                                = 3
};

// Enum Engine.LightComponent.EShadowProjectionTechnique
enum class EShadowProjectionTechnique : uint8_t
{
ShadowProjTech_Default                           = 0,
ShadowProjTech_PCF                               = 1,
ShadowProjTech_VSM                               = 2,
ShadowProjTech_BPCF_Low                          = 3,
ShadowProjTech_BPCF_Medium                       = 4,
ShadowProjTech_BPCF_High                         = 5,
ShadowProjTech_END                               = 6
};

// Enum Engine.LightComponent.ELightShadowMode
enum class ELightShadowMode : uint8_t
{
LightShadow_Normal                               = 0,
LightShadow_Modulate                             = 1,
LightShadow_ModulateBetter                       = 2,
LightShadow_END                                  = 3
};

// Enum Engine.LightComponent.ELightAffectsClassification
enum class ELightAffectsClassification : uint8_t
{
LAC_USER_SELECTED                                = 0,
LAC_DYNAMIC_AFFECTING                            = 1,
LAC_STATIC_AFFECTING                             = 2,
LAC_DYNAMIC_AND_STATIC_AFFECTING                 = 3,
LAC_END                                          = 4
};
```

```
// Enum Engine.DistributionFloatParameterBase.DistributionParamMode
enum class EDistributionParamMode : uint8_t
{
DPM_Normal                               = 0,
DPM_Abs                                  = 1,
DPM_Direct                               = 2,
DPM_END                                  = 3
};

// Enum Engine.DOFAndBloomEffect.EDOFQuality
enum class EDOFQuality : uint8_t
{
DOFQuality_Low                           = 0,
DOFQuality_Medium                          = 1,
DOFQuality_High                          = 2,
DOFQuality_END                           = 3
};

// Enum Engine.DOFAndBloomEffect.EDOFType
enum class EDOFType : uint8_t
{
DOFType_SimpleDOF                          = 0,
DOFType_ReferenceDOF                         = 1,
DOFType_BokehDOF                           = 2,
DOFType_END                              = 3
};

// Enum Engine.OnlineSubsystem.EOnlineFriendState
enum class EOnlineFriendState : uint8_t
{
OFS_Offline                              = 0,
OFS_Online                               = 1,
OFS_Away                                 = 2,
OFS_Busy                                 = 3,
OFS_Unknown                                = 4,
OFS_END                                  = 5
};

// Enum Engine.OnlineSubsystem.EFeaturePrivilege
enum class EFeaturePrivilege : uint8_t
{
FP_OnlinePlay                            = 0,
FP_CommunicationText                          = 1,
FP_CommunicationVideo                         = 2,
FP_CommunicationVoice                         = 3,
FP_ShareUserCreatedContent                      = 4,
FP_PurchaseContent                          = 5,
FP_ViewPlayerProfile                       = 6,
FP_ShowPresenceInformation                      = 7,
FP_RecordDVRClips                           = 8,
FP_CloudStorage                           = 9,
FP_PremiumContent                           = 10,
FP_PremiumVideoContent                         = 11,
FP_BrowseInternet                          = 12,
FP_SocialNetworkSharing                     = 13,
FP_KinectSharing                            = 14,
FP_FitnessUpload                            = 15,
FP_Crossplay                              = 16,
FP_END                                  = 17
};

// Enum Engine.OnlineSubsystem.ELoginStatus
enum class ELoginStatus : uint8_t
{
LS_NotLoggedIn                              = 0,
LS_UsingLocalProfile                         = 1,
LS_LoggedIn                               = 2,
LS_END                                  = 3
};

// Enum Engine.OnlineSubsystem.AvatarSize
enum class EAvatarSize : uint8_t
{
AvatarSize_Small                            = 0,
AvatarSize_Medium                             = 1,
AvatarSize_Large                            = 2,
AvatarSize_END                              = 3
};

// Enum Engine.OnlineSubsystem.EPinGrantResult
enum class EPinGrantResult : uint8_t
{
PinGrantResult_Success                         = 0,
PinGrantResult_Expired                        = 1,
PinGrantResult_END                           = 2
};

// Enum Engine.OnlineSubsystem.EFeaturePrivilegeLevel
enum class EFeaturePrivilegeLevel : uint8_t
{
FPL_Disabled                              = 0,
FPL_Premium                                = 1,
FPL_EnabledFriendsOnly                          = 2,
FPL_Enabled                               = 3,
FPL_Offline                             = 4,
FPL_END                                 = 5
};

// Enum Engine.OnlineSubsystem.EPS4ErrorDialog
enum class EPS4ErrorDialog : uint8_t
{
PS4_ERROR_SIGNED_OUT                            = 0,
PS4_ERROR_SIGNED_END                            = 1
};

// Enum Engine.OnlineSubsystem.EPS4DisplayMode
enum class EPS4DisplayMode : uint8_t
{
```

```
PS4DM_CATEGORY                                  = 0,
PS4DM_PRODUCT                                 = 1,
PS4DM_PRODUCT_CODE                              = 2,
PS4DM_CHECKOUT                                  = 3,
PS4DM_DOWNLOADLIST                              = 4,
PS4DM_PLUS                                    = 5,
PS4DM_END                                     = 6
};

// Enum Engine.OnlineSubsystem.ENATType
enum class ENATType : uint8_t
{
NAT_Unknown                                   = 0,
NAT_Open                                    = 1,
NAT_Moderate                                  = 2,
NAT_Strict                                  = 3,
NAT_END                                     = 4
};

// Enum Engine.OnlineSubsystem.EOnlineServerConnectionStatus
enum class EOnlineServerConnectionStatus : uint8_t
{
OSCS_NotConnected                               = 0,
OSCS_Connected                                = 1,
OSCS_ConnectionDropped                          = 2,
OSCS_NoNetworkConnection                        = 3,
OSCS_ServiceUnavailable                        = 4,
OSCS_UpdateRequired                            = 5,
OSCS_ServersTooBusy                            = 6,
OSCS_DuplicateLoginDetected                     = 7,
OSCS_InvalidUser                              = 8,
OSCS_InvalidResponse                           = 9,
OSCS_TooYoung                                 = 10,
OSCS_EpicAccountRequired                        = 11,
OSCS_EpicAccountLinkingFailed                   = 12,
OSCS_EpicDOBRequired                           = 13,
OSCS_EpicParentEmailRequired                    = 14,
OSCS_EpicConfirmDisplayNameRequired             = 15,
OSCS_EpicUnsupportedCorrectiveActionRequired    = 16,
OSCS_END                                      = 17
};

// Enum Engine.OnlineSubsystem.ESessionMemberStatus
enum class ESessionMemberStatus : uint8_t
{
DSMS_Reserved                                 = 0,
DSMS_Inactive                                = 1,
DSMS_Ready                                   = 2,
DSMS_Active                                  = 3,
DSMS_END                                     = 4
};

// Enum Engine.OnlineSubsystem.EOnlineNewsType
enum class EOnlineNewsType : uint8_t
```

```
{
ONT_Unknown                 = 0,
ONT_GameNews                  = 1,
ONT_ContentAnnouncements          = 2,
ONT_Misc                  = 3,
ONT_END                   = 4
};

// Enum Engine.OnlineSubsystem.EInventoryItemState
enum class EInventoryItemState : uint8_t
{
IIS_Unknown                = 0,
IIS_All                = 1,
IIS_Enabled               = 2,
IIS_Suspended               = 3,
IIS_Expired              = 4,
IIS_Canceled               = 5,
IIS_END                  = 6
};

// Enum Engine.OnlineSubsystem.EMediaItemType
enum class EMediaItemType : uint8_t
{
MIT_Unknown                 = 0,
MIT_Game                   = 1,
MIT_Application             = 2,
MIT_GameContent               = 3,
MIT_GameConsumable              = 4,
MIT_Subscription            = 5,
MIT_END                   = 6
};

// Enum Engine.OnlineSubsystem.EOnlineEnumerationReadState
enum class EOnlineEnumerationReadState : uint8_t
{
OERS_NotStarted              = 0,
OERS_InProgress              = 1,
OERS_Done                  = 2,
OERS_Failed                = 3,
OERS_END                   = 4
};

// Enum Engine.OnlineSubsystem.EOnlineGameState
enum class EOnlineGameState : uint8_t
{
OGS_NoSession                = 0,
OGS_Pending                 = 1,
OGS_Starting               = 2,
OGS_InProgress              = 3,
OGS_Ending                 = 4,
OGS_Ended                  = 5,
OGS_END                   = 6
};
```

```
// Enum Engine.OnlineSubsystem.ECatalogSortOrder
enum class ECatalogSortOrder : uint8_t
{
CSO_FreeAndPaidCountDaily                    = 0,
CSO_PaidCountAllTime                      = 1,
CSO_PaidCountDail                       = 2,
CSO_DigitalReleaseDate                    = 3,
CSO_ReleaseDate                         = 4,
CSO_UserRatings                         = 5,
CSO_END                             = 6
};

// Enum Engine.OnlineSubsystem.EAchievementUnlockType
enum class EAchievementUnlockType : uint8_t
{
AUT_Unknown                           = 0,
AUT_All                           = 1,
AUT_Persistent                       = 2,
AUT_Challenge                         = 3,
AUT_END                             = 4
};

// Enum Engine.OnlineSubsystem.EAchievementMediaAssetType
enum class EAchievementMediaAssetType : uint8_t
{
AMAT_Unknown                            = 0,
AMAT_Icon                           = 1,
AMAT_Art                           = 2,
AMAT_END                             = 3
};

// Enum Engine.OnlineSubsystem.EAchievementParticipationType
enum class EAchievementParticipationType : uint8_t
{
APT_Unknown                           = 0,
APT_Individual                       = 1,
APT_Group                           = 2,
APT_END                             = 3
};

// Enum Engine.OnlineSubsystem.EAchievementProgressState
enum class EAchievementProgressState : uint8_t
{
APS_Unknown                           = 0,
APS_Achieved                         = 1,
APS_NotStarted                        = 2,
APS_InProgress                        = 3,
APS_END                             = 4
};

// Enum Engine.OnlineSubsystem.EAchievementRewardType
enum class EAchievementRewardType : uint8_t
{
ART_Unknown                           = 0,
ART_Gamerscore                    = 1,
ART_InApp                    = 2,
ART_Art                    = 3,
ART_END                    = 4
};

// Enum Engine.Settings.ESettingsDataType
enum class ESettingsDataType : uint8_t
{
SDT_Empty                    = 0,
SDT_Int32                    = 1,
SDT_Int64                    = 2,
SDT_Double                    = 3,
SDT_String                    = 4,
SDT_Float                    = 5,
SDT_Blob                    = 6,
SDT_DateTime                    = 7,
SDT_END                    = 8
};

// Enum Engine.Settings.EOnlineDataAdvertisementType
enum class EOnlineDataAdvertisementType : uint8_t
{
ODAT_DontAdvertise                    = 0,
ODAT_OnlineService                    = 1,
ODAT_QoS                    = 2,
ODAT_OnlineServiceAndQoS                    = 3,
ODAT_END                    = 4
};

// Enum Engine.OnlineSubsystem.EOnlineFileType
enum class EOnlineFileType : uint8_t
{
OFT_Unknown                    = 0,
OFT_Binary                    = 1,
OFT_Json                    = 2,
OFT_END                    = 3
};

// Enum Engine.OnlineSubsystem.EOnlineAccountCreateStatus
enum class EOnlineAccountCreateStatus : uint8_t
{
OACS_CreateSuccessful                    = 0,
OACS_UnknownError                    = 1,
OACS_InvalidUserName                    = 2,
OACS_InvalidPassword                    = 3,
OACS_InvalidUniqueUserName                    = 4,
OACS_UniqueUserNameInUse                    = 5,
OACS_ServiceUnavailable                    = 6,
OACS_END                    = 7
};

// Enum Engine.OnlineSubsystem.ELanBeaconState
enum class ELanBeaconState : uint8_t
```

```
{
LANB_NotUsingLanBeacon                         = 0,
LANB_Hosting                                   = 1,
LANB_Searching                                 = 2,
LANB_END                                       = 3
};

// Enum Engine.OnlineSubsystem.EOnlineContentType
enum class EOnlineContentType : uint8_t
{
OCT_Downloaded                                 = 0,
OCT_SaveGame                                   = 1,
OCT_END                                        = 2
};

// Enum Engine.OnlineSubsystem.EOnlineCreateGameSessionFlag
enum class EOnlineCreateGameSessionFlag : uint8_t
{
OCGSF_New                                      = 0,
OCGSF_ReSubmit                                 = 1,
OCGSF_END                                      = 2
};

// Enum Engine.OnlineSubsystem.EOnlineAccountTier
enum class EOnlineAccountTier : uint8_t
{
OAT_Unknown                                    = 0,
OAT_NewUser                                    = 1,
OAT_Silver                                     = 2,
OAT_Gold                                       = 3,
OAT_FamilyGold                                 = 4,
OAT_END                                        = 5
};

// Enum Engine.OnlineSubsystem.ENetworkNotificationPosition
enum class ENetworkNotificationPosition : uint8_t
{
NNP_TopLeft                                    = 0,
NNP_TopCenter                                  = 1,
NNP_TopRight                                   = 2,
NNP_CenterLeft                                 = 3,
NNP_Center                                     = 4,
NNP_CenterRight                                = 5,
NNP_BottomLeft                                 = 6,
NNP_BottomCenter                               = 7,
NNP_BottomRight                                = 8,
NNP_END                                        = 9
};

// Enum Engine.OnlineSubsystem.EWordFilterCensorship
enum class EWordFilterCensorship : uint8_t
{
WordFilterCensorship_Uncensored                = 0,
WordFilterCensorship_Censored                  = 1,
WordFilterCensorship_Evil              = 2,
WordFilterCensorship_END                = 3
};

// Enum Engine.OnlineSubsystem.ECommunicationMethod
enum class ECommunicationMethod : uint8_t
{
COMM_Text                          = 0,
COMM_Voice                         = 1,
COMM_END                           = 2
};

// Enum Engine.ParticleSystemComponent.EParticleSysParamType
enum class EParticleSysParamType : uint8_t
{
PSPT_None                         = 0,
PSPT_Scalar                       = 1,
PSPT_ScalarRand                      = 2,
PSPT_Vector                       = 3,
PSPT_VectorRand                      = 4,
PSPT_Color                        = 5,
PSPT_Actor                        = 6,
PSPT_Material                      = 7,
PSPT_Mesh                         = 8,
PSPT_END                          = 9
};

// Enum Engine.ParticleSystemComponent.ParticleReplayState
enum class EParticleReplayState : uint8_t
{
PRS_Disabled                       = 0,
PRS_Capturing                      = 1,
PRS_Replaying                      = 2,
PRS_END                          = 3
};

// Enum Engine.ParticleSystemComponent.EParticleEventType
enum class EParticleEventType : uint8_t
{
EPET_Any                         = 0,
EPET_Spawn                         = 1,
EPET_Death                        = 2,
EPET_Collision                     = 3,
EPET_WorldAttractorCollision             = 4,
EPET_Kismet                        = 5,
EPET_END                          = 6
};

// Enum Engine.ParticleSystem.ParticleSystemLODMethod
enum class EParticleSystemLODMethod : uint8_t
{
PARTICLESYSTEMLODMETHOD_Automatic            = 0,
PARTICLESYSTEMLODMETHOD_DirectSet           = 1,
PARTICLESYSTEMLODMETHOD_ActivateAutomatic      = 2,
PARTICLESYSTEMLODMETHOD_END                    = 3
};

// Enum Engine.Engine.EBasicLoadResult
enum class EBasicLoadResult : uint8_t
{
BasicLoadResult_Pending                        = 0,
BasicLoadResult_Success                        = 1,
BasicLoadResult_FileNotFound                   = 2,
BasicLoadResult_WrongVersion                   = 3,
BasicLoadResult_FileCorrupt                    = 4,
BasicLoadResult_MountCorrupt                   = 5,
BasicLoadResult_UserNotSignedIn                = 6,
BasicLoadResult_Error                          = 7,
BasicLoadResult_END                            = 8
};

// Enum Engine.Engine.EImageFilterOptions
enum class EImageFilterOptions : uint8_t
{
ImageFilter_NONE                               = 0,
ImageFilter_SEPIA                              = 1,
ImageFilter_GRAYSCALE                          = 2,
ImageFilter_SCENE_DEPTH                        = 3,
ImageFilter_GLOWY                              = 4,
ImageFilter_SUBPAR                             = 5,
ImageFilter_SOBEL                              = 6,
ImageFilter_SOBEL_STRIKER                      = 7,
ImageFilter_COMIC_BOOK                         = 8,
ImageFilter_TOON                               = 9,
ImageFilter_OUTLINES                           = 10,
ImageFilter_CHROMATIC                          = 11,
ImageFilter_NIGHT_VISION                       = 12,
ImageFilter_END_FILTERS                        = 13,
ImageFilter_END                                = 14
};

// Enum Engine.Engine.EReplayFXProperties
enum class EReplayFXProperties : uint8_t
{
ReplayFX_Distance                              = 0,
ReplayFX_Kernel                                = 1,
ReplayFX_MaxFar                                = 2,
ReplayFX_MaxNear                               = 3,
ReplayFX_END                                   = 4
};

// Enum Engine.Engine.ETransitionType
enum class ETransitionType : uint8_t
{
TT_None                                        = 0,
TT_Paused                                      = 1,
TT_Loading                                     = 2,
TT_Saving                                      = 3,
TT_Connecting                  = 4,
TT_Precaching                  = 5,
TT_END                       = 6
};

// Enum Engine.FacebookIntegration.EFacebookIntegrationDelegate
enum class EFacebookIntegrationDelegate : uint8_t
{
FID_AuthorizationComplete           = 0,
FID_FacebookRequestComplete            = 1,
FID_DialogComplete               = 2,
FID_FriendsListComplete            = 3,
FID_END                      = 4
};

// Enum Engine.FileWriter.FWFileType
enum class EFWFileType : uint8_t
{
FWFT_Log                      = 0,
FWFT_Stats                    = 1,
FWFT_HTML                      = 2,
FWFT_User                     = 3,
FWFT_Debug                     = 4,
FWFT_END                      = 5
};

// Enum Engine.FluidInfluenceComponent.EInfluenceType
enum class EInfluenceType : uint8_t
{
Fluid_Flow                   = 0,
Fluid_Raindrops                = 1,
Fluid_Wave                    = 2,
Fluid_Sphere                  = 3,
Fluid_END                    = 4
};

// Enum Engine.FontImportOptions.EFontImportCharacterSet
enum class EFontImportCharacterSet : uint8_t
{
FontICS_Default                = 0,
FontICS_Ansi                  = 1,
FontICS_Symbol                  = 2,
FontICS_END                    = 3
};

// Enum Engine.ForceFeedbackWaveform.EWaveformFunction
enum class EWaveformFunction : uint8_t
{
WF_Constant                    = 0,
WF_LinearIncreasing              = 1,
WF_LinearDecreasing               = 2,
WF_Sin0to90                    = 3,
WF_Sin90to180                   = 4,
WF_Sin0to180                   = 5,
WF_Noise                                  = 6,
WF_END                                    = 7
};

// Enum Engine.GameEngine.EFullyLoadPackageType
enum class EFullyLoadPackageType : uint8_t
{
FULLYLOAD_Map                                     = 0,
FULLYLOAD_Game_PreLoadClass                       = 1,
FULLYLOAD_Game_PostLoadClass                      = 2,
FULLYLOAD_Always                                  = 3,
FULLYLOAD_Mutator                                 = 4,
FULLYLOAD_END                                     = 5
};

// Enum Engine.GameInfo.EStandbyType
enum class EStandbyType : uint8_t
{
STDBY_Rx                                  = 0,
STDBY_Tx                                  = 1,
STDBY_BadPing                             = 2,
STDBY_END                                 = 3
};

// Enum Engine.GameplayEvents.EGameStatGroups
enum class EGameStatGroups : uint8_t
{
GSG_EngineStats                           = 0,
GSG_Game                                  = 1,
GSG_Team                                  = 2,
GSG_Player                                = 3,
GSG_Weapon                                = 4,
GSG_Damage                                = 5,
GSG_Projectile                            = 6,
GSG_Pawn                                  = 7,
GSG_GameSpecific                          = 8,
GSG_Aggregate                             = 9,
GSG_END                                   = 10
};

// Enum Engine.WorldInfo.EHostMigrationProgress
enum class EHostMigrationProgress : uint8_t
{
HostMigration_None                                = 0,
HostMigration_FindingNewHost                      = 1,
HostMigration_MigratingAsHost                     = 2,
HostMigration_MigratingAsClient                   = 3,
HostMigration_ClientTravel                        = 4,
HostMigration_HostReadyToTravel                   = 5,
HostMigration_Failed                              = 6,
HostMigration_END                                 = 7
};

// Enum Engine.WorldInfo.EConsoleType
```

```
enum class EConsoleType : uint8_t
{
CONSOLE_Any                              = 0,
CONSOLE_Xbox360                           = 1,
CONSOLE_PS3                              = 2,
CONSOLE_Mobile                            = 3,
CONSOLE_IPhone                            = 4,
CONSOLE_Android                           = 5,
CONSOLE_WiiU                             = 6,
CONSOLE_Flash                            = 7,
CONSOLE_Mobile_Retina                       = 8,
CONSOLE_PC                              = 9,
CONSOLE_Kindle                           = 10,
CONSOLE_PS4                             = 11,
CONSOLE_Dingo                            = 12,
CONSOLE_NNX                             = 13,
CONSOLE_Lockhart                          = 14,
CONSOLE_Anaconda                           = 15,
CONSOLE_Prospero                          = 16,
CONSOLE_Scorpio                           = 17,
CONSOLE_END                             = 18
};

// Enum Engine.WorldInfo.EPreferredLightmapType
enum class EPreferredLightmapType : uint8_t
{
EPLT_Default                          = 0,
EPLT_Directional                        = 1,
EPLT_Simple                           = 2,
EPLT_END                             = 3
};

// Enum Engine.WorldInfo.EVisibilityAggressiveness
enum class EVisibilityAggressiveness : uint8_t
{
VIS_LeastAggressive                        = 0,
VIS_ModeratelyAggressive                     = 1,
VIS_MostAggressive                         = 2,
VIS_Max                              = 3
};

// Enum Engine.WorldInfo.ENetMode
enum class ENetMode : uint8_t
{
NM_Standalone                            = 0,
NM_DedicatedServer                          = 1,
NM_ListenServer                           = 2,
NM_Client                              = 3,
NM_END                                = 4
};

// Enum Engine.SeqAct_ControlMovieTexture.EMovieControlType
enum class EMovieControlType : uint8_t
{
```

```
MCT_Play                                = 0,
MCT_Stop                                = 1,
MCT_Pause                               = 2,
MCT_END                                 = 3
};

// Enum Engine.Settings.EPropertyValueMappingType
enum class EPropertyValueMappingType : uint8_t
{
PVMT_RawValue                           = 0,
PVMT_PredefinedValues                   = 1,
PVMT_Ranged                             = 2,
PVMT_IdMapped                           = 3,
PVMT_END                                = 4
};

// Enum Engine.OnlineGameSearch.EOnlineGameSearchComparisonType
enum class EOnlineGameSearchComparisonType : uint8_t
{
OGSCT_Equals                            = 0,
OGSCT_NotEquals                         = 1,
OGSCT_GreaterThan                       = 2,
OGSCT_GreaterThanEquals                 = 3,
OGSCT_LessThan                          = 4,
OGSCT_LessThanEquals                    = 5,
OGSCT_END                               = 6
};

// Enum Engine.OnlineGameSearch.EOnlineGameSearchEntryType
enum class EOnlineGameSearchEntryType : uint8_t
{
OGSET_Property                          = 0,
OGSET_LocalizedSetting                  = 1,
OGSET_ObjectProperty                    = 2,
OGSET_END                               = 3
};

// Enum Engine.OnlineGameSearch.EOnlineGameSearchSortType
enum class EOnlineGameSearchSortType : uint8_t
{
OGSSO_Ascending                         = 0,
OGSSO_Descending                        = 1,
OGSSO_END                               = 2
};

// Enum Engine.PlayerController.EProgressMessageType
enum class EProgressMessageType : uint8_t
{
PMT_Clear                               = 0,
PMT_Information                         = 1,
PMT_AdminMessage                        = 2,
PMT_DownloadProgress                    = 3,
PMT_ConnectionFailure                   = 4,
PMT_PeerConnectionFailure               = 5,
PMT_PeerHostMigrationFailure            = 6,
PMT_SocketFailure                  = 7,
PMT_Reconnect                     = 8,
PMT_AntiCheatKick                  = 9,
PMT_END                        = 10
};

// Enum Engine.PlayerController.EInputMatchAction
enum class EInputMatchAction : uint8_t
{
IMA_GreaterThan                   = 0,
IMA_LessThan                     = 1,
IMA_END                        = 2
};

// Enum Engine.PlayerController.EInputTypes
enum class EInputTypes : uint8_t
{
IT_XAxis                      = 0,
IT_YAxis                     = 1,
IT_END                       = 2
};

// Enum Engine.GameViewportClient.ESplitScreenType
enum class ESplitScreenType : uint8_t
{
eSST_NONE                       = 0,
eSST_2P_HORIZONTAL                    = 1,
eSST_2P_VERTICAL                    = 2,
eSST_3P_FAVOR_TOP                    = 3,
eSST_3P_FAVOR_BOTTOM                    = 4,
eSST_3P_FAVOR_SIDELEFT                  = 5,
eSST_3P_FAVOR_SIDERIGHT                 = 6,
eSST_4P                       = 7,
eSST_END                       = 8
};

// Enum Engine.GameViewportClient.ESafeZoneType
enum class ESafeZoneType : uint8_t
{
eSZ_TOP                        = 0,
eSZ_BOTTOM                       = 1,
eSZ_LEFT                       = 2,
eSZ_RIGHT                      = 3,
eSZ_END                        = 4
};

// Enum Engine.GameViewportClient.ESetMode
enum class ESetMode : uint8_t
{
SetMode_Toggle                    = 0,
SetMode_Enable                    = 1,
SetMode_Disable                   = 2,
SetMode_END                      = 3
```

```
};

// Enum Engine.InAppMessageBase.EInAppMessageInterfaceDelegate
enum class EInAppMessageInterfaceDelegate : uint8_t
{
IAMD_InAppSMSUIComplete                  = 0,
IAMD_InAppEmailComplete                  = 1,
IAMD_END                                 = 2
};

// Enum Engine.InGameAdManager.EAdManagerDelegate
enum class EAdManagerDelegate : uint8_t
{
AMD_ClickedBanner                        = 0,
AMD_UserClosedAd                         = 1,
AMD_END                                  = 2
};

// Enum Engine.InstancedFoliageSettings.FoliageCullOption
enum class EFoliageCullOption : uint8_t
{
FOLIAGECULL_Cull                         = 0,
FOLIAGECULL_ScaleZ                       = 1,
FOLIAGECULL_ScaleXYZ                     = 2,
FOLIAGECULL_TranslateZ                   = 3,
FOLIAGECULL_END                          = 4
};

// Enum Engine.Interface_NavMeshPathObstacle.EEdgeHandlingStatus
enum class EEdgeHandlingStatus : uint8_t
{
EHS_AddedBothDirs                        = 0,
EHS_Added0to1                            = 1,
EHS_Added1to0                            = 2,
EHS_AddedNone                            = 3,
EHS_END                                  = 4
};

// Enum Engine.InterpTrack.ETrackActiveCondition
enum class ETrackActiveCondition : uint8_t
{
ETAC_Always                              = 0,
ETAC_GoreEnabled                         = 1,
ETAC_GoreDisabled                        = 2,
ETAC_END                                 = 3
};

// Enum Engine.InterpTrackHeadTracking.EHeadTrackingAction
enum class EHeadTrackingAction : uint8_t
{
EHTA_DisableHeadTracking                 = 0,
EHTA_EnableHeadTracking                  = 1,
EHTA_END                                 = 2
};
```

```
// Enum Engine.InterpTrackToggle.ETrackToggleAction
enum class ETrackToggleAction : uint8_t
{
ETTA_Off                           = 0,
ETTA_On                            = 1,
ETTA_Toggle                          = 2,
ETTA_Trigger                         = 3,
ETTA_END                            = 4
};

// Enum Engine.InterpTrackVisibility.EVisibilityTrackCondition
enum class EVisibilityTrackCondition : uint8_t
{
EVTC_Always                          = 0,
EVTC_GoreEnabled                        = 1,
EVTC_GoreDisabled                       = 2,
EVTC_END                            = 3
};

// Enum Engine.InterpTrackVisibility.EVisibilityTrackAction
enum class EVisibilityTrackAction : uint8_t
{
EVTA_Hide                           = 0,
EVTA_Show                            = 1,
EVTA_Toggle                          = 2,
EVTA_END                            = 3
};

// Enum Engine.InterpTrackMove.EInterpTrackMoveRotMode
enum class EInterpTrackMoveRotMode : uint8_t
{
IMR_Keyframed                         = 0,
IMR_LookAtGroup                         = 1,
IMR_Ignore                          = 2,
IMR_END                            = 3
};

// Enum Engine.InterpTrackMove.EInterpTrackMoveFrame
enum class EInterpTrackMoveFrame : uint8_t
{
IMF_World                          = 0,
IMF_RelativeToInitial                  = 1,
IMF_END                            = 2
};

// Enum Engine.InterpTrackMoveAxis.EInterpMoveAxis
enum class EInterpMoveAxis : uint8_t
{
AXIS_TranslationX                      = 0,
AXIS_TranslationY                      = 1,
AXIS_TranslationZ                      = 2,
AXIS_RotationX                        = 3,
AXIS_RotationY                        = 4,
AXIS_RotationZ                    = 5,
AXIS_END                        = 6
};

// Enum Engine.Landscape.ELandscapeSetupErrors
enum class ELandscapeSetupErrors : uint8_t
{
LSE_None                        = 0,
LSE_NoLandscapeInfo                   = 1,
LSE_CollsionXY                    = 2,
LSE_NoLayerInfo                    = 3,
LSE_END                         = 4
};

// Enum Engine.LandscapeGizmoActiveActor.ELandscapeGizmoType
enum class ELandscapeGizmoType : uint8_t
{
LGT_None                        = 0,
LGT_Height                       = 1,
LGT_Weight                       = 2,
LGT_END                         = 3
};

// Enum Engine.LevelGridVolume.LevelGridCellShape
enum class ELevelGridCellShape : uint8_t
{
LGCS_Box                        = 0,
LGCS_Hex                        = 1,
LGCS_END                         = 2
};

// Enum Engine.LevelStreamingVolume.EStreamingVolumeUsage
enum class EStreamingVolumeUsage : uint8_t
{
SVB_Loading                       = 0,
SVB_LoadingAndVisibility              = 1,
SVB_VisibilityBlockingOnLoad            = 2,
SVB_BlockingOnLoad                    = 3,
SVB_LoadingNotVisible                 = 4,
SVB_END                         = 5
};

// Enum Engine.OnlineAuthInterface.EAuthStatus
enum class EAuthStatus : uint8_t
{
AUS_NotStarted                     = 0,
AUS_Pending                       = 1,
AUS_Authenticated                   = 2,
AUS_Failed                      = 3,
AUS_END                         = 4
};

// Enum Engine.OnlineAuthInterface.EAuthTicketServiceRequester
enum class EAuthTicketServiceRequester : uint8_t
```

```
{
AuthTicketServiceRequester_Psynet                = 0,
AuthTicketServiceRequester_EpicOnlineServices      = 1,
AuthTicketServiceRequester_END                  = 2
};

// Enum Engine.MaterialExpressionAntialiasedTextureMask.ETextureColorChannel
enum class ETextureColorChannel : uint8_t
{
TCC_Red                              = 0,
TCC_Green                             = 1,
TCC_Blue                             = 2,
TCC_Alpha                             = 3,
TCC_END                              = 4
};

// Enum Engine.MaterialExpressionCustom.ECustomMaterialOutputType
enum class ECustomMaterialOutputType : uint8_t
{
CMOT_Float1                            = 0,
CMOT_Float2                            = 1,
CMOT_Float3                            = 2,
CMOT_Float4                            = 3,
CMOT_END                              = 4
};

// Enum Engine.MaterialExpressionDepthOfFieldFunction.EDepthOfFieldFunctionValue
enum class EDepthOfFieldFunctionValue : uint8_t
{
TDOF_NearAndFarMask                          = 0,
TDOF_NearMask                            = 1,
TDOF_FarMask                            = 2,
TDOF_END                               = 3
};

// Enum Engine.MaterialExpressionFunctionInput.EFunctionInputType
enum class EFunctionInputType : uint8_t
{
FunctionInput_Scalar                        = 0,
FunctionInput_Vector2                        = 1,
FunctionInput_Vector3                        = 2,
FunctionInput_Vector4                        = 3,
FunctionInput_Texture2D                       = 4,
FunctionInput_TextureCube                      = 5,
FunctionInput_StaticBool                      = 6,
FunctionInput_END                          = 7
};

// Enum Engine.MaterialExpressionGameObjectParameter.EGameObjectShaderParameterType
enum class EGameObjectShaderParameterType : uint8_t
{
GOSPT_BallPositionAndSize                     = 0,
GOSPT_BallVelocity                          = 1,
GOSPT_BallSpeed                            = 2,
GOSPT_BallMaxSpeed                         = 3,
GOSPT_IsSuperSonic                         = 4,
GOSPT_IsBoosting                           = 5,
GOSPT_IsMakingContactWithBall              = 6,
GOSPT_IsCarOnGround                        = 7,
GOSPT_ObjectDistanceToBall                 = 8,
GOSPT_ObjectSpeed                          = 9,
GOSPT_ObjectVelocity                       = 10,
GOSPT_SuperSonicSpeed                      = 11,
GOSPT_CarMaxSpeed                          = 12,
GOSPT_END                                  = 13
};

// Enum Engine.MaterialExpressionGameParameter.EGameShaderParameterType
enum class EGameShaderParameterType : uint8_t
{
GSPT_Team0_ColorPrimary                    = 0,
GSPT_Team0_ColorSecondary                  = 1,
GSPT_Team1_ColorPrimary                    = 2,
GSPT_Team1_ColorSecondary                  = 3,
GSPT_Team0_ColorPrimaryFullBrightness      = 4,
GSPT_Team0_ColorSecondaryFullBrightness    = 5,
GSPT_Team1_ColorPrimaryFullBrightness      = 6,
GSPT_Team1_ColorSecondaryFullBrightness    = 7,
GSPT_END                                   = 8
};

// Enum Engine.MaterialExpressionLandscapeLayerBlend.ELandscapeLayerBlendType
enum class ELandscapeLayerBlendType : uint8_t
{
LB_AlphaBlend                              = 0,
LB_HeightBlend                             = 1,
LB_END                                     = 2
};

// Enum
Engine.MaterialExpressionMusicAnalysisParameter.EMusicAnalysisShaderParameterType
enum class EMusicAnalysisShaderParameterType : uint8_t
{
MAPT_Music_Band                            = 0,
MAPT_Music_Band01                          = 1,
MAPT_Music_Band02                          = 2,
MAPT_Music_Band03                          = 3,
MAPT_Music_Band04                          = 4,
MAPT_Music_Band05                          = 5,
MAPT_Music_Band06                          = 6,
MAPT_Music_Band07                          = 7,
MAPT_Music_Band08                          = 8,
MAPT_Music_Band09                          = 9,
MAPT_Music_LowFreq_Envelope                = 10,
MAPT_Music_HighFreq_Envelope               = 11,
MAPT_Music_Band_Smoothed                   = 12,
MAPT_Music_Band_Smoothed01                 = 13,
MAPT_Music_Band_Smoothed02                 = 14,
MAPT_Music_Band_Smoothed03                     = 15,
MAPT_Music_Band_Smoothed04                     = 16,
MAPT_Music_Band_Smoothed05                     = 17,
MAPT_Music_Band_Smoothed06                     = 18,
MAPT_Music_Band_Smoothed07                     = 19,
MAPT_Music_Band_Smoothed08                     = 20,
MAPT_Music_Band_Smoothed09                     = 21,
MAPT_Music_LowFreq_Envelope_Smoothed            = 22,
MAPT_Music_HighFreq_Envelope_Smoothed           = 23,
MAPT_Music_END                            = 24
};

// Enum Engine.MaterialExpressionPitchTekTextureSample.EPitchTekTextureType
enum class EPitchTekTextureType : uint8_t
{
PitchTek_ColorTexture                     = 0,
PitchTek_DataTexture                      = 1,
PitchTek_END                          = 2
};

// Enum Engine.MaterialExpressionSceneTexture.ESceneTextureType
enum class ESceneTextureType : uint8_t
{
SceneTex_Lighting                       = 0,
SceneTex_END                           = 1
};

// Enum Engine.MaterialExpressionTerrainLayerCoords.ETerrainCoordMappingType
enum class ETerrainCoordMappingType : uint8_t
{
TCMT_Auto                             = 0,
TCMT_XY                              = 1,
TCMT_XZ                              = 2,
TCMT_YZ                              = 3,
TCMT_END                              = 4
};

// Enum Engine.MaterialExpressionTransform.EMaterialVectorCoordTransform
enum class EMaterialVectorCoordTransform : uint8_t
{
TRANSFORM_World                          = 0,
TRANSFORM_View                          = 1,
TRANSFORM_Local                          = 2,
TRANSFORM_Tangent                          = 3,
TRANSFORM_END                           = 4
};

// Enum Engine.MaterialExpressionTransform.EMaterialVectorCoordTransformSource
enum class EMaterialVectorCoordTransformSource : uint8_t
{
TRANSFORMSOURCE_World                       = 0,
TRANSFORMSOURCE_Local                       = 1,
TRANSFORMSOURCE_Tangent                       = 2,
TRANSFORMSOURCE_View                       = 3,
TRANSFORMSOURCE_END                          = 4
};

// Enum Engine.MaterialExpressionTransformPosition.EMaterialPositionTransform
enum class EMaterialPositionTransform : uint8_t
{
TRANSFORMPOS_World                           = 0,
TRANSFORMPOS_END                             = 1
};

// Enum Engine.MaterialExpressionTransformPosition.EMaterialPositionTransformSource
enum class EMaterialPositionTransformSource : uint8_t
{
TRANSFORMPOSSOURCE_Local                     = 0,
TRANSFORMPOSSOURCE_PostProjection            = 1,
TRANSFORMPOSSOURCE_END                       = 2
};

// Enum Engine.MicroTransactionBase.EMicroTransactionDelegate
enum class EMicroTransactionDelegate : uint8_t
{
MTD_PurchaseQueryComplete                    = 0,
MTD_PurchaseComplete                         = 1,
MTD_END                                      = 2
};

// Enum Engine.MicroTransactionBase.EMicroTransactionResult
enum class EMicroTransactionResult : uint8_t
{
MTR_Succeeded                                = 0,
MTR_Failed                                   = 1,
MTR_Canceled                                 = 2,
MTR_RestoredFromServer                       = 3,
MTR_END                                      = 4
};

// Enum Engine.NxForceFieldGeneric.FFG_ForceFieldCoordinates
enum class EFFG_ForceFieldCoordinates : uint8_t
{
FFG_CARTESIAN                                = 0,
FFG_SPHERICAL                                = 1,
FFG_CYLINDRICAL                              = 2,
FFG_TOROIDAL                                 = 3,
FFG_END                                      = 4
};

// Enum Engine.NxGenericForceFieldBrush.FFB_ForceFieldCoordinates
enum class EFFB_ForceFieldCoordinates : uint8_t
{
FFB_CARTESIAN                                = 0,
FFB_SPHERICAL                                = 1,
FFB_CYLINDRICAL                              = 2,
FFB_TOROIDAL                                 = 3,
FFB_END                                      = 4
```

```
};

// Enum Engine.OnlineLobbySettings.ELobbyKickReason
enum class ELobbyKickReason : uint8_t
{
LKR_Unknown                              = 0,
LKR_Kicked                               = 1,
LKR_Full                                 = 2,
LKR_InGame                               = 3,
LKR_LeaveAction                          = 4,
LKR_OwnerDisappeared                     = 5,
LKR_ConnectionError                      = 6,
LKR_SignedOut                            = 7,
LKR_CrossplayDisabled                    = 8,
LKR_LeaderPartyUp                        = 9,
LKR_NotInTourParty                       = 10,
LKR_TourCheckingIn                       = 11,
LKR_END                                  = 12
};

// Enum Engine.OnlineLobbySettings.ELobbyVisibility
enum class ELobbyVisibility : uint8_t
{
LV_Public                                = 0,
LV_Friends                               = 1,
LV_Private                               = 2,
LV_Invisible                             = 3,
LV_END                                   = 4
};

// Enum Engine.OnlineLobbySettings.ELobbyDistance
enum class ELobbyDistance : uint8_t
{
LD_Best                                  = 0,
LD_Close                                 = 1,
LD_Far                                   = 2,
LD_Any                                   = 3,
LD_END                                   = 4
};

// Enum Engine.OnlinePlayerStorage.EOnlineProfilePropertyOwner
enum class EOnlineProfilePropertyOwner : uint8_t
{
OPPO_None                                = 0,
OPPO_OnlineService                       = 1,
OPPO_Game                                = 2,
OPPO_END                                 = 3
};

// Enum Engine.OnlinePlayerStorage.EOnlinePlayerStorageAsyncState
enum class EOnlinePlayerStorageAsyncState : uint8_t
{
OPAS_NotStarted                          = 0,
OPAS_Read                                = 1,
OPAS_Write                          = 2,
OPAS_Finished                       = 3,
OPAS_END                            = 4
};

// Enum Engine.OnlineProfileSettings.EProfileSettingID
enum class EProfileSettingID : uint8_t
{
PSI_Unknown                         = 0,
PSI_ControllerVibration             = 1,
PSI_YInversion                      = 2,
PSI_GamerCred                       = 3,
PSI_GamerRep                        = 4,
PSI_VoiceMuted                      = 5,
PSI_VoiceThruSpeakers               = 6,
PSI_VoiceVolume                     = 7,
PSI_GamerPictureKey                 = 8,
PSI_GamerMotto                      = 9,
PSI_GamerTitlesPlayed               = 10,
PSI_GamerAchievementsEarned         = 11,
PSI_GameDifficulty                  = 12,
PSI_ControllerSensitivity           = 13,
PSI_PreferredColor1                 = 14,
PSI_PreferredColor2                 = 15,
PSI_AutoAim                         = 16,
PSI_AutoCenter                      = 17,
PSI_MovementControl                 = 18,
PSI_RaceTransmission                = 19,
PSI_RaceCameraLocation              = 20,
PSI_RaceBrakeControl                = 21,
PSI_RaceAcceleratorControl          = 22,
PSI_GameCredEarned                  = 23,
PSI_GameAchievementsEarned          = 24,
PSI_EndLiveIds                      = 25,
PSI_ProfileVersionNum               = 26,
PSI_ProfileSaveCount                = 27,
PSI_END                             = 28
};

// Enum Engine.OnlineProfileSettings.EProfileDifficultyOptions
enum class EProfileDifficultyOptions : uint8_t
{
PDO_Normal                          = 0,
PDO_Easy                            = 1,
PDO_Hard                            = 2,
PDO_END                             = 3
};

// Enum Engine.OnlineProfileSettings.EProfileControllerSensitivityOptions
enum class EProfileControllerSensitivityOptions : uint8_t
{
PCSO_Medium                         = 0,
PCSO_Low                            = 1,
PCSO_High                           = 2,
PCSO_END                                    = 3
};

// Enum Engine.OnlineProfileSettings.EProfilePreferredColorOptions
enum class EProfilePreferredColorOptions : uint8_t
{
PPCO_None                                   = 0,
PPCO_Black                                  = 1,
PPCO_White                                  = 2,
PPCO_Yellow                                 = 3,
PPCO_Orange                                 = 4,
PPCO_Pink                                   = 5,
PPCO_Red                                    = 6,
PPCO_Purple                                 = 7,
PPCO_Blue                                   = 8,
PPCO_Green                                  = 9,
PPCO_Brown                                  = 10,
PPCO_Silver                                 = 11,
PPCO_END                                    = 12
};

// Enum Engine.OnlineProfileSettings.EProfileAutoAimOptions
enum class EProfileAutoAimOptions : uint8_t
{
PAAO_Off                                    = 0,
PAAO_On                                     = 1,
PAAO_END                                    = 2
};

// Enum Engine.OnlineProfileSettings.EProfileAutoCenterOptions
enum class EProfileAutoCenterOptions : uint8_t
{
PACO_Off                                    = 0,
PACO_On                                     = 1,
PACO_END                                    = 2
};

// Enum Engine.OnlineProfileSettings.EProfileMovementControlOptions
enum class EProfileMovementControlOptions : uint8_t
{
PMCO_L_Thumbstick                           = 0,
PMCO_R_Thumbstick                           = 1,
PMCO_END                                    = 2
};

// Enum Engine.OnlineProfileSettings.EProfileRaceTransmissionOptions
enum class EProfileRaceTransmissionOptions : uint8_t
{
PRTO_Auto                                   = 0,
PRTO_Manual                                 = 1,
PRTO_END                                    = 2
};

// Enum Engine.OnlineProfileSettings.EProfileRaceCameraLocationOptions
```

```
enum class EProfileRaceCameraLocationOptions : uint8_t
{
PRCLO_Behind                               = 0,
PRCLO_Front                                = 1,
PRCLO_Inside                               = 2,
PRCLO_END                                  = 3
};

// Enum Engine.OnlineProfileSettings.EProfileRaceBrakeControlOptions
enum class EProfileRaceBrakeControlOptions : uint8_t
{
PRBCO_Trigger                              = 0,
PRBCO_Button                               = 1,
PRBCO_END                                  = 2
};

// Enum Engine.OnlineProfileSettings.EProfileRaceAcceleratorControlOptions
enum class EProfileRaceAcceleratorControlOptions : uint8_t
{
PRACO_Trigger                              = 0,
PRACO_Button                               = 1,
PRACO_END                                  = 2
};

// Enum Engine.OnlineProfileSettings.EProfileYInversionOptions
enum class EProfileYInversionOptions : uint8_t
{
PYIO_Off                              = 0,
PYIO_On                               = 1,
PYIO_END                                = 2
};

// Enum Engine.OnlineProfileSettings.EProfileXInversionOptions
enum class EProfileXInversionOptions : uint8_t
{
PXIO_Off                              = 0,
PXIO_On                               = 1,
PXIO_END                                = 2
};

// Enum Engine.OnlineProfileSettings.EProfileOmniDirEvadeOptions
enum class EProfileOmniDirEvadeOptions : uint8_t
{
PODI_Off                              = 0,
PODI_On                               = 1,
PODI_END                                = 2
};

// Enum Engine.OnlineProfileSettings.EProfileControllerVibrationToggleOptions
enum class EProfileControllerVibrationToggleOptions : uint8_t
{
PCVTO_Off                                = 0,
PCVTO_IgnoreThis                               = 1,
PCVTO_IgnoreThis2                               = 2,
PCVTO_On                                   = 3,
PCVTO_END                                   = 4
};

// Enum Engine.OnlineProfileSettings.EProfileVoiceThruSpeakersOptions
enum class EProfileVoiceThruSpeakersOptions : uint8_t
{
PVTSO_Off                                  = 0,
PVTSO_On                                   = 1,
PVTSO_Both                                  = 2,
PVTSO_END                                   = 3
};

// Enum Engine.ParticleEmitter.EEmitterRenderMode
enum class EEmitterRenderMode : uint8_t
{
ERM_Normal                                  = 0,
ERM_Point                                  = 1,
ERM_Cross                                  = 2,
ERM_None                                   = 3,
ERM_END                                    = 4
};

// Enum Engine.ParticleEmitter.EParticleSubUVInterpMethod
enum class EParticleSubUVInterpMethod : uint8_t
{
PSUVIM_None                                 = 0,
PSUVIM_Linear                               = 1,
PSUVIM_Linear_Blend                            = 2,
PSUVIM_Random                                = 3,
PSUVIM_Random_Blend                             = 4,
PSUVIM_END                                  = 5
};

// Enum Engine.ParticleEmitter.EParticleBurstMethod
enum class EParticleBurstMethod : uint8_t
{
EPBM_Instant                                = 0,
EPBM_Interpolated                              = 1,
EPBM_END                                    = 2
};

// Enum Engine.ParticleModule.EModuleType
enum class EModuleType : uint8_t
{
EPMT_General                                 = 0,
EPMT_TypeData                                 = 1,
EPMT_Beam                                   = 2,
EPMT_Trail                                 = 3,
EPMT_Spawn                                   = 4,
EPMT_Required                                 = 5,
EPMT_Event                                  = 6,
EPMT_END                                    = 7
};
```

```
// Enum Engine.ParticleModule.EParticleSourceSelectionMethod
enum class EParticleSourceSelectionMethod : uint8_t
{
EPSSM_Random                          = 0,
EPSSM_Sequential                      = 1,
EPSSM_END                           = 2
};

// Enum Engine.ParticleModuleAttractorBoneSocket.EBoneSocketAttractorFalloffType
enum class EBoneSocketAttractorFalloffType : uint8_t
{
BSFOFF_Constant                       = 0,
BSFOFF_Linear                       = 1,
BSFOFF_Exponent                       = 2,
BSFOFF_END                          = 3
};

// Enum Engine.ParticleModuleAttractorBoneSocket.ELocationBoneSocketDestSelectionMethod
enum class ELocationBoneSocketDestSelectionMethod : uint8_t
{
BONESOCKETDESTSEL_Sequential                  = 0,
BONESOCKETDESTSEL_Random                      = 1,
BONESOCKETDESTSEL_RandomExhaustive                = 2,
BONESOCKETDESTSEL_BlendAll                   = 3,
BONESOCKETDESTSEL_END                       = 4
};

// Enum Engine.ParticleModuleAttractorBoneSocket.ELocationBoneSocketDestination
enum class ELocationBoneSocketDestination : uint8_t
{
BONESOCKETDEST_Bones                         = 0,
BONESOCKETDEST_Sockets                        = 1,
BONESOCKETDEST_END                          = 2
};

// Enum Engine.ParticleModuleAttractorParticle.EAttractorParticleSelectionMethod
enum class EAttractorParticleSelectionMethod : uint8_t
{
EAPSM_Random                          = 0,
EAPSM_Sequential                      = 1,
EAPSM_END                           = 2
};

// Enum Engine.ParticleModuleAttractorSkelVertSurface.EVertSurfaceAttractorFalloffType
enum class EVertSurfaceAttractorFalloffType : uint8_t
{
VSFOFF_Constant                       = 0,
VSFOFF_Linear                       = 1,
VSFOFF_Exponent                       = 2,
VSFOFF_END                          = 3
};

// Enum Engine.ParticleModuleAttractorSkelVertSurface.EAttractorSkelVertSurfaceDestination
```

```
enum class EAttractorSkelVertSurfaceDestination : uint8_t
{
VERTSURFACEDEST_Vert                     = 0,
VERTSURFACEDEST_Surface                  = 1,
VERTSURFACEDEST_END                      = 2
};

// Enum Engine.ParticleModuleBeamBase.Beam2SourceTargetMethod
enum class EBeam2SourceTargetMethod : uint8_t
{
PEB2STM_Default                          = 0,
PEB2STM_UserSet                          = 1,
PEB2STM_Emitter                          = 2,
PEB2STM_Particle                         = 3,
PEB2STM_Actor                            = 4,
PEB25TM_SkelMeshActor                    = 5,
Beam2SourceTargetMethod_END              = 6
};

// Enum Engine.ParticleModuleBeamBase.Beam2SourceTargetTangentMethod
enum class EBeam2SourceTargetTangentMethod : uint8_t
{
PEB2STTM_Direct                          = 0,
PEB2STTM_UserSet                         = 1,
PEB2STTM_Distribution                    = 2,
PEB2STTM_Emitter                         = 3,
PEB2STTM_END                             = 4
};

// Enum Engine.ParticleModuleBeamModifier.BeamModifierType
enum class EBeamModifierType : uint8_t
{
PEB2MT_Source                            = 0,
PEB2MT_Target                            = 1,
PEB2MT_END                               = 2
};

// Enum Engine.ParticleModuleCameraOffset.EParticleCameraOffsetUpdateMethod
enum class EParticleCameraOffsetUpdateMethod : uint8_t
{
EPCOUM_DirectSet                         = 0,
EPCOUM_Additive                          = 1,
EPCOUM_Scalar                            = 2,
EPCOUM_END                               = 3
};

// Enum Engine.ParticleModuleCollisionBase.EParticleCollisionComplete
enum class EParticleCollisionComplete : uint8_t
{
EPCC_Kill                                = 0,
EPCC_Freeze                              = 1,
EPCC_HaltCollisions                      = 2,
EPCC_FreezeTranslation                   = 3,
EPCC_FreezeRotation                      = 4,
EPCC_FreezeMovement                         = 5,
EPCC_END                                = 6
};

// Enum Engine.ParticleModuleCollision.ParticleAttractorActionType
enum class EParticleAttractorActionType : uint8_t
{
PAAT_None                                = 0,
PAAT_Destroy                              = 1,
PAAT_Freeze                               = 2,
PAAT_Event                               = 3,
PAAT_END                                 = 4
};

// Enum Engine.ParticleModuleLocationBoneSocket.ELocationBoneSocketSource
enum class ELocationBoneSocketSource : uint8_t
{
BONESOCKETSOURCE_Bones                      = 0,
BONESOCKETSOURCE_Sockets                     = 1,
BONESOCKETSOURCE_END                       = 2
};

// Enum Engine.ParticleModuleLocationBoneSocket.ELocationBoneSocketSelectionMethod
enum class ELocationBoneSocketSelectionMethod : uint8_t
{
BONESOCKETSEL_Sequential                     = 0,
BONESOCKETSEL_Random                        = 1,
BONESOCKETSEL_RandomExhaustive                  = 2,
BONESOCKETSEL_END                         = 3
};

// Enum Engine.ParticleModuleLocationEmitter.ELocationEmitterSelectionMethod
enum class ELocationEmitterSelectionMethod : uint8_t
{
ELESM_Random                              = 0,
ELESM_Sequential                            = 1,
ELESM_END                               = 2
};

// Enum Engine.ParticleModuleLocationPrimitiveCylinder.CylinderHeightAxis
enum class ECylinderHeightAxis : uint8_t
{
PMLPC_HEIGHTAXIS_X                         = 0,
PMLPC_HEIGHTAXIS_Y                         = 1,
PMLPC_HEIGHTAXIS_Z                         = 2,
PMLPC_HEIGHTAXIS_END                         = 3
};

// Enum Engine.ParticleModuleLocationSkelVertSurface.ELocationSkelVertSurfaceSource
enum class ELocationSkelVertSurfaceSource : uint8_t
{
VERTSURFACESOURCE_Vert                      = 0,
VERTSURFACESOURCE_Surface                     = 1,
VERTSURFACESOURCE_END                      = 2
```

```
};

// Enum Engine.ParticleModuleLocationStaticVertSurface.ELocationStaticVertSurfaceSource
enum class ELocationStaticVertSurfaceSource : uint8_t
{
VERTSTATICSURFACESOURCE_Vert                    = 0,
VERTSTATICSURFACESOURCE_Surface                 = 1,
VERTSTATICSURFACESOURCE_END                     = 2
};

// Enum Engine.ParticleModuleOrbit.EOrbitChainMode
enum class EOrbitChainMode : uint8_t
{
EOChainMode_Add                          = 0,
EOChainMode_Scale                        = 1,
EOChainMode_Link                         = 2,
EOChainMode_END                          = 3
};

// Enum Engine.ParticleModuleOrientationAxisLock.EParticleAxisLock
enum class EParticleAxisLock : uint8_t
{
EPAL_NONE                          = 0,
EPAL_X                             = 1,
EPAL_Y                             = 2,
EPAL_Z                             = 3,
EPAL_NEGATIVE_X                    = 4,
EPAL_NEGATIVE_Y                    = 5,
EPAL_NEGATIVE_Z                    = 6,
EPAL_ROTATE_X                      = 7,
EPAL_ROTATE_Y                      = 8,
EPAL_ROTATE_Z                      = 9,
EPAL_END                           = 10
};

// Enum Engine.ParticleModuleParameterDynamic.EEmitterDynamicParameterValue
enum class EEmitterDynamicParameterValue : uint8_t
{
EDPV_UserSet                       = 0,
EDPV_VelocityX                     = 1,
EDPV_VelocityY                     = 2,
EDPV_VelocityZ                     = 3,
EDPV_VelocityMag                   = 4,
EDPV_END                           = 5
};

// Enum Engine.ParticleModulePhysicsVolumes.EParticleLevelInfluenceType
enum class EParticleLevelInfluenceType : uint8_t
{
LIT_Never                          = 0,
LIT_OutsidePhysicsVolumes                    = 1,
LIT_Always                         = 2,
LIT_END                            = 3
};
```

```
};

// Enum Engine.ParticleSpriteEmitter.EParticleScreenAlignment
enum class EParticleScreenAlignment : uint8_t
{
PSA_Square                   = 0,
PSA_Rectangle                 = 1,
PSA_Velocity                 = 2,
PSA_TypeSpecific               = 3,
PSA_END                     = 4
};

// Enum Engine.ParticleModuleRequired.EEmitterNormalsMode
enum class EEmitterNormalsMode : uint8_t
{
ENM_CameraFacing                 = 0,
ENM_Spherical                 = 1,
ENM_Cylindrical               = 2,
ENM_END                      = 3
};

// Enum Engine.ParticleModuleRequired.EParticleSortMode
enum class EParticleSortMode : uint8_t
{
PSORTMODE_None                   = 0,
PSORTMODE_ViewProjDepth              = 1,
PSORTMODE_DistanceToView             = 2,
PSORTMODE_Age_OldestFirst            = 3,
PSORTMODE_Age_NewestFirst             = 4,
PSORTMODE_END                    = 5
};

// Enum Engine.ParticleModuleTrailSource.ETrail2SourceMethod
enum class ETrail2SourceMethod : uint8_t
{
PET2SRCM_Default                = 0,
PET2SRCM_Particle               = 1,
PET2SRCM_Actor                  = 2,
PET2SRCM_END                    = 3
};

// Enum Engine.ParticleModuleTrailSpawn.ETrail2SpawnMethod
enum class ETrail2SpawnMethod : uint8_t
{
PET2SM_Emitter                 = 0,
PET2SM_Velocity                = 1,
PET2SM_Distance                 = 2,
PET2SM_END                    = 3
};

// Enum Engine.ParticleModuleTrailTaper.ETrailTaperMethod
enum class ETrailTaperMethod : uint8_t
{
PETTM_None                    = 0,
PETTM_Full                   = 1,
PETTM_Partial                          = 2,
PETTM_END                              = 3
};

// Enum Engine.ParticleModuleTypeDataBeam.EBeamMethod
enum class EBeamMethod : uint8_t
{
PEBM_Distance                          = 0,
PEBM_EndPoints                         = 1,
PEBM_EndPoints_Interpolated            = 2,
PEBM_UserSet_EndPoints                 = 3,
PEBM_UserSet_EndPoints_Interpolated    = 4,
PEBM_END                               = 5
};

// Enum Engine.ParticleModuleTypeDataBeam.EBeamEndPointMethod
enum class EBeamEndPointMethod : uint8_t
{
PEBEPM_Calculated                      = 0,
PEBEPM_Distribution                    = 1,
PEBEPM_Distribution_Constant           = 2,
PEBEPM_END                             = 3
};

// Enum Engine.ParticleModuleTypeDataBeam2.EBeam2Method
enum class EBeam2Method : uint8_t
{
PEB2M_Distance                         = 0,
PEB2M_Target                           = 1,
PEB2M_Branch                           = 2,
PEB2M_END                              = 3
};

// Enum Engine.ParticleModuleTypeDataBeam2.EBeamTaperMethod
enum class EBeamTaperMethod : uint8_t
{
PEBTM_None                             = 0,
PEBTM_Full                             = 1,
PEBTM_Partial                          = 2,
PEBTM_END                              = 3
};

// Enum Engine.ParticleModuleTypeDataMesh.EMeshCameraFacingOptions
enum class EMeshCameraFacingOptions : uint8_t
{
XAxisFacing_NoUp                       = 0,
XAxisFacing_ZUp                        = 1,
XAxisFacing_NegativeZUp                = 2,
XAxisFacing_YUp                        = 3,
XAxisFacing_NegativeYUp                = 4,
LockedAxis_ZAxisFacing                 = 5,
LockedAxis_NegativeZAxisFacing         = 6,
LockedAxis_YAxisFacing                 = 7,
LockedAxis_NegativeYAxisFacing         = 8,
VelocityAligned_ZAxisFacing             = 9,
VelocityAligned_NegativeZAxisFacing         = 10,
VelocityAligned_YAxisFacing             = 11,
VelocityAligned_NegativeYAxisFacing         = 12,
EMeshCameraFacingOptions_END                 = 13
};

// Enum Engine.ParticleModuleTypeDataMesh.EMeshCameraFacingUpAxis
enum class EMeshCameraFacingUpAxis : uint8_t
{
CameraFacing_NoneUP                    = 0,
CameraFacing_ZUp                     = 1,
CameraFacing_NegativeZUp                 = 2,
CameraFacing_YUp                     = 3,
CameraFacing_NegativeYUp                 = 4,
CameraFacing_END                     = 5
};

// Enum Engine.ParticleModuleTypeDataMesh.EMeshScreenAlignment
enum class EMeshScreenAlignment : uint8_t
{
PSMA_MeshFaceCameraWithRoll                 = 0,
PSMA_MeshFaceCameraWithSpin                  = 1,
PSMA_MeshFaceCameraWithLockedAxis                = 2,
PSMA_END                          = 3
};

// Enum Engine.ParticleModuleTypeDataMeshPhysX.EPhysXMeshRotationMethod
enum class EPhysXMeshRotationMethod : uint8_t
{
PMRM_Disabled                       = 0,
PMRM_Spherical                      = 1,
PMRM_Box                          = 2,
PMRM_LongBox                        = 3,
PMRM_FlatBox                       = 4,
PMRM_Velocity                      = 5,
PMRM_END                          = 6
};

// Enum Engine.ParticleModuleTypeDataRibbon.ETrailsRenderAxisOption
enum class ETrailsRenderAxisOption : uint8_t
{
Trails_CameraUp                    = 0,
Trails_SourceUp                   = 1,
Trails_WorldUp                    = 2,
Trails_SourceSideIsUp                = 3,
Trails_END                      = 4
};

// Enum Engine.ParticleSystem.EParticleSystemOcclusionBoundsMethod
enum class EParticleSystemOcclusionBoundsMethod : uint8_t
{
EPSOBM_None                         = 0,
EPSOBM_ParticleBounds                   = 1,
EPSOBM_CustomBounds                    = 2,
EPSOBM_END                             = 3
};

// Enum Engine.ParticleSystem.EParticleSystemUpdateMode
enum class EParticleSystemUpdateMode : uint8_t
{
EPSUM_RealTime                         = 0,
EPSUM_FixedTime                        = 1,
EPSUM_END                              = 2
};

// Enum Engine.ProcBuildingRuleset.EProcBuildingAxis
enum class EProcBuildingAxis : uint8_t
{
EPBAxis_X                              = 0,
EPBAxis_Z                              = 1,
EPBAxis_END                            = 2
};

// Enum Engine.ProcBuilding.EScopeEdge
enum class EScopeEdge : uint8_t
{
EPSA_Top                               = 0,
EPSA_Bottom                            = 1,
EPSA_Left                              = 2,
EPSA_Right                             = 3,
EPSA_None                              = 4,
EPSA_END                               = 5
};

// Enum Engine.ProcBuilding.EBuildingStatsBrowserColumns
enum class EBuildingStatsBrowserColumns : uint8_t
{
BSBC_Name                              = 0,
BSBC_Ruleset                           = 1,
BSBC_NumStaticMeshComps                = 2,
BSBC_NumInstancedStaticMeshComps       = 3,
BSBC_NumInstancedTris                  = 4,
BSBC_LightmapMemBytes                  = 5,
BSBC_ShadowmapMemBytes                 = 6,
BSBC_LODDiffuseMemBytes                = 7,
BSBC_LODLightingMemBytes               = 8,
BSBC_END                               = 9
};

// Enum Engine.ProcBuilding.EPBCornerType
enum class EPBCornerType : uint8_t
{
EPBC_Default                           = 0,
EPBC_Chamfer                           = 1,
EPBC_Round                             = 2,
EPBC_END                               = 3
};
```

```
// Enum Engine.PBRuleNodeEdgeAngle.EProcBuildingEdge
enum class EProcBuildingEdge : uint8_t
{
EPBE_Top                                = 0,
EPBE_Bottom                               = 1,
EPBE_Left                               = 2,
EPBE_Right                               = 3,
EPBE_END                                 = 4
};

// Enum Engine.PhysicalMaterial.EPhysEffectType
enum class EPhysEffectType : uint8_t
{
EPMET_Impact                               = 0,
EPMET_Slide                              = 1,
EPMET_END                                 = 2
};

// Enum Engine.PhysXParticleSystem.ESimulationMethod
enum class ESimulationMethod : uint8_t
{
ESM_SPH                                  = 0,
ESM_NO_PARTICLE_INTERACTION                    = 1,
ESM_MIXED_MODE                                = 2,
ESM_END                                  = 3
};

// Enum Engine.PhysXParticleSystem.EPacketSizeMultiplier
enum class EPacketSizeMultiplier : uint8_t
{
EPSM                                 = 0,
EPSM01                                 = 1,
EPSM02                                 = 2,
EPSM03                                 = 3,
EPSM04                                 = 4,
EPSM05                                 = 5,
EPSM_END                                 = 6
};

// Enum Engine.PitchTekSettings.EPitchTekTargetSize
enum class EPitchTekTargetSize : uint8_t
{
PTTS                                 = 0,
PTTS01                                 = 1,
PTTS02                                 = 2,
PTTS_END                                 = 3
};

// Enum Engine.PlatformAccountSettings.ECrossPlatformChatState
enum class ECrossPlatformChatState : uint8_t
{
PCCS_Everybody                             = 0,
PCCS_InGameFriends                           = 1,
PCCS_Block                              = 2,
PCCS_END                                = 3
};

// Enum Engine.PlatformBlockListStatus.EBlockListDownloadStatus
enum class EBlockListDownloadStatus : uint8_t
{
EB_Pending                              = 0,
EB_Failure                            = 1,
EB_Success                              = 2,
EB_END                                = 3
};

// Enum Engine.SceneCaptureComponent.ESceneCapturePostMethod
enum class ESceneCapturePostMethod : uint8_t
{
SceneCapPost_None                       = 0,
SceneCapPost_Desat                      = 1,
SceneCapPost_Seamless                       = 2,
SceneCapPost_END                        = 3
};

// Enum Engine.SceneCaptureComponent.ESceneCaptureViewMode
enum class ESceneCaptureViewMode : uint8_t
{
SceneCapView_Lit                        = 0,
SceneCapView_Unlit                        = 1,
SceneCapView_LitNoShadows                       = 2,
SceneCapView_Wire                         = 3,
SceneCapView_END                          = 4
};

// Enum Engine.RB_BodySetup.ESleepFamily
enum class ESleepFamily : uint8_t
{
SF_Normal                               = 0,
SF_Sensitive                            = 1,
SF_END                                = 2
};

// Enum Engine.RB_RadialForceActor.ERadialForceType
enum class ERadialForceType : uint8_t
{
RFT_Force                               = 0,
RFT_Impulse                               = 1,
RFT_END                                 = 2
};

// Enum Engine.Route.ERouteDirection
enum class ERouteDirection : uint8_t
{
ERD_Forward                               = 0,
ERD_Reverse                               = 1,
ERD_END                                 = 2
```

```
};

// Enum Engine.Route.ERouteFillAction
enum class ERouteFillAction : uint8_t
{
RFA_Overwrite                          = 0,
RFA_Add                                = 1,
RFA_Remove                             = 2,
RFA_Clear                              = 3,
RFA_END                                = 4
};

// Enum Engine.Route.ERouteType
enum class ERouteType : uint8_t
{
ERT_Linear                             = 0,
ERT_Loop                               = 1,
ERT_Circle                             = 2,
ERT_END                                = 3
};

// Enum Engine.SeqAct_ActorFactory.EPointSelection
enum class EPointSelection : uint8_t
{
PS_Normal                              = 0,
PS_Random                              = 1,
PS_Reverse                             = 2,
PS_END                                 = 3
};

// Enum Engine.SeqAct_SetMesh.EMeshType
enum class EMeshType : uint8_t
{
MeshType_StaticMesh                    = 0,
MeshType_SkeletalMesh                  = 1,
MeshType_END                           = 2
};

// Enum Engine.WorldAttractor.EWorldAttractorFalloffType
enum class EWorldAttractorFalloffType : uint8_t
{
FOFF_Constant                          = 0,
FOFF_Linear                            = 1,
FOFF_Exponent                          = 2,
FOFF_END                               = 3
};

// Enum Engine.SeqEvent_ParticleEvent.EParticleEventOutputType
enum class EParticleEventOutputType : uint8_t
{
ePARTICLEOUT_Spawn                     = 0,
ePARTICLEOUT_Death                     = 1,
ePARTICLEOUT_Collision                 = 2,
ePARTICLEOUT_AttractorCollision        = 3,
ePARTICLEOUT_Kismet                    = 4,
ePARTICLEOUT_END                       = 5
};

// Enum Engine.SkelControlBase.EBoneControlSpace
enum class EBoneControlSpace : uint8_t
{
BCS_WorldSpace                         = 0,
BCS_ActorSpace                         = 1,
BCS_ComponentSpace                     = 2,
BCS_ParentBoneSpace                    = 3,
BCS_BoneSpace                          = 4,
BCS_OtherBoneSpace                     = 5,
BCS_BaseMeshSpace                      = 6,
BCS_END                                = 7
};

// Enum Engine.SkelControlSpline.ESplineControlRotMode
enum class ESplineControlRotMode : uint8_t
{
SCR_NoChange                           = 0,
SCR_AlongSpline                        = 1,
SCR_Interpolate                        = 2,
SCR_END                                = 3
};

// Enum Engine.SkeletalMesh.SoftBodyBoneType
enum class ESoftBodyBoneType : uint8_t
{
SOFTBODYBONE_Fixed                          = 0,
SOFTBODYBONE_BreakableAttachment            = 1,
SOFTBODYBONE_TwoWayAttachment               = 2,
SOFTBODYBONE_END                            = 3
};

// Enum Engine.SkeletalMesh.ClothBoneType
enum class EClothBoneType : uint8_t
{
CLOTHBONE_Fixed                             = 0,
CLOTHBONE_BreakableAttachment               = 1,
CLOTHBONE_TearLine                          = 2,
CLOTHBONE_END                               = 3
};

// Enum Engine.SkeletalMesh.SkeletalMeshOptimizationNormalMode
enum class ESkeletalMeshOptimizationNormalMode : uint8_t
{
SMONM_Recalculate                           = 0,
SMONM_RecalculateSoft                       = 1,
SMONM_RecalculateHard                       = 2,
SMONM_END                                   = 3
};

// Enum Engine.SkeletalMesh.SkeletalMeshOptimizationImportance
```

```
enum class ESkeletalMeshOptimizationImportance : uint8_t
{
SMOI_Off                              = 0,
SMOI_Lowest                           = 1,
SMOI_Low                              = 2,
SMOI_Normal                           = 3,
SMOI_High                             = 4,
SMOI_Highest                          = 5,
SMOI_END                              = 6
};

// Enum Engine.SkeletalMesh.SkeletalMeshOptimizationType
enum class ESkeletalMeshOptimizationType : uint8_t
{
SMOT_NumOfTriangles                     = 0,
SMOT_MaxDeviation                       = 1,
SMOT_END                                = 2
};

// Enum Engine.SkeletalMesh.TriangleSortOption
enum class ETriangleSortOption : uint8_t
{
TRISORT_None                            = 0,
TRISORT_CenterRadialDistance                    = 1,
TRISORT_Random                            = 2,
TRISORT_MergeContiguous                       = 3,
TRISORT_Custom                            = 4,
TRISORT_CustomLeftRight                      = 5,
TRISORT_END                              = 6
};

// Enum Engine.SkeletalMesh.BoneBreakOption
enum class EBoneBreakOption : uint8_t
{
BONEBREAK_SoftPreferred                     = 0,
BONEBREAK_AutoDetect                        = 1,
BONEBREAK_RigidPreferred                    = 2,
BONEBREAK_END                             = 3
};

// Enum Engine.SkeletalMesh.TriangleSortAxis
enum class ETriangleSortAxis : uint8_t
{
TSA_X_Axis                              = 0,
TSA_Y_Axis                              = 1,
TSA_Z_Axis                              = 2,
TSA_END                                = 3
};

// Enum Engine.SkeletalMesh.ClothMovementScaleGen
enum class EClothMovementScaleGen : uint8_t
{
ECMDM_DistToFixedVert                      = 0,
ECMDM_VertexBoneWeight                       = 1,
ECMDM_Empty                              = 2,
ECMDM_END                                = 3
};

// Enum Engine.SoundNodeWave.EDecompressionType
enum class EDecompressionType : uint8_t
{
DTYPE_Setup                              = 0,
DTYPE_Invalid                            = 1,
DTYPE_Preview                            = 2,
DTYPE_Native                             = 3,
DTYPE_RealTime                           = 4,
DTYPE_Procedural                         = 5,
DTYPE_Xenon                              = 6,
DTYPE_Dingo                              = 7,
DTYPE_END                                = 8
};

// Enum Engine.SpeedTreeComponent.ESpeedTreeMeshType
enum class ESpeedTreeMeshType : uint8_t
{
STMT_MinMinusOne                         = 0,
STMT_Branches1                           = 1,
STMT_Branches2                           = 2,
STMT_Fronds                              = 3,
STMT_LeafCards                           = 4,
STMT_LeafMeshes                          = 5,
STMT_Billboards                          = 6,
STMT_Max                                 = 7
};

// Enum Engine.TerrainMaterial.ETerrainMappingType
enum class ETerrainMappingType : uint8_t
{
TMT_Auto                                 = 0,
TMT_XY                                   = 1,
TMT_XZ                                   = 2,
TMT_YZ                                   = 3,
TMT_END                                  = 4
};

// Enum Engine.TextureFlipBook.TextureFlipBookMethod
enum class ETextureFlipBookMethod : uint8_t
{
TFBM_UL_ROW                              = 0,
TFBM_UL_COL                              = 1,
TFBM_UR_ROW                              = 2,
TFBM_UR_COL                              = 3,
TFBM_LL_ROW                              = 4,
TFBM_LL_COL                              = 5,
TFBM_LR_ROW                              = 6,
TFBM_LR_COL                              = 7,
TFBM_RANDOM                              = 8,
TFBM_END                                 = 9
```

```
};

// Enum Engine.TextureMovie.EMovieStreamSource
enum class EMovieStreamSource : uint8_t
{
MovieStream_File                    = 0,
MovieStream_Memory                    = 1,
MovieStream_END                     = 2
};

// Enum Engine.TwitterIntegrationBase.ETwitterRequestMethod
enum class ETwitterRequestMethod : uint8_t
{
TRM_Get                             = 0,
TRM_Post                            = 1,
TRM_Delete                           = 2,
TRM_END                             = 3
};

// Enum Engine.TwitterIntegrationBase.ETwitterIntegrationDelegate
enum class ETwitterIntegrationDelegate : uint8_t
{
TID_AuthorizeComplete                  = 0,
TID_TweetUIComplete                    = 1,
TID_RequestComplete                    = 2,
TID_END                             = 3
};

// Enum Engine.UberPostProcessEffect.EPostProcessAAType
enum class EPostProcessAAType : uint8_t
{
PostProcessAA_Off                     = 0,
PostProcessAA_FXAA0                     = 1,
PostProcessAA_FXAA1                     = 2,
PostProcessAA_FXAA2                     = 3,
PostProcessAA_FXAA3                     = 4,
PostProcessAA_FXAA4                     = 5,
PostProcessAA_FXAA5                     = 6,
PostProcessAA_MLAA                     = 7,
PostProcessAA_SMAA                     = 8,
PostProcessAA_END                     = 9
};

// Enum Engine.UberPostProcessEffect.ETonemapperType
enum class ETonemapperType : uint8_t
{
Tonemapper_Off                      = 0,
Tonemapper_Filmic                      = 1,
Tonemapper_Customizable                    = 2,
Tonemapper_END                       = 3
};

// Enum Engine.UIDataProvider_MenuItem.EMenuOptionType
enum class EMenuOptionType : uint8_t
```

```
{
MENUOT_ComboReadOnly                    = 0,
MENUOT_ComboNumeric                     = 1,
MENUOT_CheckBox                      = 2,
MENUOT_Slider                      = 3,
MENUOT_Spinner                      = 4,
MENUOT_EditBox                      = 5,
MENUOT_CollectionCheckBox                 = 6,
MENUOT_CollapsingList                   = 7,
MENUOT_END                       = 8
};

// Enum Engine.UIDataStore_OnlineStats.EStatsFetchType
enum class EStatsFetchType : uint8_t
{
SFT_Player                      = 0,
SFT_CenteredOnPlayer                   = 1,
SFT_Friends                      = 2,
SFT_TopRankings                     = 3,
SFT_END                       = 4
};


/*
#
==========================================================================
=========== #
# Classes
#
==========================================================================
=========== #
*/

// Class Engine.ScriptGroup_ORS
// 0x0008 (0x0060 - 0x0068)
class UScriptGroup_ORS : public UObject
{
public:
class UObject*                    GroupOwner;                    // 0x0060 (0x0008)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ScriptGroup_ORS");
}

return uClassPointer;
};
```

```
void ClearTimerClass(class UClass* EventClass);
void ClearTimer(class UObject* Event);
void SetGameTimer(class UObject* Event, float Delay, struct FTimerOptions Options);
void SetTimer(class UObject* Event, float Delay, struct FTimerOptions Options);
void Broadcast(class UObject* Event);
void CreateObjects(class UObject* ObjOuter, TArray<class UClass*>& ObjectClasses);
class UObject* CreateObject(class UClass* ObjectClass, class UObject* ObjOuter);
class UObject* GetOrCreateObject(class UClass* ObjectClass, class UObject* ObjOuter);
class UObject* GetObjectW(class UClass* ObjectClass);
class UObject* DestroyClass(class UClass* ObjectClass);
void RemoveAllClasses(class UClass* ObjectClass);
class UObject* RemoveClass(class UClass* ObjectClass);
void DestroyObject(class UObject* Object);
void RemoveObject(class UObject* Object);
void AddObject(class UObject* Object);
void SetGroupParent(class UObject* ParentGroup);
};

// Class Engine.Actor
// 0x0208 (0x0060 - 0x0268)
class AActor : public UObject
{
public:
TArray<class UParticleSystemComponent*>          ActorDependantPSCs;                    //
0x0060 (0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UActorComponent*>                   Components;                            // 0x0070
(0x0010) [0x000000000448000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
TArray<class UActorComponent*>                   AllComponents;                         // 0x0080
(0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
struct FVector                           Location;                              // 0x0090 (0x000C)
[0x0000000000000023] (CPF_Edit | CPF_Const | CPF_Net)
struct FRotator                          Rotation;                              // 0x009C (0x000C)
[0x0000000000000023] (CPF_Edit | CPF_Const | CPF_Net)
float                                    DrawScale;                             // 0x00A8 (0x0004)
[0x0000000300000023] (CPF_Edit | CPF_Const | CPF_Net)
struct FVector                           DrawScale3D;                           // 0x00AC (0x000C)
[0x0000000200000003] (CPF_Edit | CPF_Const)
struct FVector                           PrePivot;                              // 0x00B8 (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FColor                            EditorIconColor;                       // 0x00C4 (0x0004)
[0x0000000800000001] (CPF_Edit)
struct FRenderCommandFence                       DetachFence;                           // 0x00C8
(0x0004) [0x0000000000001002] (CPF_Const | CPF_Native)
float                                    CustomTimeDilation;                    // 0x00CC (0x0004)
[0x0000000000000000]
uint8_t                                  Physics;                               // 0x00D0 (0x0001)
[0x0000000000000023] (CPF_Edit | CPF_Const | CPF_Net)
uint8_t                                  RemoteRole;                            // 0x00D1 (0x0001)
[0x0000000000000020] (CPF_Net)
uint8_t                                  Role;                                  // 0x00D2 (0x0001)
[0x0000000000000020] (CPF_Net)
```

```
uint8_t                              CollisionType;                          // 0x00D3 (0x0001)
[0x0000000000002003] (CPF_Edit | CPF_Const | CPF_Transient)
uint8_t                              ReplicatedCollisionType;                  // 0x00D4 (0x0001)
[0x0000000000002020] (CPF_Net | CPF_Transient)
uint8_t                              TickGroup;                              // 0x00D5 (0x0001)
[0x0000000000000002] (CPF_Const)
class AActor*                          Owner;                                 // 0x00D8 (0x0008)
[0x0000000000000022] (CPF_Const | CPF_Net)
class AActor*                          Base;                                  // 0x00E0 (0x0008)
[0x0000000000000023] (CPF_Edit | CPF_Const | CPF_Net)
TArray<struct FTimerData>                    Timers;                              // 0x00E8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
unsigned long                          bStatic : 1;                           // 0x00F8 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                          bHidden : 1;                           // 0x00F8 (0x0004)
[0x0000000000000023] [0x00000002] (CPF_Edit | CPF_Const | CPF_Net)
unsigned long                          bHiddenSelf : 1;                         // 0x00F8 (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                          bNoDelete : 1;                          // 0x00F8 (0x0004)
[0x0000000000000002] [0x00000008] (CPF_Const)
unsigned long                          bDeleteMe : 1;                          // 0x00F8 (0x0004)
[0x0000000000000002] [0x00000010] (CPF_Const)
unsigned long                          bTicked : 1;                            // 0x00F8 (0x0004)
[0x0000000000002002] [0x00000020] (CPF_Const | CPF_Transient)
unsigned long                          bOnlyOwnerSee : 1;                        // 0x00F8 (0x0004)
[0x0000000000000002] [0x00000040] (CPF_Const)
unsigned long                          bTickIsDisabled : 1;                      // 0x00F8 (0x0004)
[0x0000000000000002] [0x00000080] (CPF_Const)
unsigned long                          bWorldGeometry : 1;                       // 0x00F8 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                          bIgnoreRigidBodyPawns : 1;                  // 0x00F8 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                          bOrientOnSlope : 1;                       // 0x00F8 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                          bIgnoreEncroachers : 1;                    // 0x00F8 (0x0004)
[0x0000000000000002] [0x00000800] (CPF_Const)
unsigned long                          bPushedByEncroachers : 1;                  // 0x00F8 (0x0004)
[0x0000000000000000] [0x00001000]
unsigned long                          bDestroyedByInterpActor : 1;               // 0x00F8 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                          bRouteBeginPlayEvenIfStatic : 1;            // 0x00F8 (0x0004)
[0x0000000000000002] [0x00004000] (CPF_Const)
unsigned long                          bIsMoving : 1;                          // 0x00F8 (0x0004)
[0x0000000000000002] [0x00008000] (CPF_Const)
unsigned long                          bAlwaysEncroachCheck : 1;                  // 0x00F8 (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                          bHasAlternateTargetLocation : 1;            // 0x00F8 (0x0004)
[0x0000000000000000] [0x00020000]
unsigned long                          bCanStepUpOn : 1;                        // 0x00F8 (0x0004)
[0x0000000000000001] [0x00040000] (CPF_Edit)
unsigned long                          bNetTemporary : 1;                       // 0x00F8 (0x0004)
[0x0000000000000002] [0x00080000] (CPF_Const)
unsigned long                          bOnlyRelevantToOwner : 1;                 // 0x00F8 (0x0004)
[0x0000000000000002] [0x00100000] (CPF_Const)
```

```
unsigned long                    bNetDirty : 1;                    // 0x00F8 (0x0004)
[0x0000000000002000] [0x00200000] (CPF_Transient)
unsigned long                    bAlwaysRelevant : 1;                    // 0x00F8 (0x0004)
[0x0000000000000000] [0x00400000]
unsigned long                    bReplicateInstigator : 1;                    // 0x00F8 (0x0004)
[0x0000000000000000] [0x00800000]
unsigned long                    bReplicateMovement : 1;                    // 0x00F8 (0x0004)
[0x0000000000000000] [0x01000000]
unsigned long                    bSkipActorPropertyReplication : 1;            // 0x00F8 (0x0004)
[0x0000000000000000] [0x02000000]
unsigned long                    bUpdateSimulatedPosition : 1;                // 0x00F8 (0x0004)
[0x0000000000000000] [0x04000000]
unsigned long                    bTearOff : 1;                    // 0x00F8 (0x0004)
[0x0000000000000020] [0x08000000] (CPF_Net)
unsigned long                    bOnlyDirtyReplication : 1;                   // 0x00F8 (0x0004)
[0x0000000000000000] [0x10000000]
unsigned long                    bAllowFluidSurfaceInteraction : 1;            // 0x00F8 (0x0004)
[0x0000000000000001] [0x20000000] (CPF_Edit)
unsigned long                    bDemoRecording : 1;                    // 0x00F8 (0x0004)
[0x0000000000002000] [0x40000000] (CPF_Transient)
unsigned long                    bDemoOwner : 1;                    // 0x00F8 (0x0004)
[0x0000000000000000] [0x80000000]
unsigned long                    bForceDemoRelevant : 1;                    // 0x00FC (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                    bNetInitialRotation : 1;                    // 0x00FC (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                    bReplicateRigidBodyLocation : 1;             // 0x00FC (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                    bKillDuringLevelTransition : 1;              // 0x00FC (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                    bExchangedRoles : 1;                    // 0x00FC (0x0004)
[0x0000000000000002] [0x00000010] (CPF_Const)
unsigned long                    bConsiderAllStaticMeshComponentsForStreaming : 1;//
0x00FC (0x0004) [0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                    bDebug : 1;                    // 0x00FC (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                    bPostRenderIfNotVisible : 1;                 // 0x00FC (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                    bForceNetUpdate : 1;                    // 0x00FC (0x0004)
[0x0000000000002000] [0x00000100] (CPF_Transient)
unsigned long                    bForcePacketUpdate : 1;                    // 0x00FC (0x0004)
[0x0000000000002002] [0x00000200] (CPF_Const | CPF_Transient)
unsigned long                    bPendingNetUpdate : 1;                    // 0x00FC (0x0004)
[0x0000000000002002] [0x00000400] (CPF_Const | CPF_Transient)
unsigned long                    bHardAttach : 1;                    // 0x00FC (0x0004)
[0x0000000000000023] [0x00000800] (CPF_Edit | CPF_Const | CPF_Net)
unsigned long                    bIgnoreBaseRotation : 1;                    // 0x00FC (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                    bShadowParented : 1;                    // 0x00FC (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                    bSkipAttachedMoves : 1;                    // 0x00FC (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                    bCanBeAdheredTo : 1;                    // 0x00FC (0x0004)
[0x0000000000000000] [0x00008000]
```

```
unsigned long                          bCanBeFrictionedTo : 1;                    // 0x00FC (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                          bGameRelevant : 1;                        // 0x00FC (0x0004)
[0x0000000000000000] [0x00020000]
unsigned long                          bMovable : 1;                             // 0x00FC (0x0004)
[0x0000000000000002] [0x00040000] (CPF_Const)
unsigned long                          bShouldBaseAtStartup : 1;                 // 0x00FC (0x0004)
[0x0000000000000000] [0x00080000]
unsigned long                          bPendingDelete : 1;                       // 0x00FC (0x0004)
[0x0000000000000000] [0x00100000]
unsigned long                          bCanTeleport : 1;                         // 0x00FC (0x0004)
[0x0000000000000000] [0x00200000]
unsigned long                          bAlwaysTick : 1;                          // 0x00FC (0x0004)
[0x0000000000000000] [0x00400000]
unsigned long                          bBlocksNavigation : 1;                    // 0x00FC (0x0004)
[0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                          BlockRigidBody : 1;                       // 0x00FC (0x0004)
[0x0000000000002003] [0x01000000] (CPF_Edit | CPF_Const | CPF_Transient)
unsigned long                          bCollideWhenPlacing : 1;                  // 0x00FC (0x0004)
[0x0000000000000000] [0x02000000]
unsigned long                          bCollideActors : 1;                       // 0x00FC (0x0004)
[0x0000000000000022] [0x04000000] (CPF_Const | CPF_Net)
unsigned long                          bCollideWorld : 1;                        // 0x00FC (0x0004)
[0x0000000000000020] [0x08000000] (CPF_Net)
unsigned long                          bCollideComplex : 1;                      // 0x00FC (0x0004)
[0x0000000000000001] [0x10000000] (CPF_Edit)
unsigned long                          bBlockActors : 1;                         // 0x00FC (0x0004)
[0x0000000000000020] [0x20000000] (CPF_Net)
unsigned long                          bBlocksTeleport : 1;                      // 0x00FC (0x0004)
[0x0000000000000000] [0x40000000]
unsigned long                          bMoveIgnoresDestruction : 1;              // 0x00FC (0x0004)
[0x0000000000000000] [0x80000000]
unsigned long                          bProjectileMoveSingleBlocking : 1;        // 0x0100 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                          bNoEncroachCheck : 1;                     // 0x0100 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bCollideAsEncroacher : 1;                 // 0x0100 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                          bPhysRigidBodyOutOfWorldCheck : 1;        // 0x0100
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bComponentOutsideWorld : 1;               // 0x0100 (0x0004)
[0x0000000000002002] [0x00000010] (CPF_Const | CPF_Transient)
unsigned long                          bForceOctreeSNFilter : 1;                 // 0x0100 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                          bForceOctreeMNFilter : 1;                 // 0x0100 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                          bRigidBodyWasAwake : 1;                   // 0x0100 (0x0004)
[0x0000000000002002] [0x00000080] (CPF_Const | CPF_Transient)
unsigned long                          bCallRigidBodyWakeEvents : 1;             // 0x0100 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                          bBounce : 1;                              // 0x0100 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                          bJustTeleported : 1;                      // 0x0100 (0x0004)
[0x0000000000000002] [0x00000400] (CPF_Const)
```

```
unsigned long                              bEnableMobileTouch : 1;                    // 0x0100 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                              bNetInitial : 1;                        // 0x0100 (0x0004)
[0x0000000000000002] [0x00001000] (CPF_Const)
unsigned long                              bNetOwner : 1;                          // 0x0100 (0x0004)
[0x0000000000000022] [0x00002000] (CPF_Const | CPF_Net)
unsigned long                              bHiddenEd : 1;                          // 0x0100 (0x0004)
[0x0000000000000002] [0x00004000] (CPF_Const)
unsigned long                              bEditable : 1;                        // 0x0100 (0x0004)
[0x0000000000000002] [0x00008000] (CPF_Const)
unsigned long                              bHiddenEdGroup : 1;                    // 0x0100 (0x0004)
[0x0000000020000002] [0x00010000] (CPF_Const | CPF_Deprecated)
unsigned long                              bHiddenEdLayer : 1;                    // 0x0100 (0x0004)
[0x0000000000000002] [0x00020000] (CPF_Const)
unsigned long                              bHiddenEdCustom : 1;                    // 0x0100 (0x0004)
[0x0000000000000002] [0x00040000] (CPF_Const)
unsigned long                              bHiddenEdTemporary : 1;                  // 0x0100 (0x0004)
[0x0000000800002000] [0x00080000] (CPF_Transient)
unsigned long                              bHiddenEdLevel : 1;                    // 0x0100 (0x0004)
[0x0000000800002000] [0x00100000] (CPF_Transient)
unsigned long                              bHiddenEdScene : 1;                    // 0x0100 (0x0004)
[0x0000000800002000] [0x00200000] (CPF_Transient)
unsigned long                              bHiddenEdNoPhysics : 1;                  // 0x0100 (0x0004)
[0x0000000800002000] [0x00400000] (CPF_Transient)
unsigned long                              bEdShouldSnap : 1;                     // 0x0100 (0x0004)
[0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                              bTempEditor : 1;                      // 0x0100 (0x0004)
[0x0000000000002002] [0x01000000] (CPF_Const | CPF_Transient)
unsigned long                              bPathColliding : 1;                    // 0x0100 (0x0004)
[0x0000000000000001] [0x02000000] (CPF_Edit)
unsigned long                              bPathTemp : 1;                         // 0x0100 (0x0004)
[0x0000000000002000] [0x04000000] (CPF_Transient)
unsigned long                              bScriptInitialized : 1;                // 0x0100 (0x0004)
[0x0000000000000000] [0x08000000]
unsigned long                              bLockLocation : 1;                      // 0x0100 (0x0004)
[0x0000000000000001] [0x10000000] (CPF_Edit)
unsigned long                              bForceAllowKismetModification : 1;           // 0x0100
(0x0004) [0x0000000000000002] [0x20000000] (CPF_Const)
unsigned long                              bDedicatedServerRelevant : 1;              // 0x0100 (0x0004)
[0x0000000000000001] [0x40000000] (CPF_Edit)
unsigned long                              bLockedFromEditorDeletion : 1;              // 0x0100 (0x0004)
[0x0000000800200000] [0x80000000]
unsigned long                              bComponentsDirty : 1;                  // 0x0104 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                              bUpdateComponentsIfEmpty : 1;              // 0x0104
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                              bDebugEffectIsRelevant : 1;                // 0x0104 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
int32_t                               SkelMeshCompTickTag;                     // 0x0108 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                               NetTag;                             // 0x010C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                NetUpdateTime;                           // 0x0110 (0x0004)
[0x0000000000000002] (CPF_Const)
```

```
float                          NetUpdateFrequency;                  // 0x0114 (0x0004)
[0x0000000000000000]
float                          NetPriority;                         // 0x0118 (0x0004)
[0x0000000000000000]
float                          LastNetUpdateTime;                   // 0x011C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          LastForcePacketUpdateTime;               // 0x0120 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          TimeSinceLastTick;                   // 0x0124 (0x0004)
[0x0000000000000000]
class APawn*                        Instigator;                      // 0x0128 (0x0008)
[0x0000000100000020] (CPF_Net)
class AWorldInfo*                     WorldInfo;                      // 0x0130 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          LifeSpan;                            // 0x0138 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          CreationTime;                        // 0x013C (0x0004)
[0x0000000000000002] (CPF_Const)
float                          LastRenderTime;                      // 0x0140 (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FName                        Tag;                             // 0x0144 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                        InitialState;                    // 0x014C (0x0008)
[0x0000000000000000]
struct FName                        Layer;                           // 0x0154 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                        Group;                           // 0x015C (0x0008)
[0x0000000020000000]  CPF_Deprecated)
uint64_t                          HiddenEditorViews;                 // 0x0168 (0x0008)
[0x0000000000002000] (CPF_Transient)
TArray<class AActor*>                   Touching;                      // 0x0170 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class AActor*>                   Children;                      // 0x0180 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
float                          LatentFloat;                         // 0x0190 (0x0004)
[0x0000000000000002] (CPF_Const)
class UAnimNodeSequence*                  LatentSeqNode;                       // 0x0198
(0x0008) [0x0000000000000002] (CPF_Const)
class APhysicsVolume*                   PhysicsVolume;                    // 0x01A0 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FVector                      Velocity;                        // 0x01A8 (0x000C)
[0x0000000000000020] (CPF_Net)
struct FVector                      Acceleration;                      // 0x01B4 (0x000C)
[0x0000000000000000]
struct FVector                      AngularVelocity;                   // 0x01C0 (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class USkeletalMeshComponent*                BaseSkelComponent;                   // 0x01D0
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
struct FName                        BaseBoneName;                      // 0x01D8 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<class AActor*>                   Attached;                      // 0x01E0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FVector                      RelativeLocation;                  // 0x01F0 (0x000C)
```

```
[0x0000000000000022] (CPF_Const | CPF_Net)
struct FRotator                    RelativeRotation;                    // 0x01FC (0x000C)
[0x0000000000000022] (CPF_Const | CPF_Net)
class UPrimitiveComponent*                   CollisionComponent;                    // 0x0208
(0x0008) [0x00000000040A0009] (CPF_Edit | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
int32_t                            OverlapTag;                          // 0x0210 (0x0004)
[0x0000000000001000] (CPF_Native)
struct FRotator                    RotationRate;                        // 0x0214 (0x000C)
[0x0000000000000001] (CPF_Edit)
class AActor*                      PendingTouch;                        // 0x0220 (0x0008)
[0x0000000000000000]
TArray<class UClass*>                   SupportedEvents;                    // 0x0228 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class USequenceEvent*>                GeneratedEvents;                    // 0x0238
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class USeqAct_Latent*>                LatentActions;                      // 0x0248 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class UClass*>                   IgnoredTouchClasses;                // 0x0258 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Actor");
}

return uClassPointer;
};

class USpriteComponent* GetActorSpriteComponent();
class UActorComponent* GetComponent(class UClass* ComponentClass);
class AActor* SpawnInstance(class AActor* SpawnOwner, struct FName SpawnTag, struct
FVector SpawnLocation, struct FRotator SpawnRotation, unsigned long bNoCollisionFail);
int32_t GetActorMetrics(uint8_t MetricsType);
bool eventOnMobileTouch(class APlayerController* InPC, struct FVector2D TouchLocation);
bool IsMobileTouchEnabled();
bool ShouldBeHiddenBySHOW_NavigationNodes();
void PrintDebugInfo(class UDebugDrawer* Drawer);
void ForceNetUpdatePacket();
void ForceNetUpdate();
bool WillOverlap(struct FVector PosA, struct FVector VelA, struct FVector PosB, struct FVector
VelB, float StepSize, float Radius, float& Time);
struct FVector GetAvoidanceVector(struct FVector GoalLocation, float CollisionRadius, float
MaxSpeed, int32_t NumSamples, float VelocityStepRate, float MaxTimeTilOverlap, TArray<class
AActor*>& Obstacles);
void eventReplicationEnded();
void eventPostDemoRewind();
void eventAnimTreeUpdated(class USkeletalMeshComponent* SkelMesh);
bool SupportsKismetModification(class USequenceOp* AskingOp, class FString& Reason);
```

```
class UParticleSystem* GetAnimTrailParticleSystem(class UAnimNotify_Trails* AnimNotifyData);
void eventTrailsNotifyEnd(class UAnimNotify_Trails* AnimNotifyData);
void eventTrailsNotifyTick(class UAnimNotify_Trails* AnimNotifyData);
void eventTrailsNotify(class UAnimNotify_Trails* AnimNotifyData);
bool eventCreateForceField(class UAnimNotify_ForceField* AnimNotifyData);
bool eventPlayParticleEffect(class UAnimNotify_PlayParticleEffect* AnimNotifyData);
void GetAimAdhesionExtent(float& Width, float& Height, struct FVector& Center);
void GetAimFrictionExtent(float& Width, float& Height, struct FVector& Center);
bool IsInPersistentLevel(unsigned long bIncludeLevelStreamingPersistent);
void eventOnRigidBodySpringOverextension(class URB_BodyInstance* BodyInstance);
static struct FGuid GetPackageGuid(struct FName PackageName);
void eventPostInitAnimTree(class USkeletalMeshComponent* SkelComp);
void eventRootMotionExtracted(class USkeletalMeshComponent* SkelComp, struct
FBoneAtom& ExtractedRootMotionDelta);
void eventRootMotionProcessed(class USkeletalMeshComponent* SkelComp);
void eventRootMotionModeChanged(class USkeletalMeshComponent* SkelComp);
void eventPostRenderFor(class APlayerController* PC, class UCanvas* Canvas, struct FVector
CameraPosition, struct FVector CameraDir);
void NativePostRenderFor(class APlayerController* PC, class UCanvas* Canvas, struct FVector
CameraPosition, struct FVector CameraDir);
void SetHUDLocation(struct FVector NewHUDLocation);
void eventRigidBodyCollision(class UPrimitiveComponent* HitComponent, class
UPrimitiveComponent* OtherComponent, int32_t ContactIndex, struct FCollisionImpactData&
RigidCollisionData);
void eventInterpolationChanged(class USeqAct_Interp* InterpAction);
void eventInterpolationFinished(class USeqAct_Interp* InterpAction);
void eventInterpolationStarted(class USeqAct_Interp* InterpAction, class UInterpGroupInst*
GroupInst);
void eventSpawnedByKismet();
struct FVector GetTargetLocation(class AActor* RequestedBy, unsigned long
bRequestAlternateLoc);
void FindGoodEndView(class APlayerController* PC, struct FRotator& GoodRotation);
uint8_t eventScriptGetTeamNum();
uint8_t GetTeamNum();
bool IsPlayerOwned();
void eventGetActorEyesViewPoint(struct FVector& out_Location, struct FRotator& out_Rotation);
bool IsStationary();
class UFaceFXAsset* eventGetActorFaceFXAsset();
bool CanActorPlayFaceFXAnim();
bool IsActorPlayingFaceFXAnim();
void eventTickSkelControl(float DeltaTime, class USkeletalMeshComponent* SkelComp, class
USkelControlBase* SkelCtrl);
void eventSetSkelControlScale(struct FName SkelControlName, float Scale);
void eventSetMorphWeight(struct FName MorphNodeName, float MorphWeight);
void eventStopActorFaceFXAnim();
bool eventPlayActorFaceFXAnim(class UFaceFXAnimSet* AnimSet, class FString GroupName,
class FString SeqName, class USoundCue* SoundCueToPlay, class UAkEvent* AkEventToPlay);
void eventFinishAnimControl(class UInterpGroup* InInterpGroup);
void eventSetAnimPosition(struct FName SlotName, int32_t ChannelIndex, struct FName
InAnimSeqName, float InPosition, unsigned long bFireNotifies, unsigned long bLooping, unsigned
long bEnableRootMotion);
void eventBeginAnimControl(class UInterpGroup* InInterpGroup);
void eventOnAnimPlay(class UAnimNodeSequence* SeqNode);
void eventOnAnimEnd(class UAnimNodeSequence* SeqNode, float PlayedTime, float
ExcessTime);
void DoKismetAttachment(class AActor* Attachment, class USeqAct_AttachToActor* Action);
void OnAttachToActor(class USeqAct_AttachToActor* Action);
void OnToggleHidden(class USeqAct_ToggleHidden* Action);
void OnChangeCollision(class USeqAct_ChangeCollision* Action);
void OnSetPhysics(class USeqAct_SetPhysics* Action);
void OnSetBlockRigidBody(class USeqAct_SetBlockRigidBody* Action);
void OnSetVelocity(class USeqAct_SetVelocity* Action);
void OnTeleport(class USeqAct_Teleport* Action);
void PrestreamTextures(float Seconds, unsigned long bEnableStreaming, int32_t
CinematicTextureGroups);
void eventShutDown();
void SetNetUpdateTime(float NewUpdateTime);
void eventForceNetRelevant();
void OnDestroy(class USeqAct_Destroy* Action);
void ClearLatentAction(class UClass* actionClass, unsigned long bAborted, class
USeqAct_Latent* exceptionAction);
bool FindEventsOfClass(class UClass* EventClass, unsigned long bIncludeDisabled,
TArray<class USequenceEvent*>& out_EventList);
bool ActivateEventClass(class UClass* InClass, class AActor* InInstigator, unsigned long bTest,
TArray<class USequenceEvent*>& EventList, TArray<int32_t>& ActivateIndices, TArray<class
USequenceEvent*>& ActivatedEvents);
bool TriggerGlobalEventClass(class UClass* InEventClass, class AActor* InInstigator, int32_t
ActivateIndex);
void eventReceivedNewEvent(class USequenceEvent* Evt);
bool TriggerEventClass(class UClass* InEventClass, class AActor* InInstigator, int32_t
ActivateIndex, unsigned long bTest, TArray<class USequenceEvent*>& ActivatedEvents);
void eventDebugMessagePlayer(class FString msg);
bool ImpactEffectIsRelevant(class APawn* EffectInstigator, struct FVector SpawnLocation,
unsigned long bForceDedicated, float VisibleCullDistance, float HiddenCullDistance, unsigned
long bSkipLOSCheck);
bool ActorEffectIsRelevant(class APawn* EffectInstigator, unsigned long bForceDedicated, float
VisibleCullDistance, float HiddenCullDistance);
bool EffectIsRelevant(struct FVector SpawnLocation, unsigned long bForceDedicated, float
VisibleCullDistance, float HiddenCullDistance);
void ApplyFluidSurfaceImpact(class AFluidSurfaceActor* Fluid, struct FVector HitLocation);
bool CanSplash();
void PlayTeleportEffect(unsigned long bOut, unsigned long bSound);
void eventReset();
class UAudioComponent* eventGetFaceFXAudioComponent();
void eventModifyHearSoundComponent(class UAudioComponent* AC);
class FString GetPhysicsName();
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
class FString GetDebugName();
void MatchStarting();
static class FString GetLocalString(int32_t Switch, class APlayerReplicationInfo* RelatedPRI,
class APlayerReplicationInfo* RelatedPRI01);
static void ReplaceText(class FString Replace, class FString With, class FString& Text);
class FString GetHumanReadableName();
class FString GetItemName(class FString FullName);
bool CalcCamera(float fDeltaTime, struct FVector& out_CamLoc, struct FRotator& out_CamRot,
float& out_FOV);
void eventEndViewTarget(class APlayerController* PC);
void eventBecomeViewTarget(class APlayerController* PC);
```

```
bool CheckForErrors();
void eventDebugFreezeGame(class AActor* ActorToLookAt);
struct FVector GetGravityAcceleration();
struct FVector GetGravityDirection();
float GetGravityZ();
void eventNotifySkelControlBeyondLimit(class USkelControlLookAt* LookAt);
void eventConstraintBrokenNotify(class AActor* ConOwner, class URB_ConstraintSetup*
ConSetup, class URB_ConstraintInstance* ConInstance);
void eventSetInitialState();
void eventPostBeginPlay();
void eventPreBeginPlay();
class APlayerController* GetALocalPlayerController();
void LocalPlayerControllers(class UClass* BaseClass, class APlayerController*& PC);
void AllOwnedComponents(class UClass* BaseClass, class UActorComponent*&
OutComponent);
void ComponentList(class UClass* BaseClass, class UActorComponent*& out_Component);
void OverlappingActors(class UClass* BaseClass, float Radius, struct FVector Loc, unsigned long
bIgnoreHidden, class AActor*& out_Actor);
void CollidingActors(class UClass* BaseClass, float Radius, struct FVector Loc, unsigned long
bUseOverlapCheck, class UClass* InterfaceClass, class AActor*& Actor, struct FTraceHitInfo&
HitInfo);
void VisibleCollidingActors(class UClass* BaseClass, float Radius, struct FVector Loc, unsigned
long bIgnoreHidden, struct FVector Extent, unsigned long bTraceActors, class UClass*
InterfaceClass, class AActor*& Actor, struct FTraceHitInfo& HitInfo);
void VisibleActors(class UClass* BaseClass, float Radius, struct FVector Loc, class AActor*&
Actor);
void TraceActors(class UClass* BaseClass, struct FVector End, struct FVector Start, struct
FVector Extent, int32_t ExtraTraceFlags, class AActor*& Actor, struct FVector& HitLoc, struct
FVector& HitNorm, struct FTraceHitInfo& HitInfo);
void TouchingActors(class UClass* BaseClass, class AActor*& Actor);
void BasedActors(class UClass* BaseClass, class AActor*& Actor);
void ChildActors(class UClass* BaseClass, class AActor*& Actor);
void DynamicActors(class UClass* BaseClass, class UClass* InterfaceClass, class AActor*&
Actor);
void AllActors(class UClass* BaseClass, class UClass* InterfaceClass, class AActor*& Actor);
class FString GetURLMap();
void PostTeleport(class ATeleporter* OutTeleporter);
bool PreTeleport(class ATeleporter* InTeleporter);
struct FVector GetDestination(class AController* C);
bool CalculateMinSpeedTrajectory(struct FVector End, struct FVector Start, float MaxTossSpeed,
float MinTossSpeed, struct FVector CollisionSize, float TerminalVelocity, float GravityZ, unsigned
long bOnlyTraceUp, struct FVector& out_Velocity);
bool SuggestTossVelocity(struct FVector Destination, struct FVector Start, float TossSpeed, float
BaseTossZ, float DesiredZPct, struct FVector CollisionSize, float TerminalVelocity, float
OverrideGravityZ, unsigned long bOnlyTraceUp, struct FVector& TossVelocity);
bool PlayerCanSeeMe(unsigned long bForceLOSCheck);
void MakeNoise(float Loudness, struct FName NoiseType);
void ActivateOcclusion(unsigned long bInActivate);
void PostTrigger(struct FName InTrigger);
void SetSwitch(struct FName InSwitchGroup, struct FName InSwitch);
void SetState(struct FName InStateGroup, struct FName InState);
void SetRTPCValue(struct FName InRTPC, float TargetValue);
void PostAkEventOnBone(class UAkEvent* InAkEvent, struct FName BoneName);
void PostAkEvent(class UAkEvent* InAkEvent);
void PlaySoundBase(class UAkBaseSoundObject* InSoundCue, unsigned long bNotReplicated, unsigned long bNoRepToOwner, unsigned long bStopWhenOwnerDestroyed, struct FVector SoundLocation, unsigned long bNoRepToRelevant);
void PlayAkEvent(class UAkEvent* InSoundCue, unsigned long bNotReplicated, unsigned long bNoRepToOwner, unsigned long bStopWhenOwnerDestroyed, struct FVector SoundLocation, unsigned long bNoRepToRelevant);
void PlaySound(class USoundCue* InSoundCue, unsigned long bReplicated, unsigned long bNoRepToOwner, unsigned long bStopWhenOwnerDestroyed, struct FVector SoundLocation, unsigned long bNoRepToRelevant);
class UAudioComponent* CreateAudioComponent(class USoundCue* InSoundCue, unsigned long bPlay, unsigned long bStopWhenOwnerDestroyed, unsigned long bUseLocation, struct FVector SourceLocation, unsigned long bAttachToSelf);
void ResetTimerTimeDilation(struct FName TimerName, class UObject* inObj);
void ModifyTimerTimeDilation(struct FName TimerName, float InTimerTimeDilation, class UObject* inObj);
float GetRemainingTimeForTimer(struct FName TimerFuncName, class UObject* inObj);
float GetTimerRate(struct FName TimerFuncName, class UObject* inObj);
float GetTimerCount(struct FName inTimerFunc, class UObject* inObj);
bool IsTimerActive(struct FName inTimerFunc, class UObject* inObj);
void PauseTimer(unsigned long bPause, struct FName inTimerFunc, class UObject* inObj);
void ClearAllTimers(class UObject* inObj);
void ClearTimer(struct FName inTimerFunc, class UObject* inObj);
void SetStateTimer(float InRate, unsigned long inbLoop, struct FName inTimerFunc);
void SetTimer(float InRate, unsigned long inbLoop, struct FName inTimerFunc, class UObject* inObj);
void eventTornOff();
bool Destroy();
class AActor* Spawn(class UClass* SpawnClass, class AActor* SpawnOwner, struct FName SpawnTag, struct FVector SpawnLocation, struct FRotator SpawnRotation, class AActor* ActorTemplate, unsigned long bNoCollisionFail);
bool IsBlockedBy(class AActor* Other);
void GetBoundingCylinder(float& CollisionRadius, float& CollisionHeight);
void GetComponentsBoundingBox(struct FBox& ActorBox);
bool IsOverlapping(class AActor* A);
bool ContainsPoint(struct FVector Spot);
bool FindSpot(struct FVector BoxExtent, struct FVector& SpotLocation);
bool TraceAllPhysicsAssetInteractions(class USkeletalMeshComponent* SkelMeshComp, struct FVector EndTrace, struct FVector StartTrace, struct FVector Extent, TArray<struct FImpactInfo>& out_Hits);
bool FastTrace(struct FVector TraceEnd, struct FVector TraceStart, struct FVector BoxExtent, unsigned long bTraceBullet);
bool PointCheckComponent(class UPrimitiveComponent* InComponent, struct FVector PointLocation, struct FVector PointExtent);
bool TraceComponent(class UPrimitiveComponent* InComponent, struct FVector TraceEnd, struct FVector TraceStart, struct FVector Extent, unsigned long bComplexCollision, struct FVector& HitLocation, struct FVector& HitNormal, struct FTraceHitInfo& HitInfo);
class AActor* Trace(struct FVector TraceEnd, struct FVector TraceStart, unsigned long bTraceActors, struct FVector Extent, int32_t ExtraTraceFlags, struct FVector& HitLocation, struct FVector& HitNormal, struct FTraceHitInfo& HitInfo);
void eventOutsideWorldBounds();
void eventFellOutOfWorld();
bool UsedBy(class APawn* User);
bool eventOverRotated(struct FRotator& out_Desired, struct FRotator& out_Actual);
bool ClampRotation(struct FRotator rBase, struct FRotator rUpperLimits, struct FRotator
rLowerLimits, struct FRotator& out_Rot);
void eventOnSleepRBPhysics();
void eventOnWakeRBPhysics();
void eventRanInto(class AActor* Other);
void eventEncroachedBy(class AActor* Other);
bool eventEncroachingOn(class AActor* Other);
void eventCollisionChanged();
class AActor* eventSpecialHandling(class APawn* Other);
void eventDetach(class AActor* Other);
void eventAttach(class AActor* Other);
void eventBaseChange();
void eventBump(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitNormal);
void eventUnTouch(class AActor* Other);
void eventPostTouch(class AActor* Other);
void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
void eventPhysicsVolumeChange(class APhysicsVolume* NewVolume);
void eventLanded(struct FVector HitNormal, class AActor* FloorActor);
void eventFalling();
void eventHitWall(struct FVector HitNormal, class AActor* Wall, class UPrimitiveComponent*
WallComp);
void eventTimer();
void eventOnOwnerChanged();
void eventTick(float DeltaTime);
void eventLostChild(class AActor* Other);
void eventGainedChild(class AActor* Other);
void eventDestroyForMapUnload();
void eventDestroyed();
void SetTickIsDisabled(unsigned long bInDisabled);
void SetTickGroup(uint8_t NewTickGroup);
void ReattachComponent(class UActorComponent* ComponentToReattach);
void DetachComponent(class UActorComponent* ExComponent);
void AttachComponent(class UActorComponent* NewComponent);
void UnClock(float& Time);
void Clock(float& Time);
void SetPhysics(uint8_t newPhysics);
void SetOnlyOwnerSee(unsigned long bNewOnlyOwnerSee);
void SetHidden(unsigned long bNewHidden);
void ChartData(class FString DataName, float DataValue);
static void FlushDebugStrings();
static void DrawDebugFrustrum(uint8_t R, uint8_t G, uint8_t B, unsigned long bPersistentLines,
struct FMatrix& FrustumToWorld);
static void DrawDebugString(struct FVector TextLocation, class FString Text, class AActor*
TestBaseActor, struct FColor TextColor, float Duration);
static void DrawDebugCone(struct FVector Origin, struct FVector Direction, float Length, float
AngleWidth, float AngleHeight, int32_t NumSides, struct FColor DrawColor, unsigned long
bPersistentLines);
static void DrawDebugCylinder(struct FVector Start, struct FVector End, float Radius, int32_t
Segments, uint8_t R, uint8_t G, uint8_t B, unsigned long bPersistentLines);
static void DrawDebugSphere(struct FVector Center, float Radius, int32_t Segments, uint8_t R,
uint8_t G, uint8_t B, unsigned long bPersistentLines);
static void DrawDebugCoordinateSystem(struct FVector AxisLoc, struct FRotator AxisRot, float
Scale, unsigned long bPersistentLines);
```

```
static void DrawDebugStar(struct FVector Position, float Size, uint8_t R, uint8_t G, uint8_t B,
unsigned long bPersistentLines);
static void DrawDebugBox(struct FVector Center, struct FVector Extent, uint8_t R, uint8_t G,
uint8_t B, unsigned long bPersistentLines);
static void DrawDebugPoint(struct FVector Position, float Size, struct FLinearColor PointColor,
unsigned long bPersistentLines);
static void DrawDebugLine(struct FVector LineStart, struct FVector LineEnd, uint8_t R, uint8_t G,
uint8_t B, unsigned long bPersistentLines);
static void FlushPersistentDebugLines();
static struct FVector GetBasedPosition(struct FBasedPosition BP);
static void SetBasedPosition(struct FVector pos, class AActor* ForcedBase, struct
FBasedPosition& BP);
static struct FVector BP2Vect(struct FBasedPosition BP);
static void Vect2BP(struct FVector pos, class AActor* ForcedBase, struct FBasedPosition& BP);
void SetForcedInitialReplicatedProperty(class UProperty* PropToReplicate, unsigned long bAdd);
void eventReplicatedEvent(struct FName VarName);
struct FVector GetAggregateBaseVelocity(class AActor* TestBase);
bool IsOwnedBy(class AActor* TestActor);
class AActor* GetBaseMost();
bool IsBasedOn(class AActor* TestActor);
void SearchForBaseBelow(float HeightBelow, class AActor*& NewBase, struct FVector&
HitNormal);
void FindBase();
void SetOwner(class AActor* NewOwner);
void SetBase(class AActor* NewBase, struct FVector NewFloor, class
USkeletalMeshComponent* SkelComp, struct FName AttachName);
float GetTerminalVelocity();
void AutonomousPhysics(float DeltaSeconds);
bool MoveSmooth(struct FVector delta);
int32_t fixedTurn(int32_t Current, int32_t Desired, int32_t DeltaRate);
void SetShadowParentOnAllAttachedComponents(class UPrimitiveComponent* MyPrimComp,
class ULightEnvironmentComponent* MyLightEnv);
void SetHardAttach(unsigned long bNewHardAttach);
bool SetRelativeLocation(struct FVector NewLocation);
bool SetRelativeRotation(struct FRotator NewRotation);
void SetZone(unsigned long bForceRefresh);
uint8_t MovingWhichWay(float& Amount);
bool SetRotation(struct FRotator NewRotation);
bool SetLocation(struct FVector NewLocation);
bool Move(struct FVector delta);
void SetDrawScale3D(struct FVector NewScale3D);
void SetDrawScale(float NewScale);
void SetCollisionType(uint8_t NewCollisionType);
void SetCollisionSize(float NewRadius, float NewHeight);
void SetCollision(unsigned long bNewColActors, unsigned long bNewBlockActors, unsigned long
bNewIgnoreEncroachers);
void FinishAnim(class UAnimNodeSequence* SeqNode, unsigned long bFinishOnBlendOut);
void Sleep(float Seconds);
class FString ConsoleCommand(class FString Command, unsigned long bWriteToLog);
void ForceUpdateComponents(unsigned long bCollisionUpdate, unsigned long bTransformOnly);
};

// Class Engine.Info
// 0x0000 (0x0268 - 0x0268)
```

```
class AInfo : public AActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Info");
}

return uClassPointer;
};

};

// Class Engine.ZoneInfo
// 0x000C (0x0268 - 0x0274)
class AZoneInfo : public AInfo
{
public:
float                                   KillZ;                                  // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   SoftKill;                                // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                              bSoftKillZ : 1;                          // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ZoneInfo");
}

return uClassPointer;
};

};

// Class Engine.WorldInfo
// 0x08DC (0x0274 - 0x0B50)
class AWorldInfo : public AZoneInfo
{
public:
struct FPostProcessSettings                      DefaultPostProcessSettings;                 // 0x0278
(0x0168) [0x0000000000404001] (CPF_Edit | CPF_Config | CPF_NeedCtorLink)
class UPostProcessChain*                       WorldPostProcessChain;                   // 0x03E0
```

```
(0x0008) [0x0000000000000001] (CPF_Edit)
unsigned long                    bPersistPostProcessToNextLevel : 1;          // 0x03E8
(0x0004) [0x0000000000004001] [0x00000001] (CPF_Edit | CPF_Config)
unsigned long                    bFogEnabled : 1;                      // 0x03E8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bBumpOffsetEnabled : 1;                   // 0x03E8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    bUseGammaCorrection : 1;                   // 0x03E8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bMapNeedsLightingFullyRebuilt : 1;           // 0x03E8
(0x0004) [0x0000000000000000] [0x00000010]
unsigned long                    bMapHasMultipleDominantLightsAffectingOnePrimitive : 1;//
0x03E8 (0x0004) [0x0000000000000000] [0x00000020]
unsigned long                    bMapHasPathingErrors : 1;                  // 0x03E8 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                    bRequestedBlockOnAsyncLoading : 1;           // 0x03E8
(0x0004) [0x0000000000000000] [0x00000080]
unsigned long                    bBegunPlay : 1;                        // 0x03E8 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                    bPlayersOnly : 1;                      // 0x03E8 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                    bPlayersOnlyPending : 1;                   // 0x03E8 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                    bSuspendAI : 1;                        // 0x03E8 (0x0004)
[0x0000000000000000] [0x00000800]
unsigned long                    bDropDetail : 1;                       // 0x03E8 (0x0004)
[0x0000000000002000] [0x00001000] (CPF_Transient)
unsigned long                    bAggressiveLOD : 1;                     // 0x03E8 (0x0004)
[0x0000000000002000] [0x00002000] (CPF_Transient)
unsigned long                    bStartup : 1;                        // 0x03E8 (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                    bPathsRebuilt : 1;                      // 0x03E8 (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                    bHasPathNodes : 1;                      // 0x03E8 (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                    bIsMenuLevel : 1;                      // 0x03E8 (0x0004)
[0x0000000000002002] [0x00020000] (CPF_Const | CPF_Transient)
unsigned long                    bDebugPauseExecution : 1;                  // 0x03E8 (0x0004)
[0x0000000800002002] [0x00040000] (CPF_Const | CPF_Transient)
unsigned long                    bDebugStepExecution : 1;                  // 0x03E8 (0x0004)
[0x0000000800002002] [0x00080000] (CPF_Const | CPF_Transient)
unsigned long                    bUseConsoleInput : 1;                    // 0x03E8 (0x0004)
[0x0000000000002000] [0x00100000] (CPF_Transient)
unsigned long                    bDisableGlobalGravityZ : 1;                 // 0x03E8 (0x0004)
[0x0000000000044002] [0x00200000] (CPF_Const | CPF_Config | CPF_GlobalConfig)
unsigned long                    bMinimizeBSPSections : 1;                  // 0x03E8 (0x0004)
[0x0000000000000001] [0x00400000] (CPF_Edit)
unsigned long                    bNoPathWarnings : 1;                    // 0x03E8 (0x0004)
[0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                    bNoMobileMapWarnings : 1;                  // 0x03E8 (0x0004)
[0x0000000000004001] [0x01000000] (CPF_Edit | CPF_Config)
unsigned long                    bHighPriorityLoading : 1;                  // 0x03E8 (0x0004)
[0x0000000000000020] [0x02000000] (CPF_Net)
unsigned long                    bHighPriorityLoadingLocal : 1;               // 0x03E8 (0x0004)
```

```
[0x0000000000000000] [0x04000000]
unsigned long                          bUseProcBuildingRulesetOverride : 1;         // 0x03E8
(0x0004) [0x0000000000000001] [0x08000000] (CPF_Edit)
unsigned long                          bInteractiveMode : 1;                  // 0x03E8 (0x0004)
[0x0000000000002002] [0x10000000] (CPF_Const | CPF_Transient)
unsigned long                          bForceInteractiveMode : 1;                // 0x03E8 (0x0004)
[0x0000000000000001] [0x20000000] (CPF_Edit)
unsigned long                          bSupportDoubleBufferedPhysics : 1;          // 0x03E8
(0x0004) [0x0000000000000001] [0x40000000] (CPF_Edit)
unsigned long                          bPhysicsIgnoreDeltaTime : 1;               // 0x03E8 (0x0004)
[0x0000000000000001] [0x80000000] (CPF_Edit)
unsigned long                          bEnableChanceOfPhysicsChunkOverride : 1;      // 0x03EC
(0x0004) [0x0000000000004001] [0x00000001] (CPF_Edit | CPF_Config)
unsigned long                          bLimitExplosionChunkSize : 1;              // 0x03EC (0x0004)
[0x0000000000004001] [0x00000002] (CPF_Edit | CPF_Config)
unsigned long                          bLimitDamageChunkSize : 1;                // 0x03EC (0x0004)
[0x0000000000004001] [0x00000004] (CPF_Edit | CPF_Config)
unsigned long                          bPrecomputeVisibility : 1;               // 0x03EC (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bPlaceCellsOnSurfaces : 1;               // 0x03EC (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bAffectIndirectCompositeShadowDirection : 1;   // 0x03EC
(0x0004) [0x0000000000004001] [0x00000020] (CPF_Edit | CPF_Config)
unsigned long                          bAllowTemporalAA : 1;                   // 0x03EC (0x0004)
[0x0000000000044001] [0x00000040] (CPF_Edit | CPF_Config | CPF_GlobalConfig)
unsigned long                          bUseGlobalIllumination : 1;              // 0x03EC (0x0004)
[0x0000000800000001] [0x00000080] (CPF_Edit)
unsigned long                          bForceNoPrecomputedLighting : 1;           // 0x03EC
(0x0004) [0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                          bHaveActiveCrowd : 1;                   // 0x03EC (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                          bAllowHostMigration : 1;                 // 0x03EC (0x0004)
[0x0000000000004000] [0x00000400] (CPF_Config)
unsigned long                          bGameplayFramePause : 1;                 // 0x03EC (0x0004)
[0x0000000000000000] [0x00000800]
float                            SquintModeKernelSize;                   // 0x03F0 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
class APostProcessVolume*                    HighestPriorityPostProcessVolume;          // 0x03F8
(0x0008) [0x0000000001002002] (CPF_Const | CPF_Transient)
struct FReverbSettings                      DefaultReverbSettings;                 // 0x0400 (0x0010)
[0x0000000000004001] (CPF_Edit | CPF_Config)
struct FInteriorSettings                     DefaultAmbientZoneSettings;              // 0x0410
(0x0024) [0x0000000000004001] (CPF_Edit | CPF_Config)
float                            FogStart;                        // 0x0434 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            FogEnd;                         // 0x0438 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                          FogColor;                        // 0x043C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            BumpEnd;                         // 0x0440 (0x0004)
[0x0000000000000001] (CPF_Edit)
class AReverbVolume*                       HighestPriorityReverbVolume;              // 0x0448
(0x0008) [0x0000000001002002] (CPF_Const | CPF_Transient)
TArray<class AMassiveLODOverrideVolume*>          MassiveLODOverrideVolumes;                //
```

```
0x0450 (0x0010) [0x0000000001402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class APortalVolume*>                    PortalVolumes;                          // 0x0460 (0x0010)
[0x0000000001402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class AEnvironmentVolume*>               EnvironmentVolumes;                     // 0x0470
(0x0010) [0x0000000001402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class ULevelStreaming*>                  StreamingLevels;                        // 0x0480
(0x0010) [0x0000000004420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink |
CPF_EditInline)
struct FDouble                                  LastTimeUnbuiltLightingWasEncountered;          // 0x0490
(0x0008) [0x0000000000002000] (CPF_Transient)
class UBookMark*                                BookMarks[0xA];                         // 0x0498 (0x0050)
[0x0000000800000001] (CPF_Edit)
class UKismetBookMark*                          KismetBookMarks[0xA];                   // 0x04E8
(0x0050) [0x0000000800000001] (CPF_Edit)
TArray<class UClipPadEntry*>                    ClipPadEntries;                         // 0x0538 (0x0010)
[0x0000000804400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)
float                                           TimeDilation;                           // 0x0548 (0x0004)
[0x0000004000000020] (CPF_Net)
float                                           DemoPlayTimeDilation;                   // 0x054C (0x0004)
[0x0008000000000000]
float                                           TimeSeconds;                            // 0x0550 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                           RealTimeSeconds;                        // 0x0554 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                           RealDeltaSeconds;                       // 0x0558 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                           AudioTimeSeconds;                       // 0x055C (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                           DeltaSeconds;                           // 0x0560 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                           PauseDelay;                             // 0x0564 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                           RealTimeToUnPause;                      // 0x0568 (0x0004)
[0x0000000000002000] (CPF_Transient)
class APlayerReplicationInfo*                   Pauser;                                 // 0x0570 (0x0008)
[0x0000000100000020] (CPF_Net)
class FString                                   VisibleGroups;                          // 0x0578 (0x0010)
[0x0000000820400000] (CPF_NeedCtorLink | CPF_Deprecated)
class FString                                   VisibleLayers;                          // 0x0588 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
class UTexture2D*                               DefaultTexture;                         // 0x0598 (0x0008)
[0x0000000000000000]
class UTexture2D*                               WireframeTexture;                       // 0x05A0 (0x0008)
[0x0000000000000000]
class UTexture2D*                               WhiteSquareTexture;                     // 0x05A8 (0x0008)
[0x0000000000000000]
class UTexture2D*                               LargeVertex;                            // 0x05B0 (0x0008)
[0x0000000000000000]
class UTexture2D*                               BSPVertex;                              // 0x05B8 (0x0008)
[0x0000000000000000]
TArray<class FString>                           DeferredExecs;                          // 0x05C0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class AGameReplicationInfo*                     GRI;                                    // 0x05D0 (0x0008)
[0x0000000000002000] (CPF_Transient)
```

```
uint8_t                          NetMode;                              // 0x05D8 (0x0001)
[0x0000000000000000]
uint8_t                          NextTravelType;                       // 0x05D9 (0x0001)
[0x0000000000000000]
uint8_t                          VisibilityAggressiveness;             // 0x05DA (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          PreferredLightmapType;                // 0x05DB (0x0001)
[0x0000000000000000]
uint8_t                          LevelLightingQuality;                 // 0x05DC (0x0001)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
class FString                    ComputerName;                         // 0x05E0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    EngineVersion;                        // 0x05F0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    MinNetVersion;                        // 0x0600 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class AGameInfo*                 Game;                                 // 0x0610 (0x0008)
[0x0000000000000000]
float                            StallZ;                               // 0x0618 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            WorldGravityZ;                        // 0x061C (0x0004)
[0x0000000000002020] (CPF_Net | CPF_Transient)
float                            DefaultGravityZ;                      // 0x0620 (0x0004)
[0x0000000000044002] (CPF_Const | CPF_Config | CPF_GlobalConfig)
float                            GlobalGravityZ;                       // 0x0624 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RBPhysicsGravityScaling;              // 0x0628 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FVector                   GlobalGravityDirection;               // 0x062C (0x000C)
[0x0000000000000000]
class ANavigationPoint*          NavigationPointList;                  // 0x0638 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class AController*               ControllerList;                       // 0x0640 (0x0008)
[0x0000000000000002] (CPF_Const)
class APawn*                     PawnList;                             // 0x0648 (0x0008)
[0x0000000000000002] (CPF_Const)
class ACoverLink*                CoverList;                            // 0x0650 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class APylon*                    PylonList;                            // 0x0658 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                            MoveRepSize;                          // 0x0660 (0x0004)
[0x0000000000000000]
TArray<struct FNetViewer>        ReplicationViewers;                   // 0x0668 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class FString                    NextURL;                              // 0x0678 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                            NextSwitchCountdown;                  // 0x0688 (0x0004)
[0x0000000000000000]
int32_t                          PackedLightAndShadowMapTextureSize;   // 0x068C
(0x0004) [0x0000000000000001] (CPF_Edit)
struct FVector                   DefaultColorScale;                    // 0x0690 (0x000C)
[0x0000000000000001] (CPF_Edit)
class UClass*                    DefaultGameType;                      // 0x06A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
TArray<class UClass*>                    GameTypesSupportedOnThisMap;              // 0x06A8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class UClass*                       GameTypeForPIE;                        // 0x06B8 (0x0008)
[0x0000000800000001] (CPF_Edit)
TArray<class UObject*>                   ClientDestroyedActorContent;              // 0x06C0
(0x0010) [0x0000000000420002] (CPF_Const | CPF_EditConst | CPF_NeedCtorLink)
TArray<struct FName>                     PreparingLevelNames;                   // 0x06D0
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
struct FName                        CommittedPersistentLevelName;              // 0x06E0
(0x0008) [0x0000000000002002] (CPF_Const | CPF_Transient)
class UObjectReferencer*                  PersistentMapForcedObjects;               // 0x06E8
(0x0008) [0x0000000000000000]
class UAudioComponent*                     MusicComp;                          // 0x06F0 (0x0008)
[0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component | CPF_EditInline)
struct FMusicTrackStruct                   CurrentMusicTrack;                      // 0x06F8 (0x0030)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FMusicTrackStruct                   ReplicatedMusicTrack;                    // 0x0728 (0x0030)
[0x0000000100402020] (CPF_Net | CPF_Transient | CPF_NeedCtorLink)
class FString                      Title;                            // 0x0758 (0x0010)
[0x0000000000408003] (CPF_Edit | CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                      Author;                           // 0x0768 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class UMapInfo*                       MyMapInfo;                        // 0x0778 (0x0008)
[0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
class FString                      EmitterPoolClassPath;                   // 0x0780 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class AEmitterPool*                     MyEmitterPool;                      // 0x0790 (0x0008)
[0x0000000000002000] (CPF_Transient)
class FString                      DecalManagerClassPath;                   // 0x0798 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class ADecalManager*                     MyDecalManager;                        // 0x07A8 (0x0008)
[0x0000000000002000] (CPF_Transient)
class FString                      FractureManagerClassPath;                 // 0x07B0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class AFractureManager*                   MyFractureManager;                      // 0x07C0
(0x0008) [0x0000000000002000] (CPF_Transient)
class FString                      ParticleEventManagerClassPath;             // 0x07C8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class AParticleEventManager*                MyParticleEventManager;                 // 0x07D8
(0x0008) [0x0000000000002000] (CPF_Transient)
class UProcBuildingRuleset*                 ProcBuildingRulesetOverride;              // 0x07E0
(0x0008) [0x0000000000000001] (CPF_Edit)
int32_t                          SkelMeshCompTickTagCount;                // 0x07E8 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          MaxPhysicsDeltaTime;                   // 0x07EC (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          MaxPhysicsSubsteps;                    // 0x07F0 (0x0004)
[0x0000000000004000] (CPF_Config)
struct FPhysXSceneProperties                 PhysicsProperties;                      // 0x07F4
(0x003C) [0x0000000004000001] (CPF_Edit | CPF_EditInline)
TArray<struct FCompartmentRunList>            CompartmentRunFrames;                  //
0x0830 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                          DefaultSkinWidth;                    // 0x0840 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                              ApexLODResourceBudget;                    // 0x0844 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ApexDestructionLODResourceValue;            // 0x0848 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ApexClothingLODResourceValue;              // 0x084C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FApexModuleDestructibleSettings          DestructibleSettings;                      // 0x0850
(0x0014) [0x0000000000000001] (CPF_Edit)
class UPhysicsLODVerticalEmitter*              EmitterVertical;                          // 0x0868
(0x0008) [0x0000000000000000]
struct FPhysXVerticalProperties                VerticalProperties;                       // 0x0870 (0x0018)
[0x0000000004000001] (CPF_Edit | CPF_EditInline)
TArray<struct FPointer>                        WorldAttractors;                       // 0x0888 (0x0010)
[0x0000000000001000] (CPF_Native)
TArray<class FString>                          ConsoleTypeNames;                        // 0x0898 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
float                              ChanceOfPhysicsChunkOverride;              // 0x08A8 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
float                              MaxExplosionChunkSize;                    // 0x08AC (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
float                              MaxDamageChunkSize;                       // 0x08B0 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
float                              FractureExplosionVelScale;                // 0x08B4 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
int32_t                              MaxNumFacturedChunksToSpawnInAFrame;         // 0x08B8
(0x0004) [0x0000000000000001] (CPF_Edit)
int32_t                              NumFacturedChunksSpawnedThisFrame;          // 0x08BC
(0x0004) [0x0000000000002000] (CPF_Transient)
float                              FracturedMeshWeaponDamage;                 // 0x08C0 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                              VisibilityCellSize;                     // 0x08C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CharacterLitIndirectBrightness;             // 0x08C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CharacterLitIndirectContrastFactor;          // 0x08CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CharacterShadowedIndirectBrightness;         // 0x08D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CharacterShadowedIndirectContrastFactor;      // 0x08D4
(0x0004) [0x0000000000000001] (CPF_Edit)
float                              CharacterLightingContrastFactor;            // 0x08D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                        ImageReflectionEnvironmentTexture;           // 0x08E0
(0x0008) [0x0000000000000001] (CPF_Edit)
struct FLinearColor                      ImageReflectionEnvironmentColor;            // 0x08E8
(0x0010) [0x0000000000000001] (CPF_Edit)
float                              ImageReflectionEnvironmentRotation;          // 0x08F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FMap_Mirror                       ScreenMessages;                          // 0x0900 (0x0050)
[0x0000000000003000] (CPF_Native | CPF_Transient)
TArray<struct FScreenMessageString>            PriorityScreenMessages;                    // 0x0950
(0x0010) [0x0000000000003000] (CPF_Native | CPF_Transient)
int32_t                              MaxTrianglesPerLeaf;                   // 0x0960 (0x0004)
[0x0000000800000000]
```

```
class ULightmassLevelSettings*                 LMLevelSettings;                          // 0x0968
(0x0008) [0x0000000824400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline |
CPF_Deprecated)
uint8_t                                UnknownData00[0x50];                   // 0x0970 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.WorldInfo.LandscapeInfoMap
struct FLightmassWorldInfoSettings              LightmassSettings;                         // 0x09C0
(0x0058) [0x0000000000000001] (CPF_Edit)
class UPitchTekSettings*                    PitchTek;                                  // 0x0A18 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                                UnknownData01[0x50];                   // 0x0A20 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.WorldInfo.NavMeshPathConstraintCache
uint8_t                                UnknownData02[0x50];                   // 0x0A70 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.WorldInfo.NavMeshPathGoalEvaluatorCache
class ACrowdPopulationManagerBase*             PopulationManager;                         // 0x0AC0
(0x0008) [0x0000000000000000]
struct FHostMigrationState                  PeerHostMigration;                        // 0x0AC8 (0x0028)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
float                                  HostMigrationTimeout;                  // 0x0AF0 (0x0004)
[0x0000000000004000] (CPF_Config)
class APhysicsVolume*                       FirstPhysicsVolume;                       // 0x0AF8 (0x0008)
[0x0000000000002000] (CPF_Transient)
class UObject*                           GameShare;                              // 0x0B00 (0x0008)
[0x0000000000002000] (CPF_Transient)
struct FScriptDelegate                     __EventPauseChanged__Delegate;              // 0x0B08
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                     __EventTimeDilationChanged__Delegate;        // 0x0B20
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                     __DemoPlayTimeDilation__ChangeNotify;        // 0x0B38
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WorldInfo");
}

return uClassPointer;
};

void __DemoPlayTimeDilation__ChangeNotifyFunc();
void PrintDebugInfo(class UDebugDrawer* Drawer);
bool NeedsLightingRebuild();
void EventTimeDilationChanged(class AWorldInfo* WI);
void eventSetTimeDilation(float NewVal);
void ClearObjectPools();
void ToggleHostMigration(unsigned long bEnabled);
void eventNotifyHostMigrationStateChanged(uint8_t NewState, uint8_t OldState);
bool BeginHostMigration();
bool eventCanBeginHostMigration();
class AEnvironmentVolume* FindEnvironmentVolume(struct FVector TestLocation);
```

```
static class AWorldInfo* GetWorldInfo();
struct FWorldFractureSettings GetWorldFractureSettings();
void DoMemoryTracking();
bool GetDemoRewindPoints(TArray<int32_t>& OutRewindPoints);
void GetDemoFrameInfo(int32_t& CurrentFrame, int32_t& TotalFrames);
bool IsPlayingDemo();
bool IsRecordingDemo();
uint8_t GetDetailMode();
class FString GetMapName(unsigned long bIncludePrefix);
void SetMapInfo(class UMapInfo* NewMapInfo);
class UMapInfo* GetMapInfo();
void SetSeamlessTravelMidpointPause(unsigned long bNowPaused);
bool IsInSeamlessTravel();
void SeamlessTravel(class FString URL, unsigned long bAbsolute, struct FGuid
MapPackageGuid);
void CommitMapChange();
void CancelPendingMapChange();
bool IsMapChangeReady();
bool IsPreparingMapChange();
void PrepareMapChange(TArray<struct FName>& LevelNames);
void NotifyMatchStarted(unsigned long bShouldActivateLevelStartupEvents, unsigned long
bShouldActivateLevelBeginningEvents, unsigned long bShouldActivateLevelLoadedEvents);
void AllClientConnections(class UPlayer*& ClientConnection, struct FIpAddr& ClientIP, int32_t&
ClientPort);
void AllPawns(class UClass* BaseClass, struct FVector TestLocation, float TestRadius, class
APawn*& P);
void AllControllers(class UClass* BaseClass, class AController*& C);
void NavigationPointCheck(struct FVector Point, struct FVector Extent, TArray<class
ANavigationPoint*>& Navs, TArray<class UReachSpec*>& Specs);
void RadiusNavigationPoints(class UClass* BaseClass, struct FVector Point, float Radius, class
ANavigationPoint*& N);
void AllNavigationPoints(class UClass* BaseClass, class ANavigationPoint*& N);
void Reset();
void PostBeginPlay();
void PreBeginPlay();
void ThisIsNeverExecuted(class ADefaultPhysicsVolume* P);
void eventServerTravel(class FString URL, unsigned long bAbsolute, unsigned long
bShouldSkipGameNotify);
class UClass* GetGameClass();
class FString GetAddressURL();
void VerifyNavList();
static void ForceGarbageCollection(unsigned long bFullPurge);
static bool IsHDREnabled();
static bool IsPlayInMobilePreview();
static bool IsPlayInPreview();
static bool IsPlayInEditor();
static bool IsWithGFx();
static bool IsEpicGamesStoreBuild();
static class FString GetConsoleTypeName(uint8_t ConsoleType);
static uint8_t GetConsoleType();
static bool IsConsoleBuild(uint8_t ConsoleType);
static bool IsDemoBuild();
class FString GetLocalURL();
void SetLevelRBGravity(struct FVector NewGrav);
void AllSequenceObjects(class UClass* SeqClass, class USequenceObject*& OutObj);
TArray<class USequence*> GetAllRootSequences();
class USequence* GetGameSequence();
struct FVector GetGravityDirection();
float GetGravityZ();
void UpdateMusicTrack(struct FMusicTrackStruct NewMusicTrack);
void SetMusicVolume(float VolumeMultiplier);
static bool IsMenuLevel(class FString MapName);
void AddOnScreenDebugMessage(int32_t Key, float TimeToDisplay, struct FColor DisplayColor,
class FString DebugMessage);
void eventReplicatedEvent(struct FName VarName);
class UNavMeshPathGoalEvaluator* GetNavMeshPathGoalEvaluatorFromCache(class UClass*
GoalEvalClass, class UNavigationHandle* Requestor);
class UNavMeshPathConstraint* GetNavMeshPathConstraintFromCache(class UClass*
ConstraintClass, class UNavigationHandle* Requestor);
void ReleaseCachedConstraintsAndEvaluators();
void SetPitchTekTargetsInitialState();
class AActor* GetOrSpawnActor(class UClass* ActorClass);
class AActor* GetActor(class UClass* ActorClass);
void SetPauser(class APlayerReplicationInfo* InPauser);
void EventPauseChanged();
};

// Class Engine.DownloadableContentEnumerator
// 0x0048 (0x0060 - 0x00A8)
class UDownloadableContentEnumerator : public UObject
{
public:
TArray<struct FOnlineContent>          DLCBundles;                    // 0x0060 (0x0010)
[0x0000008000400000] (CPF_NeedCtorLink)
class FString                          DLCRootDir;                    // 0x0070 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FScriptDelegate>         FindDLCDelegates;              // 0x0080
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                 __OnFindDLCComplete__Delegate;          // 0x0090
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DownloadableContentEnumerator");
}

return uClassPointer;
};

void TriggerFindDLCDelegates();
void InstallDLC(class FString DLCName);
void InstallAllDLC();
void DeleteDLC(class FString DLCName);
void ClearFindDLCDelegate(struct FScriptDelegate InDelegate);
void AddFindDLCDelegate(struct FScriptDelegate InDelegate);
void OnFindDLCComplete();
void FindDLC();
};

// Class Engine.DownloadableContentManager
// 0x00C0 (0x0060 - 0x0120)
class UDownloadableContentManager : public UObject
{
public:
TArray<struct FPointer>                DLCConfigCacheChanges;                // 0x0060
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class FString>                  InstalledDLC;                         // 0x0070 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
struct FMap_Mirror                     NonPackageFilePathMap;                // 0x0080 (0x0050)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class UClass*>                  ClassesToReload;                      // 0x00D0 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class UObject*>                 ObjectsToReload;                      // 0x00E0 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class FString>                  QueuedFullyLoadPackageInis;           // 0x00F0
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
class UGameEngine*                     GameEngine;                           // 0x0100 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FScriptDelegate                 __OnRefreshComplete__Delegate;        // 0x0108
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DownloadableContentManager");
}

return uClassPointer;
};

void OnRefreshComplete();
void AddPackagesToFullyLoad(class FString Filename);
void RefreshDLCEnumComplete();
void RefreshDLC();
void OnContentChange();
void OnStorageDeviceChange();
void OnLoginChange(uint8_t LocalUserNum);
void eventInit();
void InstallNonPackageFiles(struct FOnlineContent& DLCBundle);
void InstallPackages(struct FOnlineContent& DLCBundle);
void UpdateObjectLists();
void MarkPerObjectConfigPendingKill(class FString Section);
void AddSectionToObjectList(class FString Section);
```

```
bool GetDLCNonPackageFilePath(struct FName NonPackageFileName, class FString& Path);
void ClearDLC();
void InstallDLCs(TArray<struct FOnlineContent>& DLCBundles);
bool InstallDLC(struct FOnlineContent& DLCBundle);
};

// Class Engine.Engine
// 0x08F0 (0x0068 - 0x0958)
class UEngine : public USubsystem
{
public:
class FString                    GameShareClassName;                 // 0x0068 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                    GameShareClass;                     // 0x0078 (0x0008)
[0x0000000000000000]
class FString                    EngineShareClassName;               // 0x0080 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UEngineShare*              EngineShare;                        // 0x0090 (0x0008)
[0x0000000000002000] (CPF_Transient)
int32_t                          BuildID;                            // 0x0098 (0x0004)
[0x0000000000004000] (CPF_Config)
class UNetDriverSecurity*        NetDriverSecurity;                  // 0x00A0 (0x0008)
[0x0000000000000000]
class UFont*                     TinyFont;                           // 0x00A8 (0x0008)
[0x0000000000000000]
class FString                    TinyFontName;                       // 0x00B0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UFont*                     SmallFont;                          // 0x00C0 (0x0008)
[0x0000000000000000]
class FString                    SmallFontName;                      // 0x00C8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UFont*                     MediumFont;                         // 0x00D8 (0x0008)
[0x0000000000000000]
class FString                    MediumFontName;                     // 0x00E0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UFont*                     LargeFont;                          // 0x00F0 (0x0008)
[0x0000000000000000]
class FString                    LargeFontName;                      // 0x00F8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UFont*                     SubtitleFont;                       // 0x0108 (0x0008)
[0x0000000000000000]
class FString                    SubtitleFontName;                   // 0x0110 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UFont*                     ScalableFont;                       // 0x0120 (0x0008)
[0x0000000000000000]
class FString                    ScalableFontName;                   // 0x0128 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
TArray<class UFont*>             AdditionalFonts;                    // 0x0138 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class FString>            AdditionalFontNames;                // 0x0148 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UClass*                    ConsoleClass;                       // 0x0158 (0x0008)
[0x0000000000000000]
class FString                    ConsoleClassName;                   // 0x0160 (0x0010)
```

```
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UClass*                          GameViewportClientClass;                   // 0x0170 (0x0008)
[0x0000000000000000]
class FString                          GameViewportClientClassName;               // 0x0178 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UClass*                          DataStoreClientClass;                      // 0x0188 (0x0008)
[0x0000000000000000]
class FString                          DataStoreClientClassName;                  // 0x0190 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UClass*                          LocalPlayerClass;                          // 0x01A0 (0x0008)
[0x0000000000000000]
class FString                          LocalPlayerClassName;                      // 0x01A8 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UMaterial*                       DefaultMaterial;                           // 0x01B8 (0x0008)
[0x0000000000000000]
class FString                          DefaultMaterialName;                       // 0x01C0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       DefaultDecalMaterial;                      // 0x01D0 (0x0008)
[0x0000000000000000]
class FString                          DefaultDecalMaterialName;                  // 0x01D8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture*                        DefaultTexture;                            // 0x01E8 (0x0008)
[0x0000000000000000]
class FString                          DefaultTextureName;                        // 0x01F0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       WireframeMaterial;                         // 0x0200 (0x0008)
[0x0000000000000000]
class FString                          WireframeMaterialName;                     // 0x0208 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       EmissiveTexturedMaterial;                  // 0x0218 (0x0008)
[0x0000000000000000]
class FString                          EmissiveTexturedMaterialName;              // 0x0220 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       GeomMaterial;                              // 0x0230 (0x0008)
[0x0000000000000000]
class FString                          GeomMaterialName;                          // 0x0238 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       DefaultFogVolumeMaterial;                  // 0x0248 (0x0008)
[0x0000000000000000]
class FString                          DefaultFogVolumeMaterialName;              // 0x0250
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       TickMaterial;                              // 0x0260 (0x0008)
[0x0000000000000000]
class FString                          TickMaterialName;                          // 0x0268 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       CrossMaterial;                             // 0x0278 (0x0008)
[0x0000000000000000]
class FString                          CrossMaterialName;                         // 0x0280 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       LevelColorationLitMaterial;                // 0x0290 (0x0008)
[0x0000000000000000]
class FString                          LevelColorationLitMaterialName;            // 0x0298 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                       LevelColorationUnlitMaterial;              // 0x02A8 (0x0008)
```

```
[0x0000000000000000]
class FString                    LevelColorationUnlitMaterialName;          // 0x02B0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   LightingTexelDensityMaterial;             // 0x02C0 (0x0008)
[0x0000000000000000]
class FString                    LightingTexelDensityName;                 // 0x02C8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   ShadedLevelColorationLitMaterial;          // 0x02D8
(0x0008) [0x0000000000000000]
class FString                    ShadedLevelColorationLitMaterialName;        // 0x02E0
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   ShadedLevelColorationUnlitMaterial;         // 0x02F0
(0x0008) [0x0000000000000000]
class FString                    ShadedLevelColorationUnlitMaterialName;       // 0x02F8
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   RemoveSurfaceMaterial;                   // 0x0308 (0x0008)
[0x0000000000000000]
class FString                    RemoveSurfaceMaterialName;                // 0x0310 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   VertexColorMaterial;                    // 0x0320 (0x0008)
[0x0000000000000000]
class FString                    VertexColorMaterialName;                 // 0x0328 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   VertexColorViewModeMaterial_ColorOnly;       // 0x0338
(0x0008) [0x0000000000000000]
class FString                    VertexColorViewModeMaterialName_ColorOnly;    // 0x0340
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   VertexColorViewModeMaterial_AlphaAsColor;     // 0x0350
(0x0008) [0x0000000000000000]
class FString                    VertexColorViewModeMaterialName_AlphaAsColor;  //
0x0358 (0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   VertexColorViewModeMaterial_RedOnly;         // 0x0368
(0x0008) [0x0000000000000000]
class FString                    VertexColorViewModeMaterialName_RedOnly;      // 0x0370
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   VertexColorViewModeMaterial_GreenOnly;       // 0x0380
(0x0008) [0x0000000000000000]
class FString                    VertexColorViewModeMaterialName_GreenOnly;    // 0x0388
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   VertexColorViewModeMaterial_BlueOnly;        // 0x0398
(0x0008) [0x0000000000000000]
class FString                    VertexColorViewModeMaterialName_BlueOnly;     // 0x03A0
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   HeatmapMaterial;                        // 0x03B0 (0x0008)
[0x0000000000000000]
class FString                    HeatmapMaterialName;                     // 0x03B8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   BoneWeightMaterial;                     // 0x03C8 (0x0008)
[0x0000000000000000]
class FString                    BoneWeightMaterialName;                  // 0x03D0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                   TangentColorMaterial;                   // 0x03E0 (0x0008)
[0x0000000000000000]
class FString                    TangentColorMaterialName;                // 0x03E8 (0x0010)
```

```
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                         MobileEmulationMasterMaterial;             // 0x03F8
(0x0008) [0x0000000000000000]
class FString                            MobileEmulationMasterMaterialName;          // 0x0400
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                         ProcBuildingSimpleMaterial;                // 0x0410 (0x0008)
[0x0000000000000000]
class FString                            ProcBuildingSimpleMaterialName;             // 0x0418 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UStaticMesh*                         BuildingQuadStaticMesh;                  // 0x0428 (0x0008)
[0x0000000000000000]
class FString                            BuildingQuadStaticMeshName;                // 0x0430 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
float                                ProcBuildingLODColorTexelsPerWorldUnit;       // 0x0440 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                                ProcBuildingLODLightingTexelsPerWorldUnit;     // 0x0444
(0x0004) [0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                               MaxProcBuildingLODColorTextureSize;          // 0x0448 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                               MaxProcBuildingLODLightingTextureSize;        // 0x044C
(0x0004) [0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
unsigned long                           UseProcBuildingLODTextureCropping : 1;        // 0x0450
(0x0004) [0x0000000000044000] [0x00000001] (CPF_Config | CPF_GlobalConfig)
unsigned long                           ForcePowerOfTwoProcBuildingLODTextures : 1;    // 0x0450
(0x0004) [0x0000000000044000] [0x00000002] (CPF_Config | CPF_GlobalConfig)
unsigned long                           bCombineSimilarMappings : 1;               // 0x0450 (0x0004)
[0x0000000000044000] [0x00000004] (CPF_Config | CPF_GlobalConfig)
unsigned long                           bRenderLightMapDensityGrayscale : 1;          // 0x0450
(0x0004) [0x0000000000044000] [0x00000008] (CPF_Config | CPF_GlobalConfig)
unsigned long                           bScreenshotRequested : 1;                  // 0x0450 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                           bUseSound : 1;                         // 0x0450 (0x0004)
[0x0000000000002000] [0x00000020] (CPF_Transient)
unsigned long                           bUseBackgroundLevelStreaming : 1;            // 0x0450
(0x0004) [0x0000000000004001] [0x00000040] (CPF_Edit | CPF_Config)
unsigned long                           bSubtitlesEnabled : 1;                   // 0x0450 (0x0004)
[0x0000000000004001] [0x00000080] (CPF_Edit | CPF_Config)
unsigned long                           bSubtitlesForcedOff : 1;                  // 0x0450 (0x0004)
[0x0000000000004001] [0x00000100] (CPF_Edit | CPF_Config)
unsigned long                           bSmoothFrameRate : 1;                    // 0x0450 (0x0004)
[0x0000000000004000] [0x00000200] (CPF_Config)
unsigned long                           bCheckForMultiplePawnsSpawnedInAFrame : 1;     // 0x0450
(0x0004) [0x0000000000004000] [0x00000400] (CPF_Config)
unsigned long                           bShouldGenerateSimpleLightmaps : 1;          // 0x0450
(0x0004) [0x0000000000044000] [0x00000800] (CPF_Config | CPF_GlobalConfig)
unsigned long                           bForceStaticTerrain : 1;                  // 0x0450 (0x0004)
[0x0000000000004001] [0x00001000] (CPF_Edit | CPF_Config)
unsigned long                           bForceCPUSkinning : 1;                   // 0x0450 (0x0004)
[0x0000000000004000] [0x00002000] (CPF_Config)
unsigned long                           bUsePostProcessEffects : 1;                // 0x0450 (0x0004)
[0x0000000000004000] [0x00004000] (CPF_Config)
unsigned long                           bOnScreenKismetWarnings : 1;               // 0x0450 (0x0004)
[0x0000000000004000] [0x00008000] (CPF_Config)
unsigned long                           bEnableKismetLogging : 1;                  // 0x0450 (0x0004)
```

```
[0x0000000000004000] [0x00010000] (CPF_Config)
unsigned long                    bAllowMatureLanguage : 1;              // 0x0450 (0x0004)
[0x0000000000004000] [0x00020000] (CPF_Config)
unsigned long                    bDisablePhysXHardwareSupport : 1;           // 0x0450
(0x0004) [0x0000000000044000] [0x00040000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bEnablePhysX : 1;                      // 0x0450 (0x0004)
[0x0000000000004000] [0x00080000] (CPF_Config)
unsigned long                    bPauseOnLossOfFocus : 1;                // 0x0450 (0x0004)
[0x0000000000004000] [0x00100000] (CPF_Config)
unsigned long                    bCheckParticleRenderSize : 1;             // 0x0450 (0x0004)
[0x0000000000044000] [0x00200000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bEnableColorClear : 1;                 // 0x0450 (0x0004)
[0x0000000000044002] [0x00400000] (CPF_Const | CPF_Config | CPF_GlobalConfig)
unsigned long                    bEnableSwitchRenderMode : 1;             // 0x0450 (0x0004)
[0x0000000000044002] [0x00800000] (CPF_Const | CPF_Config | CPF_GlobalConfig)
unsigned long                    bShowAllHiddenObjects : 1;               // 0x0450 (0x0004)
[0x0000000000044002] [0x01000000] (CPF_Const | CPF_Config | CPF_GlobalConfig)
unsigned long                    bAreConstraintsDirty : 1;               // 0x0450 (0x0004)
[0x0000000000002000] [0x02000000] (CPF_Transient)
unsigned long                    bHasPendingGlobalReattach : 1;            // 0x0450 (0x0004)
[0x0000000000002000] [0x04000000] (CPF_Transient)
unsigned long                    bEnableOnScreenDebugMessages : 1;          // 0x0450
(0x0004) [0x0000000000044000] [0x08000000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bEnableOnScreenDebugMessagesDisplay : 1;     // 0x0450
(0x0004) [0x0000000000002000] [0x10000000] (CPF_Transient)
unsigned long                    bSuppressMapWarnings : 1;               // 0x0450 (0x0004)
[0x0000000000044000] [0x20000000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bCookSeparateSharedMPGameContent : 1;       // 0x0450
(0x0004) [0x0000000000044000] [0x40000000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bUseRecastNavMesh : 1;                 // 0x0450 (0x0004)
[0x0000000000044000] [0x80000000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bDisableAILogging : 1;                 // 0x0454 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                    bUseNormalMapsForSimpleLightMaps : 1;       // 0x0454
(0x0004) [0x0000000000044000] [0x00000002] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bStartWithMatineeCapture : 1;             // 0x0454 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                    bCompressMatineeCapture : 1;              // 0x0454 (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
unsigned long                    bLockReadOnlyLevels : 1;                // 0x0454 (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
float                       MaxRMSDForCombiningMappings;             // 0x0458 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                      ImageReflectionTextureSize;            // 0x045C (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FLinearColor                   LightingOnlyBrightness;               // 0x0460 (0x0010)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
TArray<struct FColor>                   LightComplexityColors;                // 0x0470 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
TArray<struct FLinearColor>                ShaderComplexityColors;               // 0x0480
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
float                       MaxPixelShaderAdditiveComplexityCount;       // 0x0490 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                       MinTextureDensity;                     // 0x0494 (0x0004)
```

```
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          IdealTextureDensity;                    // 0x0498 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          MaxTextureDensity;                      // 0x049C (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          MinLightMapDensity;                     // 0x04A0 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          IdealLightMapDensity;                   // 0x04A4 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          MaxLightMapDensity;                     // 0x04A8 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          RenderLightMapDensityGrayscaleScale;          // 0x04AC (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          RenderLightMapDensityColorScale;             // 0x04B0 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FLinearColor                 LightMapDensityVertexMappedColor;            // 0x04B4
(0x0010) [0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FLinearColor                 LightMapDensitySelectedColor;               // 0x04C4
(0x0010) [0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
TArray<struct FStatColorMapping>              StatColorMappings;                      // 0x04D8
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                    EditorBrushMaterial;                    // 0x04E8 (0x0008)
[0x0000000000000000]
class FString                     EditorBrushMaterialName;                  // 0x04F0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UPhysicalMaterial*                 DefaultPhysMaterial;                    // 0x0500 (0x0008)
[0x0000000000000000]
class FString                     DefaultPhysMaterialName;                  // 0x0508 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UPhysicalMaterial*                 LandscapeHolePhysMaterial;                // 0x0518
(0x0008) [0x0000000000000000]
class FString                     LandscapeHolePhysMaterialName;              // 0x0520
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UApexDestructibleDamageParameters*         ApexDamageParams;                         //
0x0530 (0x0008) [0x0000000000000000]
class FString                     ApexDamageParamsName;                   // 0x0538 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                    TerrainErrorMaterial;                   // 0x0548 (0x0008)
[0x0000000000000000]
class FString                     TerrainErrorMaterialName;                 // 0x0550 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
int32_t                         TerrainMaterialMaxTextureCount;             // 0x0560 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                         TerrainTessellationCheckCount;             // 0x0564 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          TerrainTessellationCheckDistance;           // 0x0568 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
class UClass*                     OnlineSubsystemClass;                   // 0x0570 (0x0008)
[0x0000000000000000]
class FString                     DefaultOnlineSubsystemName;               // 0x0578 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UPostProcessChain*                 DefaultPostProcess;                     // 0x0588
(0x0008) [0x0000000000000000]
class FString                     DefaultPostProcessName;                   // 0x0590 (0x0010)
```

```
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UPostProcessChain*                    ThumbnailSkeletalMeshPostProcess;           //
0x05A0 (0x0008) [0x0000000000000000]
class FString                           ThumbnailSkeletalMeshPostProcessName;        // 0x05A8
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UPostProcessChain*                    ThumbnailParticleSystemPostProcess;          //
0x05B8 (0x0008) [0x0000000000000000]
class FString                           ThumbnailParticleSystemPostProcessName;       // 0x05C0
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UPostProcessChain*                    ThumbnailMaterialPostProcess;              // 0x05D0
(0x0008) [0x0000000000000000]
class FString                           ThumbnailMaterialPostProcessName;            // 0x05D8
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UPostProcessChain*                    DefaultUIScenePostProcess;                 // 0x05E8
(0x0008) [0x0000000000000000]
class FString                           DefaultUIScenePostProcessName;              // 0x05F0
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UMaterial*                          DefaultUICaretMaterial;                  // 0x0600 (0x0008)
[0x0000000000000000]
class FString                           DefaultUICaretMaterialName;                // 0x0608 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                          SceneCaptureReflectActorMaterial;           // 0x0618
(0x0008) [0x0000000000000000]
class FString                           SceneCaptureReflectActorMaterialName;        // 0x0620
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UMaterial*                          SceneCaptureCubeActorMaterial;             // 0x0630
(0x0008) [0x0000000000000000]
class FString                           SceneCaptureCubeActorMaterialName;          // 0x0638
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                           ScreenDoorNoiseTexture;                  // 0x0648 (0x0008)
[0x0000000000000000]
class FString                           ScreenDoorNoiseTextureName;                // 0x0650 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                           ImageGrainNoiseTexture;                  // 0x0660 (0x0008)
[0x0000000000000000]
class FString                           ImageGrainNoiseTextureName;                // 0x0668 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                           RandomAngleTexture;                     // 0x0678 (0x0008)
[0x0000000000000000]
class FString                           RandomAngleTextureName;                   // 0x0680 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                           RandomNormalTexture;                    // 0x0690 (0x0008)
[0x0000000000000000]
class FString                           RandomNormalTextureName;                  // 0x0698 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                           RandomMirrorDiscTexture;                 // 0x06A8 (0x0008)
[0x0000000000000000]
class FString                           RandomMirrorDiscTextureName;               // 0x06B0
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture*                           WeightMapPlaceholderTexture;               // 0x06C0
(0x0008) [0x0000000000000000]
class FString                           WeightMapPlaceholderTextureName;            // 0x06C8
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                           LightMapDensityTexture;                  // 0x06D8 (0x0008)
```

```
[0x0000000000000000]
class FString                          LightMapDensityTextureName;              // 0x06E0 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                          SMAAAreaTexture;                     // 0x06F0 (0x0008)
[0x0000000000000000]
class FString                          SMAAAreaTextureName;                     // 0x06F8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                          SMAASearchTexture;                   // 0x0708 (0x0008)
[0x0000000000000000]
class FString                          SMAASearchTextureName;                   // 0x0710 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class UTexture2D*                          LightMapDensityNormal;               // 0x0720 (0x0008)
[0x0000000000000000]
class FString                          LightMapDensityNormalName;               // 0x0728 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
class USoundNodeWave*                           DefaultSound;                   // 0x0738 (0x0008)
[0x0000000000000000]
class FString                          DefaultSoundName;                    // 0x0740 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
float                             TimeBetweenPurgingPendingKillObjects;         // 0x0750 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
class UClient*                         Client;                          // 0x0758 (0x0008)
[0x0000000000000002] (CPF_Const)
TArray<class ULocalPlayer*>                     GamePlayers;                    // 0x0760 (0x0010)
[0x0000000000500000] (CPF_NeedCtorLink)
class UGameViewportClient*                      GameViewport;                   // 0x0770 (0x0008)
[0x0000000000000002] (CPF_Const)
TArray<class FString>                       DeferredCommands;                   // 0x0778 (0x0010)
[0x0000000000500000] (CPF_NeedCtorLink)
int32_t                              TickCycles;                        // 0x0788 (0x0004)
[0x0000000000000000]
int32_t                              GameCycles;                        // 0x078C (0x0004)
[0x0000000000000000]
int32_t                              ClientCycles;                      // 0x0790 (0x0004)
[0x0000000000000000]
float                             MaxSmoothedFrameRate;                     // 0x0794 (0x0004)
[0x0000000000004000] (CPF_Config)
float                             MinSmoothedFrameRate;                     // 0x0798 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                              NumPawnsAllowedToBeSpawnedInAFrame;          // 0x079C
(0x0004) [0x0000000000004000] (CPF_Config)
struct FPointer                        RemoteControlExec;                       // 0x07A0 (0x0008)
[0x0000000000001000] (CPF_Native)
struct FPointer                        MobileMaterialEmulator;                  // 0x07A8 (0x0008)
[0x0000000000001000] (CPF_Native)
struct FColor                          C_WorldBox;                          // 0x07B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                          C_BrushWire;                         // 0x07B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                          C_AddWire;                           // 0x07B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                          C_SubtractWire;                      // 0x07BC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                          C_SemiSolidWire;                     // 0x07C0 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_NonSolidWire;                      // 0x07C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_WireBackground;                    // 0x07C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_ScaleBoxHi;                        // 0x07CC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_VolumeCollision;                   // 0x07D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_BSPCollision;                      // 0x07D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_OrthoBackground;                   // 0x07D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_Volume;                            // 0x07DC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    C_BrushShape;                        // 0x07E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            StreamingDistanceFactor;             // 0x07E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
class FString                    ScoutClassName;                      // 0x07E8 (0x0010)
[0x0000000000404002] (CPF_Const | CPF_Config | CPF_NeedCtorLink)
uint8_t                          TransitionType;                      // 0x07F8 (0x0001)
[0x0000000000000000]
class FString                    TransitionDescription;               // 0x0800 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    TransitionGameType;                  // 0x0810 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                            MeshLODRange;                        // 0x0820 (0x0004)
[0x0000000000004000] (CPF_Config)
float                            CameraRotationThreshold;             // 0x0824 (0x0004)
[0x0000000000004000] (CPF_Config)
float                            CameraTranslationThreshold;          // 0x0828 (0x0004)
[0x0000000000004000] (CPF_Config)
float                            PrimitiveProbablyVisibleTime;        // 0x082C (0x0004)
[0x0000000000004000] (CPF_Config)
float                            PercentUnoccludedRequeries;          // 0x0830 (0x0004)
[0x0000000000004000] (CPF_Config)
float                            MaxOcclusionPixelsFraction;          // 0x0834 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                          MaxFluidNumVerts;                    // 0x0838 (0x0004)
[0x0000000000004000] (CPF_Config)
float                            FluidSimulationTimeLimit;            // 0x083C (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                          MaxParticleResize;                   // 0x0840 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                          MaxParticleResizeWarn;               // 0x0844 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                          MaxParticleVertexMemory;             // 0x0848 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                          MaxParticleSpriteCount;              // 0x084C (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                          MaxParticleSubUVCount;               // 0x0850 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                          BeginUPTryCount;                     // 0x0854 (0x0004)
```

```
[0x0000000000004000] (CPF_Config)
TArray<struct FDropNoteInfo>                PendingDroppedNotes;                    // 0x0858
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
class FString                               DynamicCoverMeshComponentName;            // 0x0868
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
float                                       NetClientTicksPerSecond;                  // 0x0878 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                                       MaxTrackedOcclusionIncrement;             // 0x087C (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                                       TrackedOcclusionStepSize;                 // 0x0880 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FLinearColor                         DefaultSelectedMaterialColor;             // 0x0884 (0x0010)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FLinearColor                         DefaultHoveredMaterialColor;              // 0x0894 (0x0010)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FLinearColor                         SelectedMaterialColor;                    // 0x08A4 (0x0010)
[0x0000000000002000] (CPF_Transient)
struct FLinearColor                         UnselectedMaterialColor;                  // 0x08B4 (0x0010)
[0x0000000000002000] (CPF_Transient)
TArray<struct FName>                        IgnoreSimulatedFuncWarnings;              // 0x08C8
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
int32_t                                     ScreenSaverInhibitorSemaphore;            // 0x08D8 (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FPointer                             ScreenSaverInhibitor;                     // 0x08E0 (0x0008)
[0x0000000000002000] (CPF_Transient)
class UTranslationContext*                  GlobalTranslationContext;                 // 0x08E8
(0x0008) [0x0000000000000000]
struct FDouble                              LoadingMovieStartTime;                    // 0x08F0 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class FString                               MatineeCaptureName;                       // 0x08F8 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
class FString                               MatineePackageCaptureName;                // 0x0908 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
class FString                               VisibleLevelsForMatineeCapture;           // 0x0918 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                                     MatineeCaptureFPS;                        // 0x0928 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                     MatineeCaptureType;                       // 0x092C (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                       MapLoadTimePC;                            // 0x0930 (0x0004)
[0x0000000000004000] (CPF_Config)
float                                       MapLoadTimePS4;                           // 0x0934 (0x0004)
[0x0000000000004000] (CPF_Config)
float                                       MapLoadTimeXboxOne;                       // 0x0938 (0x0004)
[0x0000000000004000] (CPF_Config)
float                                       MapLoadTimeSwitch;                        // 0x093C (0x0004)
[0x0000000000004000] (CPF_Config)
struct FScriptDelegate                      __EventPreLaunchURL__Delegate;            // 0x0940
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Engine");
}

return uClassPointer;
};

void PrintDebugInfo(class UDebugDrawer* Drawer);
float GetPlatformMapLoadTime(uint8_t Platform);
static struct FName GetFeatureName();
static class UFeatureSystem* GetFeatureSystem();
class UNetworkEncryptionKey* GetNetworkSecurityKeyByPlayerID(struct FUniqueNetId&
PlayerID);
void SetNetworkSecurityKey(class UNetworkEncryptionKey* SecurityKey, struct FUniqueNetId&
PlayerID);
void AddNetworkSecurityKey(class UNetworkEncryptionKey* SecurityKey, struct FUniqueNetId&
PlayerID);
float GetSystemSettingFloat(class FString SettingName);
int32_t GetSystemSettingInt(class FString SettingName);
bool GetSystemSettingBool(class FString SettingName);
void EventPreLaunchURL(class FString URL);
static bool LaunchURL(class FString URL);
static uint8_t BasicLoadObject(class UObject* Obj, class FString PathName, unsigned long
bIsSaveGame, int32_t Version);
static bool BasicSaveObject(class UObject* Obj, class FString PathName, unsigned long
bIsSaveGame, int32_t Version, unsigned long bEncrypt);
void AddTextureStreamingSlaveLoc(struct FVector InLoc, float BoostFactor, unsigned long
bOverrideLocation, float OverrideDuration);
static class UPostProcessChain* GetWorldPostProcessChain();
static class UPostProcessChain* GetDefaultPostProcessChain();
static class UEngine* GetEngine();
static bool IsRealDStereoEnabled();
static void AddOverlayWrapped(class UFont* Font, class FString Text, float X, float Y, float
ScaleX, float ScaleY, float WrapWidth);
static void AddOverlay(class UFont* Font, class FString Text, float X, float Y, float ScaleX, float
ScaleY, unsigned long bIsCentered);
static void RemoveAllOverlays();
static void StopMovie(unsigned long bDelayStopUntilGameHasRendered);
static bool PlayLoadMapMovie();
static class FString GetLastMovieName();
static class UAudioDevice* GetAudioDevice();
static bool IsUsingES2Renderer();
static bool IsStereoscopic3D();
static bool IsSplitScreen();
static class UFont* GetAdditionalFont(int32_t AdditionalFontIndex);
static class UFont* GetSubtitleFont();
static class UFont* GetScalableFont();
static class UFont* GetLargeFont();
static class UFont* GetMediumFont();
static class UFont* GetSmallFont();
static class UFont* GetTinyFont();
static bool HasNetworkConnection();
static class FString BuildBugSubmissionString(class FString BugField, class FString
BugFieldData);
static class FString GetDevicePushNotificationToken();
static float GetOSVersion();
static class FString GetDeviceUUID();
static class FString GetBuildDate();
static class AWorldInfo* GetCurrentWorldInfo();
static bool IsGame();
static bool IsEditor();
static bool UseSecurePackets();
};

// Class Engine.GameEngine
// 0x01F0 (0x0958 - 0x0B48)
class UGameEngine : public UEngine
{
public:
class UPendingLevel*                GPendingLevel;                  // 0x0958 (0x0008)
[0x0000000000000000]
class FString                  PendingLevelPlayerControllerClassName;       // 0x0960
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
struct FURL                    LastURL;                        // 0x0970 (0x0060)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FURL                    LastRemoteURL;                    // 0x09D0 (0x0060)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class FString>              ServerActors;                    // 0x0A30 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                  TravelURL;                       // 0x0A40 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                     TravelType;                      // 0x0A50 (0x0001)
[0x0000000000000000]
unsigned long                  bWorldWasLoadedThisTick : 1;             // 0x0A54 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                  bCheckForMovieCapture : 1;              // 0x0A54 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                  bTriggerPostLoadMap : 1;               // 0x0A54 (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                  bStartedLoadMapMovie : 1;              // 0x0A54 (0x0004)
[0x0000000000002002] [0x00000008] (CPF_Const | CPF_Transient)
unsigned long                  bUnloadingMap : 1;                  // 0x0A54 (0x0004)
[0x0000000000002002] [0x00000010] (CPF_Const | CPF_Transient)
unsigned long                  bShouldCommitPendingMapChange : 1;          // 0x0A54
(0x0004) [0x0000000000000002] [0x00000020] (CPF_Const)
unsigned long                  bClearAnimSetLinkupCachesOnLoadMap : 1;       // 0x0A54
(0x0004) [0x0000000000004000] [0x00000040] (CPF_Config)
unsigned long                  bEnableSecondaryDisplay : 1;             // 0x0A54 (0x0004)
[0x0000000000004000] [0x00000080] (CPF_Config)
unsigned long                  bEnableSecondaryViewport : 1;             // 0x0A54 (0x0004)
[0x0000000000004000] [0x00000100] (CPF_Config)
class UOnlineSubsystem*               OnlineSubsystem;                  // 0x0A58 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class UDownloadableContentEnumerator*        DLCEnumerator;                    // 0x0A60
(0x0008) [0x0000000000002002] (CPF_Const | CPF_Transient)
class FString                  DownloadableContentEnumeratorClassName;       // 0x0A68
```

```
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UDownloadableContentManager*             DLCManager;                                    // 0x0A78
(0x0008) [0x0000000000002002] (CPF_Const | CPF_Transient)
class FString                                  DownloadableContentManagerClassName;           // 0x0A80
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<struct FName>                           LevelsToLoadForPendingMapChange;               // 0x0A90
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class ULevel*>                          LoadedLevelsForPendingMapChange;               // 0x0AA0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class FString                                  PendingMapChangeFailureDescription;            // 0x0AB0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
float                                          MaxDeltaTime;                                  // 0x0AC0 (0x0004)
[0x0000000000004000] (CPF_Config)
class FString                                  SecondaryViewportClientClassName;              // 0x0AC8
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<class UScriptViewportClient*>           SecondaryViewportClients;                      // 0x0AD8
(0x0010) [0x0000000000500000] (CPF_NeedCtorLink)
TArray<struct FPointer>                        SecondaryViewportFrames;                       // 0x0AE8
(0x0010) [0x0000000000500000] (CPF_NeedCtorLink)
TArray<struct FLevelStreamingStatus>           PendingLevelStreamingStatusUpdates;            //
0x0AF8 (0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class UObjectReferencer*>               ObjectReferencers;                             // 0x0B08
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FFullyLoadedPackagesInfo>        PackagesToFullyLoad;                           // 0x0B18
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FNamedNetDriver>                 NamedNetDrivers;                               // 0x0B28
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<struct FAnimTag>                        AnimTags;                                      // 0x0B38 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameEngine");
}

return uClassPointer;
};

static bool HasSecondaryScreenActive();
static class UDownloadableContentManager* GetDLCManager();
static class UDownloadableContentEnumerator* GetDLCEnumerator();
static class UOnlineSubsystem* GetOnlineSubsystem();
void DestroyNamedNetDriver(struct FName NetDriverName);
bool CreateNamedNetDriver(struct FName NetDriverName);
};

// Class Engine.EngineBaseTypes
// 0x0000 (0x0060 - 0x0060)
class UEngineBaseTypes : public UObject
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EngineBaseTypes");
}

return uClassPointer;
};

};

// Class Engine.ISetParameter
// 0x0000 (0x0060 - 0x0060)
class UISetParameter : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ISetParameter");
}

return uClassPointer;
};

void SetActorParameter(struct FName Key, class AActor* Value);
void SetLinearColorParameter(struct FName Key, struct FLinearColor Value);
void SetVectorParameter(struct FName Key, struct FVector Value);
void SetFloatParameter(struct FName Key, float Value);
void SetNameParameter(struct FName Key, struct FName Value);
};

// Class Engine._Types_Engine
// 0x0000 (0x0060 - 0x0060)
class U_Types_Engine : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine._Types_Engine");
}

return uClassPointer;
};

};

// Class Engine.Brush
// 0x0030 (0x0268 - 0x0298)
class ABrush : public AActor
{
public:
uint8_t                                 CsgOper;                                // 0x0268 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FColor                           BrushColor;                             // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 PolyFlags;                              // 0x0270 (0x0004)
[0x0000000000000000]
unsigned long                           bColored : 1;                           // 0x0274 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bSolidWhenSelected : 1;                 // 0x0274 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                           bPlaceableFromClassBrowser : 1;         // 0x0274
(0x0004) [0x0000000000000000] [0x00000004]
class UModel*                           Brush;                                  // 0x0278 (0x0008)
[0x000000000000000A] (CPF_Const | CPF_ExportObject)
class UBrushComponent*                  BrushComponent;                         // 0x0280
(0x0008) [0x00000000040A000A] (CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
TArray<struct FGeomSelection>           SavedSelections;                        // 0x0288
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Brush");
}

return uClassPointer;
};

};

// Class Engine.BrushShape
// 0x0000 (0x0298 - 0x0298)
class ABrushShape : public ABrush
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.BrushShape");
}

return uClassPointer;
};

};

// Class Engine.Volume
// 0x000C (0x0298 - 0x02A4)
class AVolume : public ABrush
{
public:
class AActor*                        AssociatedActor;                    // 0x0298 (0x0008)
[0x0000000000000000]
unsigned long                        bForcePawnWalk : 1;                    // 0x02A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bProcessAllActors : 1;                 // 0x02A0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                        bPawnsOnly : 1;                     // 0x02A0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Volume");
}

return uClassPointer;
};

void eventProcessActorSetVolume(class AActor* Other);
void eventCollisionChanged();
void OnToggle(class USeqAct_Toggle* Action);
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void eventPostBeginPlay();
bool EncompassesPoint(struct FVector Loc);
bool Encompasses(class AActor* Other);
};
```

```
// Class Engine.BlockingVolume
// 0x0008 (0x02A4 - 0x02AC)
class ABlockingVolume : public AVolume
{
public:
unsigned long                    bBlockCamera : 1;                // 0x02A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.BlockingVolume");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
};

// Class Engine.DynamicBlockingVolume
// 0x0008 (0x02AC - 0x02B4)
class ADynamicBlockingVolume : public ABlockingVolume
{
public:
unsigned long                    bEnabled : 1;                 // 0x02B0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DynamicBlockingVolume");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct ADynamicBlockingVolume_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ADynamicBlockingVolume_FCheckpointRecord& Record);
void eventPostBeginPlay();
};

// Class Engine.CullDistanceVolume
// 0x0018 (0x02A4 - 0x02BC)
class ACullDistanceVolume : public AVolume
{
```

```
public:
TArray<struct FCullDistanceSizePair>            CullDistances;                          // 0x02A8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                                   bEnabled : 1;                           // 0x02B8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CullDistanceVolume");
}

return uClassPointer;
};

};

// Class Engine.LevelGridVolume
// 0x00AC (0x02A4 - 0x0350)
class ALevelGridVolume : public AVolume
{
public:
class FString                                   LevelGridVolumeName;                    // 0x02A8 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
uint8_t                                         CellShape;                              // 0x02B8 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                                         Subdivisions[0x3];                      // 0x02BC (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                                           LoadingDistance;                        // 0x02C8 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                                           KeepLoadedRange;                        // 0x02CC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FKConvexElem                             CellConvexElem;                         // 0x02D0 (0x0080)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelGridVolume");
}

return uClassPointer;
};

};
```

```
// Class Engine.LevelStreamingVolume
// 0x0020 (0x02A4 - 0x02C4)
class ALevelStreamingVolume : public AVolume
{
public:
TArray<class ULevelStreaming*>                  StreamingLevels;                            // 0x02A8
(0x0010) [0x0000000001420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)
unsigned long                                   bEditorPreVisOnly : 1;                      // 0x02B8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                   bDisabled : 1;                              // 0x02B8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                   bTestDistanceToVolume : 1;                  // 0x02B8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
uint8_t                                         StreamingUsage;                             // 0x02BC (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                         Usage;                                      // 0x02BD (0x0001)
[0x0000000020000000]  CPF_Deprecated)
float                                           TestVolumeDistance;                         // 0x02C0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelStreamingVolume");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct ALevelStreamingVolume_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ALevelStreamingVolume_FCheckpointRecord& Record);
void OnToggle(class USeqAct_Toggle* Action);
};

// Class Engine.LightmassCharacterIndirectDetailVolume
// 0x0004 (0x02A4 - 0x02A8)
class ALightmassCharacterIndirectDetailVolume : public AVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightmassCharacterIndirectDetailVolume");
}
return uClassPointer;
};

};

// Class Engine.LightmassImportanceVolume
// 0x0004 (0x02A4 - 0x02A8)
class ALightmassImportanceVolume : public AVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightmassImportanceVolume");
}

return uClassPointer;
};

};

// Class Engine.MassiveLODOverrideVolume
// 0x0004 (0x02A4 - 0x02A8)
class AMassiveLODOverrideVolume : public AVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MassiveLODOverrideVolume");
}

return uClassPointer;
};

};

// Class Engine.PathBlockingVolume
// 0x0004 (0x02A4 - 0x02A8)
class APathBlockingVolume : public AVolume
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PathBlockingVolume");
}

return uClassPointer;
};

};

// Class Engine.PhysicsVolume
// 0x0034 (0x02A4 - 0x02D8)
class APhysicsVolume : public AVolume
{
public:
struct FVector                    ZoneVelocity;                    // 0x02A8 (0x000C)
[0x0000000200000001] (CPF_Edit)
unsigned long                     bVelocityAffectsWalking : 1;          // 0x02B4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                     bDestructive : 1;                // 0x02B4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                     bMoveProjectiles : 1;              // 0x02B4 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                     bBounceVelocity : 1;               // 0x02B4 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                     bCrowdAgentsPlayDeathAnim : 1;          // 0x02B4
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                     bPhysicsOnContact : 1;              // 0x02B4 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
float                        GroundFriction;                  // 0x02B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        TerminalVelocity;                 // 0x02BC (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                       Priority;                     // 0x02C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        FluidFriction;                  // 0x02C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        RigidBodyDamping;                 // 0x02C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        MaxDampingForce;                  // 0x02CC (0x0004)
[0x0000000000000001] (CPF_Edit)
class APhysicsVolume*                 NextPhysicsVolume;                // 0x02D0 (0x0008)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PhysicsVolume");
}

return uClassPointer;
};

void NotifyPawnBecameViewTarget(class APawn* P, class APlayerController* PC);
void ModifyPlayer(class APawn* PlayerPawn);
void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
void eventCollisionChanged();
void OnToggle(class USeqAct_Toggle* inAction);
void eventPawnLeavingVolume(class APawn* Other);
void eventPawnEnteredVolume(class APawn* Other);
void eventActorLeavingVolume(class AActor* Other);
void eventActorEnteredVolume(class AActor* Other);
void eventPhysicsChangedFor(class AActor* Other);
struct FVector GetZoneVelocityForActor(class AActor* TheActor);
float GetGravityZ();
};

// Class Engine.DefaultPhysicsVolume
// 0x0000 (0x02D8 - 0x02D8)
class ADefaultPhysicsVolume : public APhysicsVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DefaultPhysicsVolume");
}

return uClassPointer;
};

void eventDestroyed();
};

// Class Engine.GravityVolume
// 0x0004 (0x02D8 - 0x02DC)
class AGravityVolume : public APhysicsVolume
{
public:
float                                   GravityZ;                                 // 0x02D8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GravityVolume");
}

return uClassPointer;
};

};

// Class Engine.PortalVolume
// 0x0014 (0x02A4 - 0x02B8)
class APortalVolume : public AVolume
{
public:
TArray<class APortalTeleporter*>            Portals;                         // 0x02A8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PortalVolume");
}

return uClassPointer;
};

};

// Class Engine.PostProcessVolume
// 0x017C (0x02A4 - 0x0420)
class APostProcessVolume : public AVolume
{
public:
float                                 Priority;                        // 0x02A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                         bOverrideWorldPostProcessChain : 1;        // 0x02AC
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bEnabled : 1;                        // 0x02AC (0x0004)
[0x0000000000000021] [0x00000002] (CPF_Edit | CPF_Net)
struct FPostProcessSettings                Settings;                        // 0x02B0 (0x0168)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class APostProcessVolume*                 NextLowerPriorityVolume;                // 0x0418
(0x0008) [0x0000000001002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PostProcessVolume");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
};

// Class Engine.PrecomputedVisibilityOverrideVolume
// 0x0024 (0x02A4 - 0x02C8)
class APrecomputedVisibilityOverrideVolume : public AVolume
{
public:
TArray<class AActor*>                OverrideVisibleActors;              // 0x02A8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class AActor*>                OverrideInvisibleActors;            // 0x02B8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrecomputedVisibilityOverrideVolume");
}

return uClassPointer;
};

};

// Class Engine.PrecomputedVisibilityVolume
// 0x0004 (0x02A4 - 0x02A8)
class APrecomputedVisibilityVolume : public AVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrecomputedVisibilityVolume");
}
```

```
return uClassPointer;
};

};

// Class Engine.ReverbVolume
// 0x004C (0x02A4 - 0x02F0)
class AReverbVolume : public AVolume
{
public:
float                                   Priority;                                  // 0x02A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bEnabled : 1;                              // 0x02AC (0x0004)
[0x0000000000000021] [0x00000001] (CPF_Edit | CPF_Net)
struct FReverbSettings                  Settings;                                  // 0x02B0 (0x0010)
[0x0000000000000001] (CPF_Edit)
struct FInteriorSettings                AmbientZoneSettings;                       // 0x02C0 (0x0024)
[0x0000000000000001] (CPF_Edit)
class AReverbVolume*                    NextLowerPriorityVolume;                   // 0x02E8
(0x0008) [0x0000000001002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ReverbVolume");
}

return uClassPointer;
};

};

// Class Engine.TriggerVolume
// 0x0004 (0x02A4 - 0x02A8)
class ATriggerVolume : public AVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TriggerVolume");
}

return uClassPointer;
```

```
};

};

// Class Engine.DynamicSMActor
// 0x0060 (0x0268 - 0x02C8)
class ADynamicSMActor : public AActor
{
public:
class UStaticMeshComponent*                StaticMeshComponent;                    // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
class UDynamicLightEnvironmentComponent*         LightEnvironment;                       //
0x0270 (0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject |
CPF_EditConst | CPF_Component | CPF_EditInline)
class UStaticMesh*                               ReplicatedMesh;                         // 0x0278 (0x0008)
[0x0000000100002020] (CPF_Net | CPF_Transient)
class UMaterialInterface*                         ReplicatedMaterial0;                   // 0x0280 (0x0008)
[0x0000000100000020] (CPF_Net)
class UMaterialInterface*                         ReplicatedMaterial1;                   // 0x0288 (0x0008)
[0x0000000100000020] (CPF_Net)
class UMaterialInterface*                         ReplicatedMaterial2;                   // 0x0290 (0x0008)
[0x0000000100000020] (CPF_Net)
class UMaterialInterface*                         ReplicatedMaterial3;                   // 0x0298 (0x0008)
[0x0000000100000020] (CPF_Net)
unsigned long                                   bForceStaticDecals : 1;                 // 0x02A0 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)
unsigned long                                   bPawnCanBaseOn : 1;                      // 0x02A0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                   bSafeBaseIfAsleep : 1;                  // 0x02A0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FVector                                  ReplicatedMeshTranslation;               // 0x02A4 (0x000C)
[0x0000000100000020] (CPF_Net)
struct FRotator                                 ReplicatedMeshRotation;                  // 0x02B0 (0x000C)
[0x0000000100000020] (CPF_Net)
struct FVector                                  ReplicatedMeshScale3D;                   // 0x02BC (0x000C)
[0x0000000100000020] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DynamicSMActor");
}

return uClassPointer;
};

void SetLightEnvironmentToNotBeDynamic();
void eventDetach(class AActor* Other);
void eventAttach(class AActor* Other);
```

```
bool CanBasePawn(class APawn* P);
void SetStaticMesh(class UStaticMesh* NewMesh, struct FVector NewTranslation, struct
FRotator NewRotation, struct FVector NewScale3D);
void OnSetMaterial(class USeqAct_SetMaterial* Action);
void OnSetMesh(class USeqAct_SetMesh* Action);
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.InterpActor
// 0x0050 (0x02C8 - 0x0318)
class AInterpActor : public ADynamicSMActor
{
public:
unsigned long                    bShouldSaveForCheckpoint : 1;             // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                    bMonitorMover : 1;                        // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                    bMonitorZVelocity : 1;                    // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                    bContinueOnEncroachPhysicsObject : 1;         // 0x02C8
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bStopOnEncroach : 1;                      // 0x02C8 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                    bShouldShadowParentAllAttachedActors : 1;     // 0x02C8
(0x0004) [0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                    bIsLift : 1;                              // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000040]
class ANavigationPoint*                MyMarker;                           // 0x02D0 (0x0008)
[0x0000000000000000]
float                       MaxZVelocity;                                  // 0x02D8 (0x0004)
[0x0000000000000000]
float                       StayOpenTime;                                  // 0x02DC (0x0004)
[0x0000000000000000]
class USoundCue*                    OpenSound;                             // 0x02E0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                    OpeningAmbientSound;                   // 0x02E8 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                    OpenedSound;                           // 0x02F0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                    CloseSound;                            // 0x02F8 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                    ClosingAmbientSound;                   // 0x0300 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                    ClosedSound;                           // 0x0308 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UAudioComponent*                 AmbientSoundComponent;                  // 0x0310
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpActor");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct AInterpActor_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct AInterpActor_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
void ShutDown();
void eventInterpolationChanged(class USeqAct_Interp* InterpAction);
void eventForceNetRelevant();
void eventInterpolationFinished(class USeqAct_Interp* InterpAction);
void eventInterpolationStarted(class USeqAct_Interp* InterpAction, class UInterpGroupInst*
GroupInst);
void PlayMovingSound(unsigned long bClosing);
void FinishedOpen();
void Restart();
void eventDetach(class AActor* Other);
void eventAttach(class AActor* Other);
void eventRanInto(class AActor* Other);
bool eventEncroachingOn(class AActor* Other);
void eventPostBeginPlay();
};

// Class Engine.Emitter
// 0x0014 (0x0268 - 0x027C)
class AEmitter : public AActor
{
public:
class UParticleSystemComponent*          ParticleSystemComponent;           // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
class UDynamicLightEnvironmentComponent*      LightEnvironment;              //
0x0270 (0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject |
CPF_EditConst | CPF_Component | CPF_EditInline)
unsigned long                    bDestroyOnSystemFinish : 1;          // 0x0278 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                    bPostUpdateTickGroup : 1;            // 0x0278 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bCurrentlyActive : 1;              // 0x0278 (0x0004)
[0x0000000100000020] [0x00000004] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Emitter");
}
```

```
return uClassPointer;
};

void HideSelf();
void ApplyCheckpointRecord(struct AEmitter_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct AEmitter_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
void OnSetParticleSysParam(class USeqAct_SetParticleSysParam* Action);
void SetActorParameter(struct FName ParameterName, class AActor* Param);
void SetExtColorParameter(struct FName ParameterName, uint8_t Red, uint8_t Green, uint8_t Blue, uint8_t Alpha);
void SetColorParameter(struct FName ParameterName, struct FColor Param);
void SetVectorParameter(struct FName ParameterName, struct FVector Param);
void SetFloatParameter(struct FName ParameterName, float Param);
void ShutDown();
void OnParticleEventGenerator(class USeqAct_ParticleEventGenerator* Action);
void OnToggle(class USeqAct_Toggle* Action);
void OnParticleSystemFinished(class UParticleSystemComponent* FinishedComponent);
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
void eventSetTemplate(class UParticleSystem* NewTemplate, unsigned long bDestroyOnFinish);
};

// Class Engine.StaticEmitterCollectionActor
// 0x0014 (0x027C - 0x0290)
class AStaticEmitterCollectionActor : public AEmitter
{
public:
TArray<class UParticleSystemComponent*>          ParticleSystemComponents;                    // 0x0280 (0x0010) [0x000000000448000A] (CPF_Const | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticEmitterCollectionActor");
}

return uClassPointer;
};

};

// Class Engine.EmitterPool
// 0x0070 (0x0268 - 0x02D8)
class AEmitterPool : public AActor
{
public:
class UParticleSystemComponent*                  PSCTemplate;                            // 0x0268
```

```
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
TArray<class UParticleSystemComponent*>          PoolComponents;                         // 0x0270
(0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UParticleSystemComponent*>          ActiveComponents;                        // 0x0280
(0x0010) [0x0000000004482008] (CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
int32_t                              MaxActiveEffects;                      // 0x0290 (0x0004)
[0x0000000000000000]
unsigned long                          bLogPoolOverflow : 1;                   // 0x0294 (0x0004)
[0x0000000000044000] [0x00000001] (CPF_Config | CPF_GlobalConfig)
unsigned long                          bLogPoolOverflowList : 1;                // 0x0294 (0x0004)
[0x0000000000044000] [0x00000002] (CPF_Config | CPF_GlobalConfig)
TArray<struct FEmitterBaseInfo>                RelativePSCs;                          // 0x0298 (0x0010)
[0x0000000000482000] (CPF_Transient | CPF_Component | CPF_NeedCtorLink)
float                                SMC_MIC_ReductionTime;                  // 0x02A8 (0x0004)
[0x0000000000000000]
float                                SMC_MIC_CurrentReductionTime;             // 0x02AC (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                              IdealStaticMeshComponents;               // 0x02B0 (0x0004)
[0x0000000000000000]
int32_t                              IdealMaterialInstanceConstants;            // 0x02B4 (0x0004)
[0x0000000000000000]
TArray<class UStaticMeshComponent*>            FreeSMComponents;                        // 0x02B8
(0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UMaterialInstanceConstant*>         FreeMatInstConsts;                     // 0x02C8
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EmitterPool");
}

return uClassPointer;
};

class UParticleSystemComponent* SpawnEmitterCustomLifetime(class UParticleSystem*
EmitterTemplate, unsigned long bSkipAutoActivate);
class UParticleSystemComponent* SpawnEmitterMeshAttachment(class UParticleSystem*
EmitterTemplate, class USkeletalMeshComponent* Mesh, struct FName AttachPointName,
unsigned long bAttachToSocket, struct FVector RelativeLoc, struct FRotator RelativeRot);
class UParticleSystemComponent* SpawnEmitter(class UParticleSystem* EmitterTemplate,
struct FVector SpawnLocation, struct FRotator SpawnRotation, class AActor* AttachToActor,
class AActor* InInstigator, int32_t MaxDLEPooledReuses, unsigned long bInheritScaleFromBase);
class UParticleSystemComponent* GetPooledComponent(class UParticleSystem*
EmitterTemplate, unsigned long bAutoActivate);
class UMaterialInstanceConstant* GetFreeMatInstConsts(unsigned long bCreateNewObject);
void FreeMaterialInstanceConstants(class UStaticMeshComponent* SMC);
```

```
class UStaticMeshComponent* GetFreeStaticMeshComponent(unsigned long
bCreateNewObject);
void FreeStaticMeshComponents(class UParticleSystemComponent* PSC);
void ReturnToPool(class UParticleSystemComponent* PSC);
void ClearPoolComponents(unsigned long bClearActive);
void OnParticleSystemFinished(class UParticleSystemComponent* PSC);
};

// Class Engine.HUD
// 0x00A0 (0x0268 - 0x0308)
class AHUD : public AActor
{
public:
struct FColor                    WhiteColor;                   // 0x0268 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FColor                    GreenColor;                   // 0x026C (0x0004)
[0x0000000000000002] (CPF_Const)
struct FColor                    RedColor;                     // 0x0270 (0x0004)
[0x0000000000000002] (CPF_Const)
class APlayerController*              PlayerOwner;                  // 0x0278 (0x0008)
[0x0000000000000000]
unsigned long                     bLostFocusPaused : 1;              // 0x0280 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                     bShowHUD : 1;                  // 0x0280 (0x0004)
[0x0000000000004000] [0x00000002] (CPF_Config)
unsigned long                     bShowScores : 1;                // 0x0280 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                     bShowDebugInfo : 1;               // 0x0280 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                     bShowBadConnectionAlert : 1;            // 0x0280 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                     bShowDirectorInfoDebug : 1;            // 0x0280 (0x0004)
[0x0000000000004000] [0x00000020] (CPF_Config)
unsigned long                     bShowDirectorInfoHUD : 1;             // 0x0280 (0x0004)
[0x0000000000004000] [0x00000040] (CPF_Config)
unsigned long                     bRenderFullScreen : 1;              // 0x0280 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                     bScaleCanvasForCinematicMode : 1;          // 0x0280
(0x0004) [0x0000000000000000] [0x00000100]
unsigned long                     bShowOverlays : 1;                // 0x0280 (0x0004)
[0x0000000000000000] [0x00000200]
float                         HudCanvasScale;                 // 0x0284 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
TArray<class AActor*>                  PostRenderedActors;              // 0x0288 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FConsoleMessage>               ConsoleMessages;                 // 0x0298
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
struct FColor                    ConsoleColor;                  // 0x02A8 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                        ConsoleMessageCount;               // 0x02AC (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                        ConsoleFontSize;                 // 0x02B0 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                         ConsoleMessagePosX;                // 0x02B4 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                           ConsoleMessagePosY;                  // 0x02B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UCanvas*                        Canvas;                        // 0x02C0 (0x0008)
[0x0000000000000000]
float                           LastHUDRenderTime;                   // 0x02C8 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           RenderDelta;                         // 0x02CC (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           SizeX;                          // 0x02D0 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           SizeY;                          // 0x02D4 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           CenterX;                         // 0x02D8 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           CenterY;                         // 0x02DC (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           RatioX;                          // 0x02E0 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           RatioY;                          // 0x02E4 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FName>                 DebugDisplay;                   // 0x02E8 (0x0010)
[0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
TArray<struct FKismetDrawTextInfo>         KismetTextInfo;                   // 0x02F8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HUD");
}

return uClassPointer;
};

void eventOnLostFocusPause(unsigned long bEnable);
void NotifyBindPostProcessEffects();
static class UFont* GetFontSizeIndex(int32_t FontSize);
void DrawTextW(class FString Text, struct FVector2D Position, class UFont* TextFont, struct
FVector2D FontScale, struct FColor TextColor, struct FFontRenderInfo& RenderInfo);
void DisplayKismetMessages();
void AddConsoleMessage(class FString M, class APlayerReplicationInfo* PRI, float LifeTime);
bool ShouldShowConsoleMessage(struct FConsoleMessage InConsoleMessage);
void DisplayConsoleMessages();
void Message(class APlayerReplicationInfo* PRI, class FString msg, struct FName MsgType,
float LifeTime);
void DisplayBadConnectionAlert();
void DrawHUD();
void eventPostRender();
void PreCalcValues();
void DrawRoute(class APawn* Target);
void ShowDebugInfo(float& out_YL, float& out_YPos);
void ToggleDirectorInfoDebug();
void ToggleDirectorInfoHUD();
bool ShouldDisplayDebug(struct FName DebugType);
void ShowDebug(struct FName DebugType);
void SetShowScores(unsigned long bNewValue);
void ShowScores();
void ShowHUD();
void ToggleHUD();
void AddPostRenderedActor(class AActor* A);
void RemovePostRenderedActor(class AActor* A);
void DrawActorOverlays(struct FVector ViewPoint, struct FRotator ViewRotation);
void Init();
void eventPostBeginPlay();
void Draw2DLine(int32_t X1, int32_t Y1, int32_t X2, int32_t Y2, struct FColor LineColor);
void Draw3DLine(struct FVector Start, struct FVector End, struct FColor LineColor);
};

// Class Engine.AutoTestManager
// 0x00D0 (0x0268 - 0x0338)
class AAutoTestManager : public AInfo
{
public:
unsigned long                  bAutomatedPerfTesting : 1;               // 0x0268 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                  bAutoContinueToNextRound : 1;            // 0x0268 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                  bUsingAutomatedTestingMapList : 1;       // 0x0268
(0x0004) [0x0000000000000000] [0x00000004]
unsigned long                  bAutomatedTestingWithOpen : 1;           // 0x0268
(0x0004) [0x0000000000000000] [0x00000008]
unsigned long                  bExitOnCyclesComplete : 1;               // 0x0268 (0x0004)
[0x0000000000044000] [0x00000010] (CPF_Config | CPF_GlobalConfig)
unsigned long                  bCheckingForFragmentation : 1;           // 0x0268 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                  bCheckingForMemLeaks : 1;                // 0x0268 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                  bDoingASentinelRun : 1;                  // 0x0268 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                  bSentinelStreamingLevelStillLoading : 1;     // 0x0268
(0x0004) [0x0000000000002000] [0x00000100] (CPF_Transient)
int32_t                   AutomatedPerfRemainingTime;                  // 0x026C (0x0004)
[0x0000000000000000]
int32_t                   AutomatedTestingMapIndex;                    // 0x0270 (0x0004)
[0x0000000000000000]
TArray<class FString>              AutomatedMapTestingList;               // 0x0278
(0x0010) [0x0000000000444000] (CPF_Config | CPF_GlobalConfig | CPF_NeedCtorLink)
int32_t                   NumAutomatedMapTestingCycles;                // 0x0288 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                   NumberOfMatchesPlayed;                       // 0x028C (0x0004)
[0x0000000000000000]
int32_t                   NumMapListCyclesDone;                        // 0x0290 (0x0004)
[0x0000000000000000]
```

```
class FString                          AutomatedTestingExecCommandToRunAtStartMatch;  //
0x0298 (0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class FString                          AutomatedMapTestingTransitionMap;              // 0x02A8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class FString                          SentinelTaskDescription;                       // 0x02B8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                          SentinelTaskParameter;                         // 0x02C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                          SentinelTagDesc;                               // 0x02D8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class APlayerController*                   SentinelPC;                                // 0x02E8 (0x0008)
[0x0000000000002000] (CPF_Transient)
TArray<struct FVector>                     SentinelTravelArray;                       // 0x02F0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                                SentinelNavigationIdx;                         // 0x0300 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                SentinelIdx;                                   // 0x0304 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                NumRotationsIncrement;                         // 0x0308 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                TravelPointsIncrement;                         // 0x030C (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                NumMinutesPerMap;                              // 0x0310 (0x0004)
[0x0000000000004000] (CPF_Config)
TArray<class FString>                      CommandsToRunAtEachTravelTheWorldNode;     //
0x0318 (0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                          CommandStringToExec;                           // 0x0328 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AutoTestManager");
}

return uClassPointer;
};

float CalcTravelTheWorldTime(int32_t NumTravelLocations, int32_t NumRotations);
void PrintOutTravelWorldTimes(int32_t TotalTimeInSeconds);
void SetIncrementsForLoops(float NumTravelLocations);
bool CheckForSentinelRun();
void StartMatch();
class FString GetNextAutomatedTestingMap();
void IncrementNumberOfMatchesPlayed();
void IncrementAutomatedTestingMapIndex();
void CloseAutomatedMapTestTimer();
void StartAutomatedMapTestTimerWorker();
void eventStartAutomatedMapTestTimer();
void DoMemoryTracking();
void DoTimeBasedSentinelStatGathering();
void DoSentinel_ViewDependentMemoryAtSpecificLocation(struct FVector& InLocation, struct
FRotator& InRotation);
void DoSentinel_PerfAtSpecificLocation(struct FVector& InLocation, struct FRotator& InRotation);
void DoSentinel_MemoryAtSpecificLocation(struct FVector InLocation, struct FRotator
InRotation);
void GetTravelLocations(struct FName LevelName, class APlayerController* PC, TArray<struct
FVector>& TravelPoints);
void HandlePerLoadedMapAudioStats();
void DoSentinelActionPerLoadedMap();
void DoTravelTheWorld();
void EndSentinelRun(uint8_t RunResult);
void AddSentinelPerTimePeriodStats(struct FVector InLocation, struct FRotator InRotation);
void BeginSentinelRun(class FString TaskDescription, class FString TaskParameter, class FString
TagDesc);
void InitializeOptions(class FString Options);
void eventTimer();
void eventPostBeginPlay();
};

// Class Engine.CoverGroup
// 0x0018 (0x0268 - 0x0280)
class ACoverGroup : public AInfo
{
public:
TArray<struct FActorReference>          CoverLinkRefs;                    // 0x0268 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                   AutoSelectRadius;                 // 0x0278 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   AutoSelectHeight;                 // 0x027C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CoverGroup");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
void ToggleGroup();
void DisableGroup();
void EnableGroup();
};

// Class Engine.FileWriter
// 0x0020 (0x0268 - 0x0288)
class AFileWriter : public AInfo
```

```
{
public:
struct FPointer                         ArchivePtr;                             // 0x0268 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
class FString                           Filename;                               // 0x0270 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
uint8_t                                 FileType;                               // 0x0280 (0x0001)
[0x0000000000000002] (CPF_Const)
unsigned long                           bFlushEachWrite : 1;                    // 0x0284 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                           bWantsAsyncWrites : 1;                  // 0x0284 (0x0004)
[0x0000000000000000] [0x00000002]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FileWriter");
}

return uClassPointer;
};

void eventDestroyed();
void Logf(class FString logString);
void CloseFile();
bool OpenFile(class FString InFilename, uint8_t InFileType, class FString InExtension, unsigned
long bUnique, unsigned long bIncludeTimeStamp);
};

// Class Engine.FileLog
// 0x0000 (0x0288 - 0x0288)
class AFileLog : public AFileWriter
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FileLog");
}

return uClassPointer;
};

void CloseLog();
void OpenLog(class FString LogFilename, class FString extension, unsigned long bUnique);
```

```
};

// Class Engine.GameInfo
// 0x0208 (0x0268 - 0x0470)
class AGameInfo : public AInfo
{
public:
class UGroupComponent_ORS*                    RegistryGroup;                          // 0x0268
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
int32_t                          BulletScenesCount;                      // 0x0270 (0x0004)
[0x0000000000000000]
unsigned long                          bRestartLevel : 1;                      // 0x0274 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                          bPauseable : 1;                         // 0x0274 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                          bTeamGame : 1;                          // 0x0274 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                          bGameEnded : 1;                         // 0x0274 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                          bOverTime : 1;                          // 0x0274 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                          bDelayedStart : 1;                      // 0x0274 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                          bWaitingToStartMatch : 1;               // 0x0274 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                          bChangeLevels : 1;                      // 0x0274 (0x0004)
[0x0000000000044000] [0x00000080] (CPF_Config | CPF_GlobalConfig)
unsigned long                          bAlreadyChanged : 1;                    // 0x0274 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                          bGameRestarted : 1;                     // 0x0274 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                          bLevelChange : 1;                       // 0x0274 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                          bKickLiveIdlers : 1;                    // 0x0274 (0x0004)
[0x0000000000044000] [0x00000800] (CPF_Config | CPF_GlobalConfig)
unsigned long                          bFixedPlayerStart : 1;                  // 0x0274 (0x0004)
[0x0000000000000000] [0x00001000]
unsigned long                          bDoFearCostFallOff : 1;                 // 0x0274 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                          bUseSeamlessTravel : 1;                 // 0x0274 (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                          bHasNetworkError : 1;                   // 0x0274 (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                          bRequiresPushToTalk : 1;                // 0x0274 (0x0004)
[0x0000000000000002] [0x00010000] (CPF_Const)
unsigned long                          bIsStandbyCheckingEnabled : 1;          // 0x0274 (0x0004)
[0x0000000000004000] [0x00020000] (CPF_Config)
unsigned long                          bIsStandbyCheckingOn : 1;               // 0x0274 (0x0004)
[0x0000000000000000] [0x00040000]
unsigned long                          bHasStandbyCheatTriggered : 1;          // 0x0274 (0x0004)
[0x0000000000000000] [0x00080000]
unsigned long                          bKeepingLoadingMovieOpen : 1;           // 0x0274 (0x0004)
[0x0000000000000000] [0x00100000]
```

```
class FString                    CausEventCommand;                    // 0x0278 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    BugLocString;                        // 0x0288 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    BugRotString;                        // 0x0298 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                         GameDifficulty;                      // 0x02A8 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                         GameSpeed;                           // 0x02AC (0x0004)
[0x0000000000000000]
class UClass*                     DefaultPawnClass;                     // 0x02B0 (0x0008)
[0x0000000000000000]
class UClass*                     HUDType;                              // 0x02B8 (0x0008)
[0x0000000000000000]
class UClass*                     SecondaryHUDType;                     // 0x02C0 (0x0008)
[0x0000000000000000]
int32_t                        MaxSpectators;                        // 0x02C8 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                        MaxSpectatorsAllowed;                 // 0x02CC (0x0004)
[0x0000000000000000]
int32_t                        NumSpectators;                        // 0x02D0 (0x0004)
[0x0000000000000000]
int32_t                        MaxPlayers;                          // 0x02D4 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                        MaxPlayersAllowed;                    // 0x02D8 (0x0004)
[0x0000000000000000]
int32_t                        NumPlayers;                          // 0x02DC (0x0004)
[0x0000000000000000]
int32_t                        NumBots;                             // 0x02E0 (0x0004)
[0x0000000000000000]
int32_t                        NumTravellingPlayers;                 // 0x02E4 (0x0004)
[0x0000000000000000]
int32_t                        CurrentID;                           // 0x02E8 (0x0004)
[0x0000000000000000]
class FString                    DefaultPlayerName;                    // 0x02F0 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    GameName;                             // 0x0300 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
float                         FearCostFallOff;                      // 0x0310 (0x0004)
[0x0000000000000000]
int32_t                        GoalScore;                           // 0x0314 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                        MaxLives;                            // 0x0318 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                        TimeLimit;                           // 0x031C (0x0004)
[0x0000000000004000] (CPF_Config)
class AMutator*                    BaseMutator;                         // 0x0320 (0x0008)
[0x0000000000000000]
class UClass*                     AutoTestManagerClass;                 // 0x0328 (0x0008)
[0x0000000000000000]
class AAutoTestManager*               MyAutoTestManager;                   // 0x0330
(0x0008) [0x0000000000000000]
class UClass*                     PlayerControllerClass;                // 0x0338 (0x0008)
[0x0000000000000000]
```

```
class UClass*                       PlayerReplicationInfoClass;              // 0x0340 (0x0008)
[0x0000000000000000]
class UClass*                       GameReplicationInfoClass;                // 0x0348 (0x0008)
[0x0000000000000001] (CPF_Edit)
class AGameReplicationInfo*               GameReplicationInfo;                  // 0x0350
(0x0008) [0x0000000000000000]
class ACrowdPopulationManagerBase*             PopulationManager;                      // 0x0358
(0x0008) [0x0000000000000000]
class UClass*                       PopulationManagerClass;                  // 0x0360 (0x0008)
[0x0000000000000000]
float                          MaxIdleTime;                       // 0x0368 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          MaxTimeMargin;                      // 0x036C (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          TimeMarginSlack;                    // 0x0370 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          MinTimeMargin;                      // 0x0374 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
TArray<class APlayerReplicationInfo*>           InactivePRIArray;                      // 0x0378
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FScriptDelegate>               Pausers;                             // 0x0388 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class UOnlineSubsystem*                    OnlineSub;                            // 0x0398 (0x0008)
[0x0000000000000000]
class UOnlineGameInterface*                 GameInterface_Object;                      // 0x03A0
(0x0008) [0x0000000000000000]
class UOnlineGameInterface*                 GameInterface_Interface;                   // 0x03A8
(0x0008) [0x0000000000000000]
class UClass*                       OnlineStatsWriteClass;                  // 0x03B0 (0x0008)
[0x0000000000000000]
class ACoverReplicator*                    CoverReplicatorBase;                       // 0x03B8 (0x0008)
[0x0000000000000000]
class UClass*                       OnlineGameSettingsClass;                 // 0x03C0 (0x0008)
[0x0000000000000002] (CPF_Const)
class FString                      ServerOptions;                       // 0x03C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                          AdjustedNetSpeed;                    // 0x03D8 (0x0004)
[0x0000000000000000]
float                          LastNetSpeedUpdateTime;                 // 0x03DC (0x0004)
[0x0000000000000000]
int32_t                          TotalNetBandwidth;                   // 0x03E0 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                          MinDynamicBandwidth;                 // 0x03E4 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                          MaxDynamicBandwidth;                 // 0x03E8 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                          StandbyRxCheatTime;                  // 0x03EC (0x0004)
[0x0000000000004000] (CPF_Config)
float                          StandbyTxCheatTime;                  // 0x03F0 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                          BadPingThreshold;                   // 0x03F4 (0x0004)
[0x0000000000004000] (CPF_Config)
float                          PercentMissingForRxStandby;              // 0x03F8 (0x0004)
[0x0000000000004000] (CPF_Config)
```

```
float                              PercentMissingForTxStandby;                // 0x03FC (0x0004)
[0x0000000000004000] (CPF_Config)
float                              PercentForBadPing;                         // 0x0400 (0x0004)
[0x0000000000004000] (CPF_Config)
float                              JoinInProgressStandbyWaitTime;             // 0x0404 (0x0004)
[0x0000000000004000] (CPF_Config)
class UMaterial*                          StreamingPauseIcon;                 // 0x0408 (0x0008)
[0x0000000000000000]
TArray<struct FGameClassShortName>             GameInfoClassAliases;                  // 0x0410
(0x0010) [0x0000000000404003] (CPF_Edit | CPF_Const | CPF_Config | CPF_NeedCtorLink)
class FString                             DefaultGameType;                     // 0x0420 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<struct FGameTypePrefix>                 DefaultMapPrefixes;                    // 0x0430
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<struct FGameTypePrefix>                 CustomMapPrefixes;                     // 0x0440
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
int32_t                              AnimTreePoolSize;                        // 0x0450 (0x0004)
[0x0000000000004000] (CPF_Config)
struct FScriptDelegate                      __CanUnpause__Delegate;                 // 0x0458
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameInfo");
}

return uClassPointer;
};

void eventClientMapLoadFail(struct FUniqueNetId PlayerID, class FString MapName);
void ClearOnlineDelegates();
void InitCrowdPopulationManager();
void eventOnEngineHasLoaded();
void OnDestroyOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
void eventStandbyCheatDetected(uint8_t StandbyType);
void EnableStandbyCheatDetection(unsigned long bIsEnabled);
void BeginBVT(class FString TagDesc);
bool CheckForSentinelRun();
bool ShouldAutoContinueToNextRound();
bool IsDoingASentinelRun();
bool IsCheckingForMemLeaks();
bool IsCheckingForFragmentation();
bool IsAutomatedPerfTesting();
void SetBandwidthLimit(float AsyncIOBandwidthLimit);
void DoTravelTheWorld();
void TellClientsToTravelToSession(struct FName SessionName, class UClass* SearchClass,
uint8_t PlatformSpecificInfo);
void TellClientsPartyHostIsLeaving(struct FUniqueNetId PartyHostPlayerId);
void TellClientsToReturnToPartyHost();
void OnServerCreateComplete(struct FName SessionName, unsigned long bWasSuccessful);
void RegisterServer();
void OnLoginChange(uint8_t LocalUserNum);
void OnLoginFailed(uint8_t LocalUserNum, uint8_t ErrorCode);
void ClearAutoLoginDelegates();
bool ProcessServerLogin();
void eventMatineeCancelled();
void RecalculateSkillRating();
void UpdateGameplayMuteList(class APlayerController* PC);
bool MatchIsInProgress();
void UpdateGameSettingsCounts();
void SetSeamlessTravelViewTarget(class APlayerController* PC);
void eventHandleSeamlessTravelPlayer(class AController*& C);
void UpdateGameSettings();
void eventPostSeamlessTravel();
void SwapPlayerControllers(class APlayerController* OldPC, class APlayerController* NewPC);
void eventGetSeamlessTravelActorList(unsigned long bToEntry, TArray<class AActor*>&
ActorList);
void OverridePRI(class APlayerController* PC, class APlayerReplicationInfo* OldPRI);
bool FindInactivePRI(class APlayerController* PC);
void AddInactivePRI(class APlayerReplicationInfo* PRI, class APlayerController* PC);
void eventPostCommitMapChange();
void eventPreCommitMapChange(class FString PreviousMapName, class FString
NextMapName);
bool AllowPausing(class APlayerController* PC);
bool AllowCheats(class APlayerController* P);
static bool AllowMutator(class FString MutatorClassName);
bool PlayerCanRestart(class APlayerController* aPlayer);
bool PlayerCanRestartGame(class APlayerController* aPlayer);
void ModifyScoreKill(class AController* Killer, class AController* Other);
void ScoreKill(class AController* Killer, class AController* Other);
bool CheckScore(class APlayerReplicationInfo* Scorer);
void ScoreObjective(class APlayerReplicationInfo* Scorer, int32_t Score);
void AddObjectiveScore(class APlayerReplicationInfo* Scorer, int32_t Score);
float RatePlayerStart(class APlayerStart* P, uint8_t Team, class AController* Player);
class APlayerStart* ChoosePlayerStart(class AController* Player, uint8_t InTeam);
class ANavigationPoint* FindPlayerStart(class AController* Player, uint8_t InTeam, class FString
IncomingName);
bool ShouldSpawnAtStartSpot(class AController* Player);
void EndLogging(class FString Reason);
void GameEventsPoll();
void EndOnlineGame();
void PerformEndGameHandling();
void EndGame(class APlayerReplicationInfo* Winner, class FString Reason);
void WriteOnlineStats();
bool CheckEndGame(class APlayerReplicationInfo* Winner, class FString Reason);
bool CheckModifiedEndGame(class APlayerReplicationInfo* Winner, class FString Reason);
void RestartGame();
bool GetTravelType();
class FString GetNextMap();
void SendPlayer(class APlayerController* aPlayer, class FString URL);
uint8_t PickTeam(uint8_t Current, class AController* C);
bool ChangeTeam(class AController* Other, int32_t N, unsigned long bNewTeam);
```

```
void ChangeName(class AController* Other, class FString S, unsigned long bNameChange);
bool CheckRelevance(class AActor* Other);
bool CanSpectate(class APlayerController* Viewer, class APlayerReplicationInfo* ViewTarget);
void SetPlayerDefaults(class APawn* PlayerPawn);
void Mutate(class FString MutateString, class APlayerController* Sender);
void UnregisterPlayer(class APlayerController* PC);
void Logout(class AController* Exiting);
void eventPreExit();
void eventPostLogin(class APlayerController* NewPlayer);
void UpdateBestNextHosts();
int32_t BestNextHostSort(class APlayerController* A, class APlayerController* B);
void GenericPlayerInitialization(class AController* C);
void ReplicateStreamingStatus(class APlayerController* PC);
class UClass* GetDefaultPlayerClass(class AController* C);
class APawn* SpawnDefaultPawnFor(class AController* NewPlayer, class ANavigationPoint*
StartSpot);
void RestartPlayer(class AController* NewPlayer);
void StartBots();
void StartHumans();
void OnStartOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
void StartOnlineGame();
void StartMatch();
class APlayerController* eventLogin(class FString Portal, class FString Options, struct
FUniqueNetId UniqueId, class FString& ErrorMessage);
class APlayerController* SpawnPlayerController(struct FVector SpawnLocation, struct FRotator
SpawnRotation);
int32_t GetNextPlayerID();
bool AtCapacity(unsigned long bSpectator);
static void RejectLogin(class UPlayer* InPlayer, class FString Error);
void eventOnRejectLogin(class UPlayer* InPlayer, class FString Error);
static void ResumeLogin(class UPlayer* InPlayer);
static class UPlayer* PauseLogin();
void eventPreLoginSplitscreen(struct FUniqueNetId PrimaryPlayerUniqueId, class FString
Options, class FString Address, struct FUniqueNetId UniqueId, unsigned long bSupportsAuth,
class FString& ErrorMessage);
void eventPreLogin(class FString Options, class FString Address, struct FUniqueNetId UniqueId,
unsigned long bSupportsAuth, class FString& ErrorMessage);
void SendNetworkReconnectMessages();
class APlayerController* ProcessClientTravel(struct FGuid NextMapGuid, unsigned long
bSeamless, unsigned long bAbsolute, class FString& URL);
void ProcessServerTravel(class FString URL, unsigned long bAbsolute);
void RemoveMutator(class AMutator* MutatorToRemove);
void AddMutator(class FString mutname, unsigned long bUserAdded);
void eventNotifyPendingConnectionLost();
void eventInitGame(class FString Options, class FString& ErrorMessage);
static class UClass* eventSetGameType(class FString MapName, class FString Options, class
FString Portal);
static class FString eventGetDefaultGameClassPath(class FString MapName, class FString
Options, class FString Portal);
static float GetFloatOption(class FString Options, class FString ParseString, float CurrentValue);
static int32_t GetIntOption(class FString Options, class FString ParseString, int32_t
CurrentValue);
static class FString SanitizePlayerName(class FString PlayerName);
static class FString SanitizeWhitespace(class FString Text);
static class FString DecodeURL(class FString Encoded);
static class FString EncodeURL(class FString Decoded);
static bool HasOption(class FString Options, class FString InKey);
static class FString ParseOption(class FString Options, class FString InKey);
static void GetKeyValue(class FString Pair, class FString& Key, class FString& Value);
static bool GrabOption(class FString& Options, class FString& Result);
void SetGameSpeed(float T);
void DebugPause();
void ForceClearUnpauseDelegates(class AActor* PauseActor);
void eventClearPause();
bool SetPause(class APlayerController* PC, struct FScriptDelegate CanUnpauseDelegate);
bool CanUnpause();
int32_t GetNumPlayers();
class FString GetNetworkNumber();
void InitGameReplicationInfo();
void eventGameEnding();
void NotifyNavigationChanged(class ANavigationPoint* N);
void DoNavFearCostFallOff();
bool ShouldStartInCinematicMode(int32_t& OutHidePlayer, int32_t& OutHideHud, int32_t&
OutDisableMovement, int32_t& OutDisableTurning, int32_t& OutDisableInput);
void ResetLevel();
bool ShouldReset(class AActor* ActorToReset);
void Reset();
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void eventPostBeginPlay();
class ACoverReplicator* GetCoverReplicator();
void eventPreBeginPlay();
bool GetMapCommonPackageName(class FString& InFilename, class FString&
OutCommonPackageName);
bool GetSupportedGameTypes(unsigned long bCheckExt, class FString& InFilename, struct
FGameTypePrefix& OutGameType);
};

// Class Engine.Mutator
// 0x001C (0x0268 - 0x0284)
class AMutator : public AInfo
{
public:
class AMutator*                    NextMutator;                    // 0x0268 (0x0008)
[0x0000000000000000]
TArray<class FString>                 GroupNames;                    // 0x0270 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                    bUserAdded : 1;                // 0x0280 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Mutator");
}

return uClassPointer;
};

void ScoreKill(class AController* Killer, class AController* Killed);
void ScoreObjective(class APlayerReplicationInfo* Scorer, int32_t Score);
bool CheckEndGame(class APlayerReplicationInfo* Winner, class FString Reason);
bool HandleRestartGame();
class ANavigationPoint* FindPlayerStart(class AController* Player, uint8_t InTeam, class FString
IncomingName);
void GetSeamlessTravelActorList(unsigned long bToEntry, TArray<class AActor*>& ActorList);
void InitMutator(class FString Options, class FString& ErrorMessage);
void NotifyLogin(class AController* NewPlayer);
void NotifyLogout(class AController* Exiting);
bool CheckReplacement(class AActor* Other);
bool CheckRelevance(class AActor* Other);
bool IsRelevant(class AActor* Other);
bool AlwaysKeep(class AActor* Other);
void AddMutator(class AMutator* M);
void ModifyPlayer(class APawn* Other);
void ModifyLogin(class FString& Portal, class FString& Options);
void Mutate(class FString MutateString, class APlayerController* Sender);
void eventDestroyed();
bool MutatorIsAllowed();
void eventPreBeginPlay();
};

// Class Engine.Route
// 0x0028 (0x0268 - 0x0290)
class ARoute : public AInfo
{
public:
struct FPointer                    VfTable_IEditorLinkSelectionInterface;      // 0x0268
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
uint8_t                            RouteType;                       // 0x0270 (0x0001)
[0x0000000000000001] (CPF_Edit)
TArray<struct FActorReference>             RouteList;                     // 0x0278 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                              FudgeFactor;                     // 0x0288 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                            RouteIndexOffset;                   // 0x028C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Route");
}
```

```
return uClassPointer;
};

int32_t MoveOntoRoutePath(class APawn* P, uint8_t RouteDirection, float DistFudgeFactor);
int32_t ResolveRouteIndex(int32_t Idx, uint8_t RouteDirection, uint8_t& out_bComplete, uint8_t&
out_bReverse);
};

// Class Engine.WindPointSource
// 0x0008 (0x0268 - 0x0270)
class AWindPointSource : public AInfo
{
public:
class UWindPointSourceComponent*                Component;                              // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WindPointSource");
}

return uClassPointer;
};

};

// Class Engine.Keypoint
// 0x0008 (0x0268 - 0x0270)
class AKeypoint : public AActor
{
public:
class USpriteComponent*                        SpriteComp;                             // 0x0268 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Keypoint");
}

return uClassPointer;
};

};
```

```
}

// Class Engine.TargetPoint
// 0x000C (0x0270 - 0x027C)
class ATargetPoint : public AKeypoint
{
public:
class UTexture2D*                    SpawnSpriteTexture;                    // 0x0270 (0x0008)
[0x0000000800002000] (CPF_Transient)
int32_t                              SpawnRefCount;                         // 0x0278 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TargetPoint");
}

return uClassPointer;
};

};

// Class Engine.MaterialInstanceActor
// 0x0008 (0x0268 - 0x0270)
class AMaterialInstanceActor : public AActor
{
public:
class UMaterialInstanceConstant*          MatInst;                          // 0x0268 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialInstanceActor");
}

return uClassPointer;
};

};

// Class Engine.MatineeActor
// 0x00E4 (0x0268 - 0x034C)
class AMatineeActor : public AActor
{
public:
```

```
class USeqAct_Interp*                    InterpAction;                          // 0x0268 (0x0008)
[0x0000000000000022] (CPF_Const | CPF_Net)
unsigned long                            bIsPlaying : 1;                        // 0x0270 (0x0004)
[0x0000000000000020] [0x00000001] (CPF_Net)
unsigned long                            bReversePlayback : 1;                  // 0x0270 (0x0004)
[0x0000000000000020] [0x00000002] (CPF_Net)
unsigned long                            bPaused : 1;                           // 0x0270 (0x0004)
[0x0000000000000020] [0x00000004] (CPF_Net)
unsigned long                            AllAIGroupsInitialized : 1;            // 0x0270 (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
float                              PlayRate;                             // 0x0274 (0x0004)
[0x0000000000000020] (CPF_Net)
float                              Position;                             // 0x0278 (0x0004)
[0x0000000000000020] (CPF_Net)
struct FName                             AIGroupNames[0xA];                     // 0x027C (0x0050)
[0x0000000000000020] (CPF_Net)
class APawn*                             AIGroupPawns[0xA];                     // 0x02D0 (0x0050)
[0x0000000000000020] (CPF_Net)
int32_t                            AIGroupInitStage[0xA];                 // 0x0320 (0x0028)
[0x0000000000002000] (CPF_Transient)
float                              ClientSidePositionErrorTolerance;           // 0x0348 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MatineeActor");
}

return uClassPointer;
};

void CheckPriorityRefresh();
void eventUpdate();
void AddAIGroupActor(class UInterpGroupInstAI* AIGroupInst);
};

// Class Engine.NavigationPoint
// 0x011C (0x0268 - 0x0384)
class ANavigationPoint : public AActor
{
public:
unsigned long                            bEndPoint : 1;                         // 0x0268 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                            bTransientEndPoint : 1;                // 0x0268 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                            bHideEditorPaths : 1;                  // 0x0268 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                            bCanReach : 1;                         // 0x0268 (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
```

```
unsigned long                          bBlocked : 1;                          // 0x0268 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bOneWayPath : 1;                       // 0x0268 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bNeverUseStrafing : 1;                 // 0x0268 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                          bAlwaysUseStrafing : 1;                // 0x0268 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                          bForceNoStrafing : 1;                  // 0x0268 (0x0004)
[0x0000000000000002] [0x00000100] (CPF_Const)
unsigned long                          bAutoBuilt : 1;                        // 0x0268 (0x0004)
[0x0000000000000002] [0x00000200] (CPF_Const)
unsigned long                          bSpecialMove : 1;                      // 0x0268 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                          bNoAutoConnect : 1;                    // 0x0268 (0x0004)
[0x0000000000000000] [0x00000800]
unsigned long                          bNotBased : 1;                         // 0x0268 (0x0004)
[0x0000000000000002] [0x00001000] (CPF_Const)
unsigned long                          bPathsChanged : 1;                     // 0x0268 (0x0004)
[0x0000000000000002] [0x00002000] (CPF_Const)
unsigned long                          bDestinationOnly : 1;                  // 0x0268 (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                          bSourceOnly : 1;                       // 0x0268 (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                          bSpecialForced : 1;                    // 0x0268 (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                          bMustBeReachable : 1;                  // 0x0268 (0x0004)
[0x0000000000000000] [0x00020000]
unsigned long                          bBlockable : 1;                        // 0x0268 (0x0004)
[0x0000000000000000] [0x00040000]
unsigned long                          bFlyingPreferred : 1;                  // 0x0268 (0x0004)
[0x0000000000000000] [0x00080000]
unsigned long                          bAlreadyVisited : 1;                   // 0x0268 (0x0004)
[0x0000000000002000] [0x00100000] (CPF_Transient)
unsigned long                          bMakeSourceOnly : 1;                   // 0x0268 (0x0004)
[0x0000000000000001] [0x00200000] (CPF_Edit)
unsigned long                          bMustTouchToReach : 1;                 // 0x0268 (0x0004)
[0x0000000000000000] [0x00400000]
unsigned long                          bCanWalkOnToReach : 1;                 // 0x0268 (0x0004)
[0x0000000000000000] [0x00800000]
unsigned long                          bBuildLongPaths : 1;                   // 0x0268 (0x0004)
[0x0000000000000000] [0x01000000]
unsigned long                          bHasCrossLevelPaths : 1;               // 0x0268 (0x0004)
[0x0000000000000002] [0x02000000] (CPF_Const)
unsigned long                          bShouldSaveForCheckpoint : 1;          // 0x0268 (0x0004)
[0x0000000000002000] [0x04000000] (CPF_Transient)
struct FNavigationOctreeObject                 NavOctreeObject;               // 0x0270
(0x0040) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class UReachSpec*>                      PathList;                      // 0x02B0 (0x0010)
[0x0000000004620003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink |
CPF_EditInline)
TArray<struct FActorReference>                 EditorProscribedPaths;         // 0x02C0
(0x0010) [0x0000000800600000] (CPF_NeedCtorLink)
TArray<struct FActorReference>                 EditorForcedPaths;             // 0x02D0
```

```
(0x0010) [0x0000000800600000] (CPF_NeedCtorLink)
TArray<struct FActorReference>                Volumes;                              // 0x02E0 (0x0010)
[0x0000000000420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)
int32_t                             visitedWeight;                     // 0x02F0 (0x0004)
[0x0000000000000000]
int32_t                             bestPathWeight;                    // 0x02F4 (0x0004)
[0x0000000000000002] (CPF_Const)
class ANavigationPoint*             nextNavigationPoint;               // 0x02F8 (0x0008)
[0x0000000000000002] (CPF_Const)
class ANavigationPoint*             nextOrdered;                       // 0x0300 (0x0008)
[0x0000000000000002] (CPF_Const)
class ANavigationPoint*             prevOrdered;                       // 0x0308 (0x0008)
[0x0000000000000002] (CPF_Const)
class ANavigationPoint*             previousPath;                      // 0x0310 (0x0008)
[0x0000000000000002] (CPF_Const)
int32_t                             Cost;                              // 0x0318 (0x0004)
[0x0000000000000000]
int32_t                             ExtraCost;                         // 0x031C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             TransientCost;                     // 0x0320 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                             FearCost;                          // 0x0324 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FDebugNavCost>                CostArray;                         // 0x0328 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                              LastDetourWeight;                   // 0x0338 (0x0004)
[0x0000000000000002] (CPF_Const)
class UCylinderComponent*                CylinderComponent;                  // 0x0340
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FCylinder                       MaxPathSize;                      // 0x0348 (0x0008)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
struct FGuid                          NavGuid;                           // 0x0350 (0x0010)
[0x0000000000220003] (CPF_Edit | CPF_Const | CPF_EditConst)
class USpriteComponent*                 GoodSprite;                        // 0x0360 (0x0008)
[0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)
class USpriteComponent*                 BadSprite;                         // 0x0368 (0x0008)
[0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)
int32_t                             NetworkID;                         // 0x0370 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
class APawn*                           AnchoredPawn;                      // 0x0378 (0x0008)
[0x0000000000002000] (CPF_Transient)
float                              LastAnchoredPawnTime;               // 0x0380 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavigationPoint");
}

return uClassPointer;
};

class FString eventGetDebugAbbrev();
void ApplyCheckpointRecord(struct ANavigationPoint_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ANavigationPoint_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
void eventShutDown();
void OnToggle(class USeqAct_Toggle* inAction);
bool IsOnDifferentNetwork(class ANavigationPoint* Nav);
static bool GetAllNavInRadius(class AActor* ChkActor, struct FVector ChkPoint, float Radius,
unsigned long bSkipBlocked, int32_t inNetworkID, struct FCylinder MinSize, TArray<class
ANavigationPoint*>& out_NavList);
static class ANavigationPoint* GetNearestNavToPoint(class AActor* ChkActor, struct FVector
ChkPoint, class UClass* RequiredClass, TArray<class ANavigationPoint*> ExcludeList);
static class ANavigationPoint* GetNearestNavToActor(class AActor* ChkActor, class UClass*
RequiredClass, TArray<class ANavigationPoint*> ExcludeList, float MinDist);
bool ProceedWithMove(class APawn* Other);
bool eventSuggestMovePreparation(class APawn* Other);
float eventDetourWeight(class APawn* Other, float PathWeight);
bool eventAccept(class AActor* Incoming, class AActor* Source);
int32_t eventSpecialCost(class APawn* Seeker, class UReachSpec* Path);
bool CanTeleport(class AActor* A);
bool IsUsableAnchorFor(class APawn* P);
class UReachSpec* GetReachSpecTo(class ANavigationPoint* Nav, class UClass* SpecClass);
void GetBoundingCylinder(float& CollisionRadius, float& CollisionHeight);
};

// Class Engine.CoverLink
// 0x009D (0x0384 - 0x0421)
class ACoverLink : public ANavigationPoint
{
public:
unsigned long                   GLOBAL_bUseSlotMarkers : 1;          // 0x0388 (0x0004)
[0x0000000000044000] [0x00000001] (CPF_Config | CPF_GlobalConfig)
unsigned long                   bDisabled : 1;                       // 0x0388 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                   bClaimAllSlots : 1;                  // 0x0388 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                   bAutoSort : 1;                       // 0x0388 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                   bAutoAdjust : 1;                     // 0x0388 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                   bCircular : 1;                       // 0x0388 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                   bLooped : 1;                         // 0x0388 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                   bPlayerOnly : 1;                     // 0x0388 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                   bDynamicCover : 1;                   // 0x0388 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                   bFractureOnTouch : 1;                // 0x0388 (0x0004)
```

```
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                       bDebug_FireLinks : 1;                  // 0x0388 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                       bDebug_ExposedLinks : 1;                  // 0x0388 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                       bDebug_CoverGen : 1;                  // 0x0388 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                       bDoAutoSlotDensityFixup : 1;                // 0x0388 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
float                              LeanTraceDist;                       // 0x038C (0x0004)
[0x0000000000000000]
TArray<struct FCoverSlot>                Slots;                          // 0x0390 (0x0010)
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)
TArray<struct FDynamicLinkInfo>              DynamicLinkInfos;                     // 0x03A0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<class APawn*>                    Claims;                          // 0x03B0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                              InvalidateDistance;                    // 0x03C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              MaxFireLinkDist;                     // 0x03C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                        CircularOrigin;                      // 0x03C8 (0x000C)
[0x0000000000000002] (CPF_Const)
float                              CircularRadius;                      // 0x03D4 (0x0004)
[0x0000000000000002] (CPF_Const)
float                              AlignDist;                         // 0x03D8 (0x0004)
[0x0000000000000002] (CPF_Const)
float                              AutoCoverSlotInterval;                  // 0x03DC (0x0004)
[0x0000000000000002] (CPF_Const)
float                              StandHeight;                        // 0x03E0 (0x0004)
[0x0000000000000002] (CPF_Const)
float                              MidHeight;                         // 0x03E4 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FVector                        StandingLeanOffset;                    // 0x03E8 (0x000C)
[0x0000000000000002] (CPF_Const)
struct FVector                        CrouchLeanOffset;                     // 0x03F4 (0x000C)
[0x0000000000000002] (CPF_Const)
struct FVector                        PopupOffset;                        // 0x0400 (0x000C)
[0x0000000000000002] (CPF_Const)
float                              SlipDist;                          // 0x040C (0x0004)
[0x0000000000000002] (CPF_Const)
float                              TurnDist;                          // 0x0410 (0x0004)
[0x0000000000000002] (CPF_Const)
float                              DangerScale;                        // 0x0414 (0x0004)
[0x0000000000000001] (CPF_Edit)
class ACoverLink*                       NextCoverLink;                      // 0x0418 (0x0008)
[0x0000000000000002] (CPF_Const)
uint8_t                             LocationDescription;                   // 0x0420 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CoverLink");
}

return uClassPointer;
};

class FString eventGetDebugAbbrev();
uint8_t GetLocationDescription(int32_t SlotIdx);
class FString eventGetDebugString(int32_t SlotIdx);
int32_t AddCoverSlot(struct FVector SlotLocation, struct FRotator SlotRotation, int32_t SlotIdx,
unsigned long bForceSlotUpdate, class AScout* Scout);
bool GetSwatTurnTarget(int32_t SlotIdx, int32_t Direction, struct FCoverInfo& out_Info);
void eventShutDown();
void ApplyCheckpointRecord(struct ANavigationPoint_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ANavigationPoint_FCheckpointRecord& Record);
void OnToggle(class USeqAct_Toggle* inAction);
bool IsEnabled();
bool AutoAdjustSlot(int32_t SlotIdx, unsigned long bOnlyCheckLeans);
void OnModifyCover(class USeqAct_ModifyCover* Action);
void eventSetSlotPlayerOnly(int32_t SlotIdx, unsigned long bInPlayerOnly);
void NotifySlotOwnerCoverDisabled(int32_t SlotIdx, unsigned long bAIOnly);
void eventSetSlotEnabled(int32_t SlotIdx, unsigned long bEnable);
void eventSetDisabled(unsigned long bNewDisabled);
void GetSlotActions(int32_t SlotIdx, TArray<uint8_t>& Actions);
bool HasFireLinkTo(int32_t SlotIdx, struct FCoverInfo ChkCover, unsigned long
bAllowFallbackLinks);
bool GetFireLinkTo(int32_t SlotIdx, struct FCoverInfo ChkCover, uint8_t ChkAction, uint8_t
ChkType, int32_t& out_FireLinkIdx, TArray<int32_t>& out_Items);
bool AllowLeftTransition(int32_t SlotIdx);
bool AllowRightTransition(int32_t SlotIdx);
int32_t GetSlotIdxToRight(int32_t SlotIdx, int32_t Cnt);
int32_t GetSlotIdxToLeft(int32_t SlotIdx, int32_t Cnt);
bool IsRightEdgeSlot(int32_t SlotIdx, unsigned long bIgnoreLeans);
bool IsLeftEdgeSlot(int32_t SlotIdx, unsigned long bIgnoreLeans);
bool IsEdgeSlot(int32_t SlotIdx, unsigned long bIgnoreLeans);
bool FindSlots(struct FVector CheckLocation, float MaxDistance, int32_t& LeftSlotIdx, int32_t&
RightSlotIdx);
bool IsStationarySlot(int32_t SlotIdx);
bool IsValidClaimBetween(class APawn* ChkClaim, int32_t StartSlotIdx, int32_t EndSlotIdx,
unsigned long bSkipTeamCheck, unsigned long bSkipOverlapCheck);
bool IsValidClaim(class APawn* ChkClaim, int32_t SlotIdx, unsigned long bSkipTeamCheck,
unsigned long bSkipOverlapCheck);
bool eventUnClaim(class APawn* OldClaim, int32_t SlotIdx, unsigned long bUnclaimAll);
bool eventClaim(class APawn* NewClaim, int32_t SlotIdx);
void eventSetInvalidUntil(int32_t SlotIdx, float TimeToBecomeValid);
bool IsExposedTo(int32_t SlotIdx, struct FCoverInfo ChkSlot, float& out_ExposedScale);
struct FVector GetSlotViewPoint(int32_t SlotIdx, uint8_t Type, uint8_t Action);
struct FRotator GetSlotRotation(int32_t SlotIdx, unsigned long bForceUseOffset);
struct FVector GetSlotLocation(int32_t SlotIdx, unsigned long bForceUseOffset);
static void UnPackFireLinkInteractionInfo(uint8_t PackedByte, uint8_t& SrcType, uint8_t&
SrcAction, uint8_t& DestType, uint8_t& DestAction);
```

```
static uint8_t PackFireLinkInteractionInfo(uint8_t SrcType, uint8_t SrcAction, uint8_t DestType,
uint8_t DestAction);
bool GetFireLinkTargetCoverInfo(int32_t SlotIdx, int32_t FireLinkIdx, uint8_t ArrayID, struct
FCoverInfo& out_Info);
};

// Class Engine.DynamicAnchor
// 0x000C (0x0384 - 0x0390)
class ADynamicAnchor : public ANavigationPoint
{
public:
class AController*                    CurrentUser;                    // 0x0388 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DynamicAnchor");
}

return uClassPointer;
};

};

// Class Engine.LiftCenter
// 0x002C (0x0384 - 0x03B0)
class ALiftCenter : public ANavigationPoint
{
public:
class AInterpActor*                   MyLift;                         // 0x0388 (0x0008)
[0x0000000000000000]
float                             MaxDist2D;                      // 0x0390 (0x0004)
[0x0000000000000000]
struct FVector                        LiftOffset;                     // 0x0394 (0x000C)
[0x0000000000000000]
unsigned long                         bJumpLift : 1;                  // 0x03A0 (0x0004)
[0x0000000000000000] [0x00000001]
float                             CollisionHeight;                // 0x03A4 (0x0004)
[0x0000000000000000]
class ATrigger*                       LiftTrigger;                    // 0x03A8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.LiftCenter");
}

return uClassPointer;
};

bool ProceedWithMove(class APawn* Other);
bool eventSuggestMovePreparation(class APawn* Other);
class AActor* eventSpecialHandling(class APawn* Other);
void eventPostBeginPlay();
};

// Class Engine.LiftExit
// 0x0010 (0x0384 - 0x0394)
class ALiftExit : public ANavigationPoint
{
public:
class ALiftCenter*                    MyLiftCenter;                    // 0x0388 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                        bExitOnly : 1;                  // 0x0390 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LiftExit");
}

return uClassPointer;
};

bool eventSuggestMovePreparation(class APawn* Other);
void WaitForLift(class APawn* Other);
bool CanBeReachedFromLiftBy(class APawn* Other);
};

// Class Engine.PathNode
// 0x0004 (0x0384 - 0x0388)
class APathNode : public ANavigationPoint
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PathNode");
```

```
}

return uClassPointer;
};

class FString eventGetDebugAbbrev();
};

// Class Engine.VolumePathNode
// 0x0008 (0x0388 - 0x0390)
class AVolumePathNode : public APathNode
{
public:
float                       StartingRadius;                 // 0x0388 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       StartingHeight;                 // 0x038C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.VolumePathNode");
}

return uClassPointer;
};

};

// Class Engine.PlayerStart
// 0x0014 (0x0384 - 0x0398)
class APlayerStart : public ANavigationPoint
{
public:
unsigned long                    bEnabled : 1;                  // 0x0388 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bPrimaryStart : 1;               // 0x0388 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bBestStart : 1;                // 0x0388 (0x0004)
[0x0000000000000000] [0x00000004]
int32_t                          TeamIndex;                     // 0x038C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          Score;                       // 0x0390 (0x0004)
[0x0000000000000000]
int32_t                          SelectionIndex;                  // 0x0394 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlayerStart");
}

return uClassPointer;
};

void eventPostRenderFor(class APlayerController* PC, class UCanvas* Canvas, struct FVector
CameraPosition, struct FVector CameraDir);
void OnToggle(class USeqAct_Toggle* Action);
};

// Class Engine.PortalMarker
// 0x000C (0x0384 - 0x0390)
class APortalMarker : public ANavigationPoint
{
public:
class APortalTeleporter*                MyPortal;                        // 0x0388 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PortalMarker");
}

return uClassPointer;
};

bool CanTeleport(class AActor* A);
};

// Class Engine.Pylon
// 0x0120 (0x0384 - 0x04A4)
class APylon : public ANavigationPoint
{
public:
struct FPointer                     VfTable_IEditorLinkSelectionInterface;       // 0x0388
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                     VfTable_IInterface_NavigationHandle;          // 0x0390
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                     NavMeshPtr;                          // 0x0398 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                     ObstacleMesh;                        // 0x03A0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                     DynamicObstacleMesh;                     // 0x03A8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
```

```
struct FPointer                         WorkingSetPtr;                          // 0x03B0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                         PathObjectsThatAffectThisPylon;                 // 0x03B8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FVector>                          NextPassSeedList;                       // 0x03C0 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
struct FOctreeElementId                         OctreeId;                               // 0x03D0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                         OctreeIWasAddedTo;                      // 0x03E0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
class APylon*                           NextPylon;                              // 0x03E8 (0x0008)
[0x0000000000000002] (CPF_Const)
TArray<class AVolume*>                          ExpansionVolumes;                       // 0x03F0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                   ExpansionRadius;                        // 0x0400 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MaxExpansionRadius;                     // 0x0404 (0x0004)
[0x0000000000000002] (CPF_Const)
class UDrawPylonRadiusComponent*                PylonRadiusPreview;                     // 0x0408
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
unsigned long                           bImportedMesh : 1;                      // 0x0410 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                           bUseExpansionSphereOverride : 1;                // 0x0410
(0x0004) [0x0000000000000000] [0x00000002]
unsigned long                           bNeedsCostCheck : 1;                    // 0x0410 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                           bPylonInHighLevelPath : 1;                      // 0x0410 (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
unsigned long                           bUseRecast : 1;                         // 0x0410 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                           bAllowRecastGenerator : 1;                      // 0x0410 (0x0004)
[0x0000000000002000] [0x00000020] (CPF_Transient)
unsigned long                           bDrawEdgePolys : 1;                     // 0x0410 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                           bDrawPolyBounds : 1;                    // 0x0410 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                           bRenderInShowPaths : 1;                 // 0x0410 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                           bDrawWalkableSurface : 1;                       // 0x0410 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                           bDrawObstacleSurface : 1;                       // 0x0410 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                           bSolidObstaclesInGame : 1;                      // 0x0410 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                           bBuildThisPylon : 1;                    // 0x0410 (0x0004)
[0x0000000000002000] [0x00001000] (CPF_Transient)
unsigned long                           bDisabled : 1;                          // 0x0410 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                           bForceObstacleMeshCollision : 1;                // 0x0410 (0x0004)
[0x0000000000000000] [0x00004000]
struct FVector                          ExpansionSphereCenter;                  // 0x0414 (0x000C)
[0x0000000000000000]
class UNavMeshRenderingComponent*               RenderingComp;                          // 0x0420
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
```

```
class USpriteComponent*                    BrokenSprite;                        // 0x0428 (0x0008)
[0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)
TArray<class APylon*>                      ImposterPylons;                      // 0x0430 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class AActor*>                      OnBuild_DisableCollisionForThese;          // 0x0440
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class AActor*>                      OnBuild_EnableCollisionForThese;           // 0x0450
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                              MaxPolyHeight_Optional;                 // 0x0460 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                            NavMeshGenerator;                      // 0x0464 (0x0001)
[0x0000000000000000]
TArray<struct FKAggregateGeom>             VoxelFilterBounds;                     // 0x0468
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FMatrix>                     VoxelFilterTM;                        // 0x0478 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                            DebugEdgeCount;                        // 0x0488 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                         DebugPathExtent;                       // 0x048C (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FVector                         DebugPathStartLocation;                  // 0x0498 (0x000C)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Pylon");
}

return uClassPointer;
};

bool CanReachPylon(class APylon* DestPylon, class AController* C);
void OnToggle(class USeqAct_Toggle* Action);
bool eventIsEnabled();
void eventSetEnabled(unsigned long bEnabled);
void PostBeginPlay();
void eventNotifyPathChanged();
void VerifyTopLevelConnections();
struct FVector GetTestPathExtent();
void FlushDynamicEdges();
void UpdateMeshForPreExistingNavMeshObstacles();
void OnPylonStatusChange();
};

// Class Engine.AISwitchablePylon
// 0x0008 (0x04A4 - 0x04AC)
class AAISwitchablePylon : public APylon
{
```

```
public:
unsigned long                    bOpen : 1;                    // 0x04A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AISwitchablePylon");
}

return uClassPointer;
};

bool eventIsEnabled();
void eventSetEnabled(unsigned long bEnabled);
void PostBeginPlay();
};

// Class Engine.DynamicPylon
// 0x0008 (0x04A4 - 0x04AC)
class ADynamicPylon : public APylon
{
public:
unsigned long                    bMoving : 1;                    // 0x04A8 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DynamicPylon");
}

return uClassPointer;
};

void eventStoppedMoving();
void eventStartedMoving();
void FlushDynamicEdges();
void RebuildDynamicEdges();
void PostBeginPlay();
};

// Class Engine.Teleporter
// 0x0030 (0x0384 - 0x03B4)
class ATeleporter : public ANavigationPoint
{
```

```
public:
class FString                              URL;                                // 0x0388 (0x0010)
[0x0000000000400021] (CPF_Edit | CPF_Net | CPF_NeedCtorLink)
struct FName                               ProductRequired;                    // 0x0398 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                              bChangesVelocity : 1;               // 0x03A0 (0x0004)
[0x0000000000000021] [0x00000001] (CPF_Edit | CPF_Net)
unsigned long                              bChangesYaw : 1;                    // 0x03A0 (0x0004)
[0x0000000000000021] [0x00000002] (CPF_Edit | CPF_Net)
unsigned long                              bReversesX : 1;                     // 0x03A0 (0x0004)
[0x0000000000000021] [0x00000004] (CPF_Edit | CPF_Net)
unsigned long                              bReversesY : 1;                     // 0x03A0 (0x0004)
[0x0000000000000021] [0x00000008] (CPF_Edit | CPF_Net)
unsigned long                              bReversesZ : 1;                     // 0x03A0 (0x0004)
[0x0000000000000021] [0x00000010] (CPF_Edit | CPF_Net)
unsigned long                              bEnabled : 1;                       // 0x03A0 (0x0004)
[0x0000000000000021] [0x00000020] (CPF_Edit | CPF_Net)
struct FVector                             TargetVelocity;                     // 0x03A4 (0x000C)
[0x0000000000000021] (CPF_Edit | CPF_Net)
float                                      LastFired;                          // 0x03B0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Teleporter");
}

return uClassPointer;
};

class AActor* eventSpecialHandling(class APawn* Other);
void eventPostTouch(class AActor* Other);
void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
bool eventAccept(class AActor* Incoming, class AActor* Source);
void eventPostBeginPlay();
bool CanTeleport(class AActor* A);
};

// Class Engine.Note
// 0x0010 (0x0268 - 0x0278)
class ANote : public AActor
{
public:
class FString                              Text;                               // 0x0268 (0x0010)
[0x0000001000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Note");
}

return uClassPointer;
};

};

// Class Engine.RigidBodyBase
// 0x0000 (0x0268 - 0x0268)
class ARigidBodyBase : public AActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RigidBodyBase");
}

return uClassPointer;
};

};

// Class Engine.SceneCaptureActor
// 0x0008 (0x0268 - 0x0270)
class ASceneCaptureActor : public AActor
{
public:
class USceneCaptureComponent*                    SceneCapture;                              // 0x0268
(0x0008) [0x000000000408000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCaptureActor");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
};

// Class Engine.SceneCapture2DActor
// 0x0008 (0x0270 - 0x0278)
class ASceneCapture2DActor : public ASceneCaptureActor
{
public:
class UDrawFrustumComponent*                    DrawFrustum;                              // 0x0270
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCapture2DActor");
}

return uClassPointer;
};

};

// Class Engine.SceneCaptureCubeMapActor
// 0x0010 (0x0270 - 0x0280)
class ASceneCaptureCubeMapActor : public ASceneCaptureActor
{
public:
class UStaticMeshComponent*                    StaticMesh;                              // 0x0270 (0x0008)
[0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component | CPF_EditInline)
class UMaterialInstanceConstant*               CubeMaterialInst;                        // 0x0278
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCaptureCubeMapActor");
}

return uClassPointer;
};

};
```

```
// Class Engine.SceneCaptureReflectActor
// 0x0010 (0x0270 - 0x0280)
class ASceneCaptureReflectActor : public ASceneCaptureActor
{
public:
class UStaticMeshComponent*                    StaticMesh;                              // 0x0270 (0x0008)
[0x000000000408000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class UMaterialInstanceConstant*               ReflectMaterialInst;                     // 0x0278
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCaptureReflectActor");
}

return uClassPointer;
};

};

// Class Engine.SceneCapturePortalActor
// 0x0000 (0x0280 - 0x0280)
class ASceneCapturePortalActor : public ASceneCaptureReflectActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCapturePortalActor");
}

return uClassPointer;
};

};

// Class Engine.PortalTeleporter
// 0x001C (0x0280 - 0x029C)
class APortalTeleporter : public ASceneCapturePortalActor
{
public:
class APortalTeleporter*                    SisterPortal;                              // 0x0280 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                          TextureResolutionX;                   // 0x0288 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          TextureResolutionY;                   // 0x028C (0x0004)
[0x0000000000000001] (CPF_Edit)
class APortalMarker*                     MyMarker;                          // 0x0290 (0x0008)
[0x0000000000000000]
unsigned long                        bMovablePortal : 1;                   // 0x0298 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bAlwaysTeleportNonPawns : 1;              // 0x0298 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                        bCanTeleportVehicles : 1;              // 0x0298 (0x0004)
[0x0000000000000000] [0x00000004]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PortalTeleporter");
}

return uClassPointer;
};

class UTextureRenderTarget2D* CreatePortalTexture();
struct FVector TransformHitLocation(struct FVector HitLocation);
struct FVector TransformVectorDir(struct FVector V);
bool TransformActor(class AActor* A);
};

// Class Engine.StaticMeshActorBase
// 0x0000 (0x0268 - 0x0268)
class AStaticMeshActorBase : public AActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticMeshActorBase");
}

return uClassPointer;
};

};

// Class Engine.StaticMeshActor
// 0x0020 (0x0268 - 0x0288)
class AStaticMeshActor : public AStaticMeshActorBase
{
public:
class UStaticMeshComponent*                    StaticMeshComponent;                     // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                          bDisableAutoBaseOnProcBuilding : 1;          // 0x0270
(0x0004) [0x0000000800000001] [0x00000001] (CPF_Edit)
unsigned long                          bProxy : 1;                           // 0x0270 (0x0004)
[0x0000000800000000] [0x00000002]
unsigned long                          bHiddenByProxy : 1;                      // 0x0270 (0x0004)
[0x0000000800000000] [0x00000004]
unsigned long                          OldCastShadow : 1;                       // 0x0270 (0x0004)
[0x0000000800000000] [0x00000008]
unsigned long                          OldAcceptsLights : 1;                    // 0x0270 (0x0004)
[0x0000000800000000] [0x00000010]
uint8_t                              OldCollisionType;                        // 0x0274 (0x0001)
[0x0000000800000000]
TArray<struct FPreCombinedStaticMeshActor>        PreCombinedStaticMeshActors;                //
0x0278 (0x0010) [0x0000000800420003] (CPF_Edit | CPF_Const | CPF_EditConst |
CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticMeshActor");
}

return uClassPointer;
};

void eventPreBeginPlay();
};

// Class Engine.StaticMeshCollectionActor
// 0x0014 (0x0268 - 0x027C)
class AStaticMeshCollectionActor : public AStaticMeshActorBase
{
public:
TArray<class UStaticMeshComponent*>              StaticMeshComponents;                    //
0x0268 (0x0010) [0x000000000448000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
int32_t                              MaxStaticMeshComponents;                 // 0x0278 (0x0004)
[0x0000000000004000] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticMeshCollectionActor");
}

return uClassPointer;
};

};

// Class Engine.StaticMeshActorBasedOnExtremeContent
// 0x0028 (0x0268 - 0x0290)
class AStaticMeshActorBasedOnExtremeContent : public AActor
{
public:
class UStaticMeshComponent*                StaticMeshComponent;                    // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
TArray<struct FSMMaterialSetterDatum>          ExtremeContent;                         // 0x0270
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FSMMaterialSetterDatum>          NonExtremeContent;                        // 0x0280
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticMeshActorBasedOnExtremeContent");
}

return uClassPointer;
};

void SetMaterialBasedOnExtremeContent();
void eventPostBeginPlay();
};

// Class Engine.Trigger
// 0x0010 (0x0268 - 0x0278)
class ATrigger : public AActor
{
public:
class UCylinderComponent*                CylinderComponent;                    // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                        bRecentlyTriggered : 1;                 // 0x0270 (0x0004)
[0x0000000000000000] [0x00000001]
float                            AITriggerDelay;                       // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Trigger");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct ATrigger_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ATrigger_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
void UnTrigger();
void NotifyTriggered();
void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
void eventPostBeginPlay();
};

// Class Engine.Trigger_PawnsOnly
// 0x0000 (0x0278 - 0x0278)
class ATrigger_PawnsOnly : public ATrigger
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Trigger_PawnsOnly");
}

return uClassPointer;
};

};

// Class Engine.ActorComponent
// 0x002D (0x0070 - 0x009D)
class UActorComponent : public UComponent
{
public:
TArray<class APlayerController*>                LocalViewers;                            // 0x0070 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
uint8_t                                 BulletSceneGroup;                        // 0x0080 (0x0001)
[0x0000000000000002] (CPF_Const)
struct FPointer                         Scene;                                   // 0x0088 (0x0008)
```

```
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
class AActor*                         Owner;                          // 0x0090 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                         bAttached : 1;                  // 0x0098 (0x0004)
[0x0000000000003002] [0x00000001] (CPF_Const | CPF_Native | CPF_Transient)
unsigned long                         bTickInEditor : 1;              // 0x0098 (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                         bNeedsReattach : 1;             // 0x0098 (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                         bNeedsUpdateTransform : 1;      // 0x0098 (0x0004)
[0x0000000000002002] [0x00000008] (CPF_Const | CPF_Transient)
uint8_t                          TickGroup;                          // 0x009C (0x0001)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorComponent");
}

return uClassPointer;
};

class APlayerController* GetFirstLocalViewer();
void SetLocalViewer(class APlayerController* NewLocalViewer);
void ClearLocalViewers();
void RemoveLocalViewer(class APlayerController* NewLocalViewer);
void AddLocalViewer(class APlayerController* NewLocalViewer);
void DetachFromAny();
void ForceUpdate(unsigned long bTransformOnly);
void SetComponentRBFixed(unsigned long bFixed);
void SetTickGroup(uint8_t NewTickGroup);
};

// Class Engine.AudioComponent
// 0x026B (0x009D - 0x0308)
class UAudioComponent : public UActorComponent
{
public:
class USoundCue*                      SoundCue;                       // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundNode*                      CueFirstNode;                   // 0x00A8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FAudioComponentParam>          InstanceParameters;              // 0x00B0
(0x0010) [0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)
unsigned long                         bUseOwnerLocation : 1;          // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                         bAutoPlay : 1;                  // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                         bAutoDestroy : 1;               // 0x00C0 (0x0004)
```

```
[0x0000000000000000] [0x00000004]
unsigned long                    bStopWhenOwnerDestroyed : 1;          // 0x00C0
(0x0004) [0x0000000000000000] [0x00000008]
unsigned long                    bShouldRemainActiveIfDropped : 1;      // 0x00C0
(0x0004) [0x0000000000000000] [0x00000010]
unsigned long                    bWasOccluded : 1;                      // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                    bSuppressSubtitles : 1;                // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000040] (CPF_Transient)
unsigned long                    bWasPlaying : 1;                       // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000080] (CPF_Transient)
unsigned long                    bAllowSpatialization : 1;              // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                    bFinished : 1;                         // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000200] (CPF_Transient)
unsigned long                    bApplyRadioFilter : 1;                 // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000400] (CPF_Transient)
unsigned long                    bRadioFilterSelected : 1;              // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000800] (CPF_Transient)
unsigned long                    bPreviewComponent : 1;                 // 0x00C0 (0x0004)
[0x0000000000002000] [0x00001000] (CPF_Transient)
unsigned long                    bIgnoreForFlushing : 1;                // 0x00C0 (0x0004)
[0x0000000000002000] [0x00002000] (CPF_Transient)
float                        StereoBleed;                           // 0x00C4 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                        LFEBleed;                              // 0x00C8 (0x0004)
[0x0000000000002000] (CPF_Transient)
unsigned long                    bEQFilterApplied : 1;                  // 0x00CC (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                    bAlwaysPlay : 1;                       // 0x00CC (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                    bIsUISound : 1;                        // 0x00CC (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                    bIsMusic : 1;                          // 0x00CC (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
unsigned long                    bReverb : 1;                           // 0x00CC (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                    bCenterChannelOnly : 1;                // 0x00CC (0x0004)
[0x0000000000002000] [0x00000020] (CPF_Transient)
unsigned long                    bIsCachedInPool : 1;                   // 0x00CC (0x0004)
[0x0000000000000000] [0x00000040]
TArray<struct FPointer>              WaveInstances;                     // 0x00D0 (0x0010)
[0x0000000000201002] (CPF_Const | CPF_Native)
TArray<uint8_t>                  SoundNodeData;                         // 0x00E0 (0x0010)
[0x0000000000201002] (CPF_Const | CPF_Native)
uint8_t                          UnknownData00[0x50];                   // 0x00F0 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AudioComponent.SoundNodeOffsetMap
struct FMultiMap_Mirror              SoundNodeResetWaveMap;             // 0x0140
(0x0050) [0x0000000000201002] (CPF_Const | CPF_Native)
struct FPointer                  Listener;                              // 0x0190 (0x0008)
[0x0000000000201002] (CPF_Const | CPF_Native)
float                        PlaybackTime;                          // 0x0198 (0x0004)
[0x0000000000201002] (CPF_Const | CPF_Native)
class APortalVolume*                 PortalVolume;                      // 0x01A0 (0x0008)
```

```
[0x0000000000201002] (CPF_Const | CPF_Native)
struct FVector                     Location;                         // 0x01A8 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                     ComponentLocation;                      // 0x01B4 (0x000C)
[0x0000000000201002] (CPF_Const | CPF_Native)
struct FRotator                     Rotation;                         // 0x01C0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FRotator                     ComponentRotation;                      // 0x01CC (0x000C)
[0x0000000000201002] (CPF_Const | CPF_Native)
class AActor*                      LastOwner;                          // 0x01D8 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                         SubtitlePriority;                    // 0x01E0 (0x0004)
[0x0000000000001000] (CPF_Native)
float                         FadeInStartTime;                      // 0x01E4 (0x0004)
[0x0000000000000000]
float                         FadeInStopTime;                       // 0x01E8 (0x0004)
[0x0000000000000000]
float                         FadeInTargetVolume;                     // 0x01EC (0x0004)
[0x0000000000000000]
float                         FadeOutStartTime;                      // 0x01F0 (0x0004)
[0x0000000000000000]
float                         FadeOutStopTime;                       // 0x01F4 (0x0004)
[0x0000000000000000]
float                         FadeOutTargetVolume;                     // 0x01F8 (0x0004)
[0x0000000000000000]
float                         AdjustVolumeStartTime;                    // 0x01FC (0x0004)
[0x0000000000000000]
float                         AdjustVolumeStopTime;                     // 0x0200 (0x0004)
[0x0000000000000000]
float                         AdjustVolumeTargetVolume;                   // 0x0204 (0x0004)
[0x0000000000000000]
float                         CurrAdjustVolumeTargetVolume;                 // 0x0208 (0x0004)
[0x0000000000000000]
class USoundNode*                         CurrentNotifyBufferFinishedHook;             // 0x0210
(0x0008) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FVector                     CurrentLocation;                    // 0x0218 (0x000C)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FRotator                     CurrentRotation;                    // 0x0224 (0x000C)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FVector                     CurrentVelocity;                    // 0x0230 (0x000C)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                         CurrentVolume;                      // 0x023C (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                         CurrentPitch;                     // 0x0240 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                         CurrentHighFrequencyGain;                  // 0x0244 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                        CurrentUseSpatialization;                  // 0x0248 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                        CurrentNotifyOnLoop;                    // 0x024C (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                         OmniRadius;                       // 0x0250 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                         CurrentVolumeMultiplier;                  // 0x0254 (0x0004)
```

```
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          CurrentPitchMultiplier;              // 0x0258 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          CurrentHighFrequencyGainMultiplier;         // 0x025C (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          CurrentVoiceCenterChannelVolume;           // 0x0260 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          CurrentRadioFilterVolume;               // 0x0264 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          CurrentRadioFilterVolumeThreshold;          // 0x0268 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                      LastUpdateTime;                    // 0x0270 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          SourceInteriorVolume;                // 0x0278 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          SourceInteriorLPF;                 // 0x027C (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          CurrentInteriorVolume;               // 0x0280 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          CurrentInteriorLPF;                // 0x0284 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FVector                     LastLocation;                    // 0x0288 (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FInteriorSettings                LastInteriorSettings;                 // 0x0294 (0x0024)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                          LastReverbVolumeIndex;              // 0x02B8 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                          VolumeMultiplier;                 // 0x02BC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          PitchMultiplier;                 // 0x02C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          HighFrequencyGainMultiplier;            // 0x02C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          OcclusionCheckInterval;              // 0x02C8 (0x0004)
[0x0000000000000000]
float                          LastOcclusionCheckTime;               // 0x02CC (0x0004)
[0x0000000000002000] (CPF_Transient)
class UDrawSoundRadiusComponent*          PreviewSoundRadius;                 // 0x02D0
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
struct FScriptDelegate                __OnAudioFinished__Delegate;           // 0x02D8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                __OnQueueSubtitles__Delegate;          // 0x02F0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AudioComponent");
}
return uClassPointer;
};

void eventOcclusionChanged(unsigned long bNowOccluded);
void OnQueueSubtitles(TArray<struct FSubtitleCue> Subtitles, float CueDuration);
void OnAudioFinished(class UAudioComponent* AC);
void ResetToDefaults();
void SetWaveParameter(struct FName InName, class USoundNodeWave* InWave);
void SetFloatParameter(struct FName InName, float InFloat);
void AdjustVolume(float AdjustVolumeDuration, float AdjustVolumeLevel);
void FadeOut(float FadeOutDuration, float FadeVolumeLevel);
void FadeIn(float FadeInDuration, float FadeVolumeLevel);
bool IsFadingOut();
bool IsFadingIn();
bool IsPlaying();
void Stop();
void Play();
};

// Class Engine.SplineAudioComponent
// 0x0018 (0x0308 - 0x0320)
class USplineAudioComponent : public UAudioComponent
{
public:
float                                   ListenerScopeRadius;                      // 0x0308 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 ClosestPointOnSplineIndex;                // 0x030C (0x0004)
[0x0000000000000000]
TArray<struct FInterpPointOnSpline>             Points;                                 // 0x0310 (0x0010)
[0x0000000000500000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SplineAudioComponent");
}

return uClassPointer;
};

};

// Class Engine.MultiCueSplineAudioComponent
// 0x0014 (0x0320 - 0x0334)
class UMultiCueSplineAudioComponent : public USplineAudioComponent
{
public:
TArray<struct FMultiCueSplineSoundSlot>          SoundSlots;                              // 0x0320
(0x0010) [0x0000000000500001] (CPF_Edit | CPF_NeedCtorLink)
```

```
int32_t                               CurrentSlotIndex;                         // 0x0330 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MultiCueSplineAudioComponent");
}

return uClassPointer;
};

};

// Class Engine.SimpleSplineAudioComponent
// 0x0038 (0x0320 - 0x0358)
class USimpleSplineAudioComponent : public USplineAudioComponent
{
public:
unsigned long                          bAttenuateWithLPF : 1;                    // 0x0320 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                 LPFRadiusMin;                             // 0x0324 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 LPFRadiusMax;                             // 0x0328 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 dBAttenuationAtMax;                       // 0x032C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 FlattenAttenuationRadius;                 // 0x0330 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                               DistanceAlgorithm;                        // 0x0334 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                 RadiusMin;                                // 0x0338 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 RadiusMax;                                // 0x033C (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FSplineSoundSlot>                SoundSlots;                              // 0x0340 (0x0010)
[0x0000000000500001] (CPF_Edit | CPF_NeedCtorLink)
class USoundNode*                            NotifyBufferFinishedHook;                 // 0x0350
(0x0008) [0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SimpleSplineAudioComponent");
}
return uClassPointer;
};

};

// Class Engine.SimpleSplineNonLoopAudioComponent
// 0x0028 (0x0358 - 0x0380)
class USimpleSplineNonLoopAudioComponent : public USimpleSplineAudioComponent
{
public:
float                          DelayMin;                      // 0x0358 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DelayMax;                      // 0x035C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          PitchMin;                      // 0x0360 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          PitchMax;                      // 0x0364 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          VolumeMin;                     // 0x0368 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          VolumeMax;                     // 0x036C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                        CurrentSlotIndex;              // 0x0370 (0x0004)
[0x0000000000000000]
float                          UsedVolumeModulation;          // 0x0374 (0x0004)
[0x0000000000000000]
float                          UsedPitchModulation;           // 0x0378 (0x0004)
[0x0000000000000000]
float                          NextSoundTime;                 // 0x037C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SimpleSplineNonLoopAudioComponent");
}

return uClassPointer;
};

};

// Class Engine.GroupComponent_ORS
// 0x0053 (0x009D - 0x00F0)
class UGroupComponent_ORS : public UActorComponent
{
public:
TArray<struct FComponentTemplate>          Components;                          // 0x00A0
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<struct FComponentTemplate>          AllComponents;                       // 0x00B0
```

```
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class UObject*>                    AllObjects;                        // 0x00C0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
unsigned long                             bGlobalGroup : 1;                  // 0x00D0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                             bInitializedComponents : 1;        // 0x00D0 (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                             bRegisteredWithORS : 1;            // 0x00D0 (0x0004)
[0x0000000000000002] [0x00000004] (CPF_Const)
struct FScriptDelegate                    __ComponentCondition__Delegate;          // 0x00D8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GroupComponent_ORS");
}

return uClassPointer;
};

static void DiffComponents(TArray<struct FComponentTemplate>& SetA, TArray<struct
FComponentTemplate>& SetB, TArray<struct FComponentTemplate>& OnlyInSetA, TArray<struct
FComponentTemplate>& CommonToBothSets, TArray<struct FComponentTemplate>&
OnlyInSetB);
void SetParent(class UObject* NewParentGroup);
void RemoveComponents(TArray<struct FComponentTemplate>& ComponentsToRemove);
void AddComponents(TArray<struct FComponentTemplate>& ComponentsToAdd);
void RemoveComponent(struct FComponentTemplate ComponentToRemove);
void AddComponent(struct FComponentTemplate ComponentToAdd);
void RemoveClassDefaultObject(class UClass* ClassToRemove);
void AddClassDefaultObject(class UClass* ClassToAdd);
void RemoveObject(class UObject* ObjectToRemove);
void AddObject(class UObject* ObjectToAdd);
void UnregisterWithORS();
void RegisterWithORS();
bool ComponentCondition();
};

// Class Engine.HeightFogComponent
// 0x001F (0x009D - 0x00BC)
class UHeightFogComponent : public UActorComponent
{
public:
unsigned long                             bEnabled : 1;                      // 0x00A0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
float                                Height;                          // 0x00A4 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                Density;                         // 0x00A8 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
```

```
float                              LightBrightness;                        // 0x00AC (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
struct FColor                           LightColor;                          // 0x00B0 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                              ExtinctionDistance;                     // 0x00B4 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                              StartDistance;                        // 0x00B8 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HeightFogComponent");
}

return uClassPointer;
};

void SetEnabled(unsigned long bSetEnabled);
};

// Class Engine.PrimitiveComponent
// 0x01BB (0x009D - 0x0258)
class UPrimitiveComponent : public UActorComponent
{
public:
int32_t                              Tag;                                // 0x00A0 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FBoxSphereBounds                      Bounds;                            // 0x00A4 (0x001C)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                         SceneInfo;                          // 0x00C0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                              DetachFence;                         // 0x00C8 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                              LocalToWorldDeterminant;                  // 0x00CC (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FMatrix                          LocalToWorld;                        // 0x00D0 (0x0040)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                              MotionBlurInfoIndex;                    // 0x0110 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                    DecalList;                          // 0x0118 (0x0010)
[0x0000000001001002] (CPF_Const | CPF_Native)
TArray<class UDecalComponent*>                  DecalsToReattach;                         // 0x0128
(0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
class UPrimitiveComponent*                  ShadowParent;                         // 0x0138 (0x0008)
[0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component | CPF_EditInline)
class UPrimitiveComponent*                  ReplacementPrimitive;                    // 0x0140
(0x0008) [0x0000100004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
```

```
class UFogVolumeDensityComponent*              FogVolumeComponent;                      //
0x0148 (0x0008) [0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)
class ULightComponent*                    OverrideLightComponent;                  // 0x0150
(0x0008) [0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)
class ULightEnvironmentComponent*              LightEnvironment;                      // 0x0158
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class ULightEnvironmentComponent*              PreviousLightEnvironment;                // 0x0160
(0x0008) [0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)
float                                MinDrawDistance;                        // 0x0168 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                MassiveLODDistance;                      // 0x016C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                MaxDrawDistance;                        // 0x0170 (0x0004)
[0x0000000000800003] (CPF_Edit | CPF_Const | CPF_NoExport)
float                                CachedMaxDrawDistance;                    // 0x0174 (0x0004)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
float                                MotionBlurInstanceScale;                  // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                CullDistance;                          // 0x017C (0x0004)
[0x0000000020800002] (CPF_Const | CPF_NoExport | CPF_Deprecated)
float                                CachedCullDistance;                      // 0x0180 (0x0004)
[0x0000000020020000] (CPF_EditConst | CPF_Deprecated)
uint8_t                               DepthPriorityGroup;                      // 0x0184 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                               ViewOwnerDepthPriorityGroup;               // 0x0185 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                               DetailMode;                           // 0x0186 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                               RBChannel;                            // 0x0187 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                               RBDominanceGroup;                        // 0x0188 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                               PreviewEnvironmentShadowing;               // 0x0189 (0x0001)
[0x0000000000000000]
unsigned long                            bUseViewOwnerDepthPriorityGroup : 1;          // 0x018C
(0x0004) [0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                            bOnlyBlockActorMovement : 1;               // 0x018C (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                            bAllowCullDistanceVolume : 1;              // 0x018C (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                            HiddenGame : 1;                        // 0x018C (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                            HiddenEditor : 1;                       // 0x018C (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                            bOwnerNoSee : 1;                        // 0x018C (0x0004)
[0x0000000000000003] [0x00000020] (CPF_Edit | CPF_Const)
unsigned long                            bOnlyOwnerSee : 1;                      // 0x018C (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                            bIgnoreOwnerHidden : 1;                   // 0x018C (0x0004)
[0x0000000000000003] [0x00000080] (CPF_Edit | CPF_Const)
```

```
unsigned long                    bUseAsOccluder : 1;                  // 0x018C (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                    bAllowApproximateOcclusion : 1;           // 0x018C (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                    bFirstFrameOcclusion : 1;              // 0x018C (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                    bIgnoreNearPlaneIntersection : 1;          // 0x018C (0x0004)
[0x0000000000000000] [0x00000800]
unsigned long                    bSelectable : 1;                  // 0x018C (0x0004)
[0x0000000000000000] [0x00001000]
unsigned long                    bForceMipStreaming : 1;               // 0x018C (0x0004)
[0x0000000000000003] [0x00002000] (CPF_Edit | CPF_Const)
unsigned long                    bAcceptsStaticDecals : 1;              // 0x018C (0x0004)
[0x0000000000000003] [0x00004000] (CPF_Edit | CPF_Const)
unsigned long                    bAcceptsDynamicDecals : 1;              // 0x018C (0x0004)
[0x0000000000000003] [0x00008000] (CPF_Edit | CPF_Const)
unsigned long                    bIsRefreshingDecals : 1;               // 0x018C (0x0004)
[0x0000000000003002] [0x00010000] (CPF_Const | CPF_Native | CPF_Transient)
unsigned long                    bAllowDecalAutomaticReAttach : 1;          // 0x018C
(0x0004) [0x0000000000002000] [0x00020000] (CPF_Transient)
unsigned long                    bUsePerInstanceHitProxies : 1;            // 0x018C (0x0004)
[0x0000000000000000] [0x00040000]
unsigned long                    CastShadow : 1;                   // 0x018C (0x0004)
[0x0000000000000001] [0x00080000] (CPF_Edit)
unsigned long                    bForceDirectLightMap : 1;              // 0x018C (0x0004)
[0x0000000000000002] [0x00100000] (CPF_Const)
unsigned long                    bCastDynamicShadow : 1;               // 0x018C (0x0004)
[0x0000000000000001] [0x00200000] (CPF_Edit)
unsigned long                    bCastStaticShadow : 1;               // 0x018C (0x0004)
[0x0000000000000001] [0x00400000] (CPF_Edit)
unsigned long                    bSelfShadowOnly : 1;                // 0x018C (0x0004)
[0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                    bNoModSelfShadow : 1;                // 0x018C (0x0004)
[0x0000000000000001] [0x01000000] (CPF_Edit)
unsigned long                    bAcceptsDynamicDominantLightShadows : 1;     // 0x018C
(0x0004) [0x0000000000000001] [0x02000000] (CPF_Edit)
unsigned long                    bCastHiddenShadow : 1;               // 0x018C (0x0004)
[0x0000000000000001] [0x04000000] (CPF_Edit)
unsigned long                    bCastShadowAsTwoSided : 1;              // 0x018C (0x0004)
[0x0000000000000001] [0x08000000] (CPF_Edit)
unsigned long                    bAcceptsLights : 1;                 // 0x018C (0x0004)
[0x0000000000000003] [0x10000000] (CPF_Edit | CPF_Const)
unsigned long                    bAcceptsDynamicLights : 1;              // 0x018C (0x0004)
[0x0000000000000003] [0x20000000] (CPF_Edit | CPF_Const)
unsigned long                    bUseOnePassLightingOnTranslucency : 1;       // 0x018C
(0x0004) [0x0000000000000003] [0x40000000] (CPF_Edit | CPF_Const)
unsigned long                    bUsePrecomputedShadows : 1;             // 0x018C
(0x0004) [0x0000000000000003] [0x80000000] (CPF_Edit | CPF_Const)
unsigned long                    bHasExplicitShadowParent : 1;            // 0x0190 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                    bAllowAmbientOcclusion : 1;             // 0x0190 (0x0004)
[0x0000000020000000] [0x00000002]  CPF_Deprecated)
unsigned long                    CollideActors : 1;                 // 0x0190 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
```

```
unsigned long                          AlwaysCheckCollision : 1;                   // 0x0190 (0x0004)
[0x0000000000000002] [0x00000008] (CPF_Const)
unsigned long                          BlockActors : 1;                      // 0x0190 (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                          BlockZeroExtent : 1;                   // 0x0190 (0x0004)
[0x0000000000000003] [0x00000020] (CPF_Edit | CPF_Const)
unsigned long                          BlockNonZeroExtent : 1;                  // 0x0190 (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                          CanBlockCamera : 1;                    // 0x0190 (0x0004)
[0x0000000000000003] [0x00000080] (CPF_Edit | CPF_Const)
unsigned long                          BlockRigidBody : 1;                   // 0x0190 (0x0004)
[0x0000000000000003] [0x00000100] (CPF_Edit | CPF_Const)
unsigned long                          bBlockFootPlacement : 1;                 // 0x0190 (0x0004)
[0x0000000000000003] [0x00000200] (CPF_Edit | CPF_Const)
unsigned long                          bDisableAllRigidBody : 1;                // 0x0190 (0x0004)
[0x0000000000000003] [0x00000400] (CPF_Edit | CPF_Const)
unsigned long                          bSkipRBGeomCreation : 1;                 // 0x0190 (0x0004)
[0x0000000000000003] [0x00000800] (CPF_Edit | CPF_Const)
unsigned long                          bNotifyRigidBodyCollision : 1;             // 0x0190 (0x0004)
[0x0000000000000003] [0x00001000] (CPF_Edit | CPF_Const)
unsigned long                          bFluidDrain : 1;                     // 0x0190 (0x0004)
[0x0000000000000003] [0x00002000] (CPF_Edit | CPF_Const)
unsigned long                          bFluidTwoWay : 1;                     // 0x0190 (0x0004)
[0x0000000000000003] [0x00004000] (CPF_Edit | CPF_Const)
unsigned long                          bIgnoreRadialImpulse : 1;                // 0x0190 (0x0004)
[0x0000000000000001] [0x00008000] (CPF_Edit)
unsigned long                          bIgnoreRadialForce : 1;                 // 0x0190 (0x0004)
[0x0000000000000001] [0x00010000] (CPF_Edit)
unsigned long                          bIgnoreForceField : 1;                  // 0x0190 (0x0004)
[0x0000000000000001] [0x00020000] (CPF_Edit)
unsigned long                          bUseCompartment : 1;                   // 0x0190 (0x0004)
[0x0000000000000003] [0x00040000] (CPF_Edit | CPF_Const)
unsigned long                          AlwaysLoadOnClient : 1;                 // 0x0190 (0x0004)
[0x0000000000000002] [0x00080000] (CPF_Const)
unsigned long                          AlwaysLoadOnServer : 1;                  // 0x0190 (0x0004)
[0x0000000000000002] [0x00100000] (CPF_Const)
unsigned long                          bIgnoreHiddenActorsMembership : 1;          // 0x0190
(0x0004) [0x0000000000000001] [0x00200000] (CPF_Edit)
unsigned long                          AbsoluteTranslation : 1;                // 0x0190 (0x0004)
[0x0000000000000003] [0x00400000] (CPF_Edit | CPF_Const)
unsigned long                          AbsoluteRotation : 1;                  // 0x0190 (0x0004)
[0x0000000000000003] [0x00800000] (CPF_Edit | CPF_Const)
unsigned long                          AbsoluteScale : 1;                    // 0x0190 (0x0004)
[0x0000000000000003] [0x01000000] (CPF_Edit | CPF_Const)
unsigned long                          bAllowShadowFade : 1;                   // 0x0190 (0x0004)
[0x0000000000000000] [0x02000000]
unsigned long                          bSupportedOnMobile : 1;                  // 0x0190 (0x0004)
[0x0000000000000000] [0x04000000]
unsigned long                          bWasSNFiltered : 1;                   // 0x0190 (0x0004)
[0x0000000000003002] [0x08000000] (CPF_Const | CPF_Native | CPF_Transient)
TArray<int32_t>                         OctreeNodes;                       // 0x0198 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<uint8_t>                         AlwaysShowInSelectedPlatforms;            // 0x01A8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
```

```
int32_t                              TranslucencySortPriority;                  // 0x01B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                              VisibilityId;                              // 0x01BC (0x0004)
[0x0000000000200000]
struct FLightingChannelContainer               LightingChannels;                          // 0x01C0
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                                bHideInLowEffectsIntensity : 1;            // 0x01C4 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
struct FRBCollisionChannelContainer             RBCollideWithChannels;                     // 0x01C8
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
class UPhysicalMaterial*                        PhysMaterialOverride;                      // 0x01D0 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class URB_BodyInstance*                          BodyInstance;                              // 0x01D8 (0x0008)
[0x0000000000201002] (CPF_Const | CPF_Native)
struct FMatrix                                  CachedParentToWorld;                       // 0x01E0 (0x0040)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FVector                                  Translation;                               // 0x0220 (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FRotator                                 Rotation;                                  // 0x022C (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                                          Scale;                                     // 0x0238 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FVector                                  Scale3D;                                   // 0x023C (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                                          BoundsScale;                               // 0x0248 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                                          LastSubmitTime;                            // 0x024C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                          LastRenderTime;                            // 0x0250 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                          ScriptRigidBodyCollisionThreshold;         // 0x0254 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrimitiveComponent");
}

return uClassPointer;
};

void SetCastShadow(unsigned long bNewCastShadow, unsigned long
bNewCastDynamicShadow);
uint8_t ClosestPointOnComponentToComponent(class UPrimitiveComponent*&
OtherComponent, struct FVector& PointOnComponentA, struct FVector& PointOnComponentB);
uint8_t ClosestPointOnComponentToPoint(struct FVector& POI, struct FVector& Extent, struct
FVector& OutPointA, struct FVector& OutPointB);
struct FRotator GetRotation();
struct FVector GetPosition();
```

```
void SetAbsolute(unsigned long NewAbsoluteTranslation, unsigned long NewAbsoluteRotation,
unsigned long NewAbsoluteScale);
void SetScale3D(struct FVector NewScale3D);
void SetScale(float NewScale);
void SetRotation(struct FRotator NewRotation);
void SetTranslation(struct FVector NewTranslation);
void SetActorCollision(unsigned long NewCollideActors, unsigned long NewBlockActors,
unsigned long NewAlwaysCheckCollision);
void SetTraceBlocking(unsigned long NewBlockZeroExtent, unsigned long
NewBlockNonZeroExtent);
void SetViewOwnerDepthPriorityGroup(unsigned long bNewUseViewOwnerDepthPriorityGroup,
uint8_t NewViewOwnerDepthPriorityGroup);
void SetDepthPriorityGroup(uint8_t NewDepthPriorityGroup);
void SetLightingChannels(struct FLightingChannelContainer NewLightingChannels);
void SetCullDistance(float NewCullDistance);
void SetLightEnvironment(class ULightEnvironmentComponent* NewLightEnvironment);
void SetShadowParent(class UPrimitiveComponent* NewShadowParent);
void SetIgnoreOwnerHidden(unsigned long bNewIgnoreOwnerHidden);
void SetOnlyOwnerSee(unsigned long bNewOnlyOwnerSee);
void SetOwnerNoSee(unsigned long bNewOwnerNoSee);
void SetHidden(unsigned long NewHidden);
bool ShouldComponentAddToScene();
void SetRBDominanceGroup(uint8_t InDomGroup);
class URB_BodyInstance* GetRootBodyInstance();
void SetPhysMaterialOverride(class UPhysicalMaterial* NewPhysMaterial);
void TermRBPhys();
void InitRBPhys();
void SetNotifyRigidBodyCollision(unsigned long bNewNotifyRigidBodyCollision);
void SetRBChannel(uint8_t Channel);
void SetRBCollisionChannels(struct FRBCollisionChannelContainer Channels);
void SetRBCollidesWithChannel(uint8_t Channel, unsigned long bNewCollides);
void SetBlockRigidBody(unsigned long bNewBlockRigidBody);
bool RigidBodyIsAwake(struct FName BoneName);
void PutRigidBodyToSleep(struct FName BoneName);
void WakeRigidBody(struct FName BoneName);
void SetRBQuat(struct FName BoneName, struct FQuat& NewQuat);
void SetRBRotation(struct FRotator NewRot, struct FName BoneName);
void SetRBPosition(struct FVector NewPos, struct FName BoneName);
void RetardRBLinearVelocity(struct FVector RetardDir, float VelScale);
void SetRBAngularVelocity(struct FVector NewAngVel, unsigned long bAddToCurrent);
void SetRBLinearVelocity(struct FVector NewVel, unsigned long bAddToCurrent);
void AddTorque(struct FVector Torque, struct FName BoneName, uint8_t ForceMode);
void AddRadialForce(struct FVector Origin, float Radius, float Strength, uint8_t Falloff);
float GetMass(struct FName BoneName);
void AddForce(struct FVector Force, struct FVector Position, struct FName BoneName, uint8_t
ForceMode);
void AddRadialImpulse(struct FVector Origin, float Radius, float Strength, uint8_t Falloff,
unsigned long bVelChange);
void AddImpulse(struct FVector Impulse, struct FVector Position, struct FName BoneName,
unsigned long bVelChange);
};

// Class Engine.ArrowComponent
// 0x0014 (0x0258 - 0x026C)
```

```
class UArrowComponent : public UPrimitiveComponent
{
public:
struct FColor                          ArrowColor;                            // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  ArrowSize;                             // 0x025C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bTreatAsASprite : 1;                   // 0x0260 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FName                           SpriteCategoryName;                    // 0x0264 (0x0008)
[0x0000000800000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ArrowComponent");
}

return uClassPointer;
};

};

// Class Engine.BrushComponent
// 0x0078 (0x0258 - 0x02D0)
class UBrushComponent : public UPrimitiveComponent
{
public:
class UModel*                          Brush;                                 // 0x0258 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FKAggregateGeom                 BrushAggGeom;                          // 0x0260 (0x0050)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FPointer                        BrushPhysDesc;                         // 0x02B0 (0x0008)
[0x0000000001003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FKCachedConvexData_Mirror       CachedPhysBrushData;                   // 0x02B8
(0x0010) [0x0000000001003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                                CachedPhysBrushDataVersion;            // 0x02C8 (0x0004)
[0x0000000000000002] (CPF_Const)
unsigned long                          bBlockComplexCollisionTrace : 1;       // 0x02CC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.BrushComponent");
}
```

```
return uClassPointer;
};

};

// Class Engine.CameraConeComponent
// 0x0000 (0x0258 - 0x0258)
class UCameraConeComponent : public UPrimitiveComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CameraConeComponent");
}

return uClassPointer;
};

};

// Class Engine.CylinderComponent
// 0x0010 (0x0258 - 0x0268)
class UCylinderComponent : public UPrimitiveComponent
{
public:
float                                    CollisionHeight;                         // 0x0258 (0x0004)
[0x000000000000000B] (CPF_Edit | CPF_Const | CPF_ExportObject)
float                                    CollisionRadius;                         // 0x025C (0x0004)
[0x000000000000000B] (CPF_Edit | CPF_Const | CPF_ExportObject)
struct FColor                                 CylinderColor;                           // 0x0260 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                                 bDrawBoundingBox : 1;                    // 0x0264 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                                 bDrawNonColliding : 1;                   // 0x0264 (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                                 bAlwaysRenderIfSelected : 1;             // 0x0264 (0x0004)
[0x0000000000000002] [0x00000004] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CylinderComponent");
}
```

```
return uClassPointer;
};

struct FVector GetExtent();
void SetCylinderSize(float NewRadius, float NewHeight);
};

// Class Engine.DrawBoxComponent
// 0x0020 (0x0258 - 0x0278)
class UDrawBoxComponent : public UPrimitiveComponent
{
public:
struct FColor                    BoxColor;                      // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterial*                  BoxMaterial;                   // 0x0260 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                    BoxExtent;                     // 0x0268 (0x000C)
[0x0000000000000001] (CPF_Edit)
unsigned long                     bDrawWireBox : 1;              // 0x0274 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                     bDrawLitBox : 1;               // 0x0274 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                     bDrawOnlyIfSelected : 1;           // 0x0274 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawBoxComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawCapsuleComponent
// 0x001C (0x0258 - 0x0274)
class UDrawCapsuleComponent : public UPrimitiveComponent
{
public:
struct FColor                    CapsuleColor;                  // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterial*                  CapsuleMaterial;               // 0x0260 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                         CapsuleHeight;                 // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         CapsuleRadius;                 // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
unsigned long                          bDrawWireCapsule : 1;                       // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bDrawLitCapsule : 1;                       // 0x0270 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bDrawOnlyIfSelected : 1;                   // 0x0270 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawCapsuleComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawConeComponent
// 0x0010 (0x0258 - 0x0268)
class UDrawConeComponent : public UPrimitiveComponent
{
public:
struct FColor                          ConeColor;                                 // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  ConeRadius;                                // 0x025C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  ConeAngle;                                 // 0x0260 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                ConeSides;                                 // 0x0264 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawConeComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawCylinderComponent
// 0x0028 (0x0258 - 0x0280)
class UDrawCylinderComponent : public UPrimitiveComponent
```

```
{
public:
struct FColor                              CylinderColor;                              // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterial*                           CylinderMaterial;                           // 0x0260 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                      CylinderRadius;                             // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      CylinderTopRadius;                          // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      CylinderHeight;                             // 0x0270 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      CylinderHeightOffset;                       // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                    CylinderSides;                              // 0x0278 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                              bDrawWireCylinder : 1;                      // 0x027C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                              bDrawLitCylinder : 1;                       // 0x027C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                              bDrawOnlyIfSelected : 1;                    // 0x027C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawCylinderComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawFrustumComponent
// 0x0020 (0x0258 - 0x0278)
class UDrawFrustumComponent : public UPrimitiveComponent
{
public:
struct FColor                              FrustumColor;                               // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      FrustumAngle;                               // 0x025C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      FrustumAspectRatio;                         // 0x0260 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      FrustumStartDist;                           // 0x0264 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      FrustumEndDist;                             // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture*                            Texture;                                    // 0x0270 (0x0008)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawFrustumComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawQuadComponent
// 0x0010 (0x0258 - 0x0268)
class UDrawQuadComponent : public UPrimitiveComponent
{
public:
class UTexture*                              Texture;                              // 0x0258 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                    Width;                              // 0x0260 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    Height;                              // 0x0264 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawQuadComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawSphereComponent
// 0x001C (0x0258 - 0x0274)
class UDrawSphereComponent : public UPrimitiveComponent
{
public:
struct FColor                              SphereColor;                              // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterial*                              SphereMaterial;                              // 0x0260 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                    SphereRadius;                              // 0x0268 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
int32_t                                  SphereSides;                              // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bDrawWireSphere : 1;                      // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bDrawLitSphere : 1;                       // 0x0270 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                            bDrawOnlyIfSelected : 1;                  // 0x0270 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawSphereComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawPylonRadiusComponent
// 0x0004 (0x0274 - 0x0278)
class UDrawPylonRadiusComponent : public UDrawSphereComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawPylonRadiusComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawSoundRadiusComponent
// 0x0004 (0x0274 - 0x0278)
class UDrawSoundRadiusComponent : public UDrawSphereComponent
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawSoundRadiusComponent");
}

return uClassPointer;
};

};

// Class Engine.LevelGridVolumeRenderingComponent
// 0x0000 (0x0258 - 0x0258)
class ULevelGridVolumeRenderingComponent : public UPrimitiveComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelGridVolumeRenderingComponent");
}

return uClassPointer;
};

};

// Class Engine.LineBatchComponent
// 0x0034 (0x0258 - 0x028C)
class ULineBatchComponent : public UPrimitiveComponent
{
public:
struct FPointer                    FPrimitiveDrawInterfaceVfTable;           // 0x0258 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                    FPrimitiveDrawInterfaceView;              // 0x0260 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
TArray<struct FPointer>                 BatchedLines;                        // 0x0268 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                 BatchedPoints;                       // 0x0278 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
float                              DefaultLifeTime;                    // 0x0288 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LineBatchComponent");
}

return uClassPointer;
};

};

// Class Engine.ModelComponent
// 0x0030 (0x0258 - 0x0288)
class UModelComponent : public UPrimitiveComponent
{
public:
class UObject*                          Model;                              // 0x0258 (0x0008)
[0x0000000000803002] (CPF_Const | CPF_Native | CPF_Transient | CPF_NoExport)
int32_t                                 ZoneIndex;                          // 0x0260 (0x0004)
[0x0000000000803002] (CPF_Const | CPF_Native | CPF_Transient | CPF_NoExport)
int32_t                                 ComponentIndex;                         // 0x0264 (0x0004)
[0x0000000000803002] (CPF_Const | CPF_Native | CPF_Transient | CPF_NoExport)
TArray<struct FPointer>                     Nodes;                              // 0x0268 (0x0010)
[0x0000000000803002] (CPF_Const | CPF_Native | CPF_Transient | CPF_NoExport)
TArray<struct FPointer>                     Elements;                             // 0x0278 (0x0010)
[0x0000000000803002] (CPF_Const | CPF_Native | CPF_Transient | CPF_NoExport)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ModelComponent");
}

return uClassPointer;
};

};

// Class Engine.SpriteComponent
// 0x0028 (0x0258 - 0x0280)
class USpriteComponent : public UPrimitiveComponent
{
public:
class UTexture2D*                         Sprite;                             // 0x0258 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bIsScreenSizeScaled : 1;                  // 0x0260 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                   ScreenSize;                         // 0x0264 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   U;                              // 0x0268 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                          UL;                            // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          V;                             // 0x0270 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          VL;                            // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                   SpriteCategoryName;            // 0x0278 (0x0008)
[0x0000000800000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpriteComponent");
}

return uClassPointer;
};

void SetSpriteAndUV(class UTexture2D* NewSprite, int32_t NewU, int32_t NewUL, int32_t NewV,
int32_t NewVL);
void SetUV(int32_t NewU, int32_t NewUL, int32_t NewV, int32_t NewVL);
void SetSprite(class UTexture2D* NewSprite);
};

// Class Engine.RadialBlurComponent
// 0x0073 (0x009D - 0x0110)
class URadialBlurComponent : public UActorComponent
{
public:
class UMaterialInterface*      Material;                      // 0x00A0 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                        DepthPriorityGroup;            // 0x00A8 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          BlurScale;                     // 0x00AC (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                          BlurFalloffExponent;           // 0x00B0 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                          BlurOpacity;                   // 0x00B4 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                          MaxCullDistance;               // 0x00B8 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          DistanceFalloffExponent;       // 0x00BC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                  bRenderAsVelocity : 1;         // 0x00C0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                  bEnabled : 1;                  // 0x00C0 (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
uint8_t                        UnknownData00[0xC];            // 0x00C4 (0x000C) MISSED
OFFSET
```

```
struct FMatrix                          LocalToWorld;                       // 0x00D0 (0x0040)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RadialBlurComponent");
}

return uClassPointer;
};

void OnUpdatePropertyBlurOpacity();
void OnUpdatePropertyBlurFalloffExponent();
void OnUpdatePropertyBlurScale();
void SetEnabled(unsigned long bInEnabled);
void SetBlurOpacity(float InBlurOpacity);
void SetBlurFalloffExponent(float InBlurFalloffExponent);
void SetBlurScale(float InBlurScale);
void SetMaterial(class UMaterialInterface* InMaterial);
};

// Class Engine.SceneCaptureComponent
// 0x0063 (0x009D - 0x0100)
class USceneCaptureComponent : public UActorComponent
{
public:
float                                   MaxCaptureTime;                     // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   CaptureTimeRemaining;                // 0x00A4 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                           bEnabled : 1;                       // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bEnablePostProcess : 1;                 // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                           bEnableFog : 1;                      // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           bUseMainScenePostProcessSettings : 1;       // 0x00A8
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           bSkipUpdateIfTextureUsersOccluded : 1;       // 0x00A8
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                           bSkipUpdateIfOwnerOccluded : 1;            // 0x00A8
(0x0004) [0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                           bSkipRenderingDepthPrepass : 1;            // 0x00A8
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
struct FColor                           ClearColor;                         // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 ViewMode;                           // 0x00B0 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 PostMethod;                         // 0x00B1 (0x0001)
```

```
[0x0000000000000001] (CPF_Edit)
int32_t                          SceneLOD;                            // 0x00B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          CubemapDesaturationAmount;                // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            CubemapSeamlessRoughness;                 // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            FrameRate;                           // 0x00C0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UPostProcessChain*                    PostProcess;                          // 0x00C8 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            MaxUpdateDist;                         // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            MaxViewDistanceOverride;                   // 0x00D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            MaxStreamingUpdateDist;                    // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                         CaptureInfo;                          // 0x00E0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                         ViewState;                            // 0x00E8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                      PostProcessProxies;                     // 0x00F0 (0x0010)
[0x0000000001203002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCaptureComponent");
}

return uClassPointer;
};

void ForceRecapture();
void SetEnabled(unsigned long bEnable);
void SetFrameRate(float NewFrameRate);
};

// Class Engine.SceneCapture2DComponent
// 0x00A0 (0x0100 - 0x01A0)
class USceneCapture2DComponent : public USceneCaptureComponent
{
public:
class UTextureRenderTarget2D*                 TextureTarget;                         // 0x0100 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            FieldOfView;                          // 0x0108 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            NearPlane;                           // 0x010C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            FarPlane;                           // 0x0110 (0x0004)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                         bUpdateMatrices : 1;                     // 0x0114 (0x0004)
[0x0000000000000000] [0x00000001]
uint8_t                               UnknownData00[0x8];                      // 0x0118 (0x0008) MISSED
OFFSET
struct FMatrix                        ViewMatrix;                              // 0x0120 (0x0040)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FMatrix                        ProjMatrix;                              // 0x0160 (0x0040)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCapture2DComponent");
}

return uClassPointer;
};

void SetView(struct FVector NewLocation, struct FRotator NewRotation);
void SetCaptureParameters(class UTextureRenderTarget2D* NewTextureTarget, float NewFOV,
float NewNearPlane, float NewFarPlane);
};

// Class Engine.SceneCapture2DHitMaskComponent
// 0x002C (0x0100 - 0x012C)
class USceneCapture2DHitMaskComponent : public USceneCaptureComponent
{
public:
class UTextureRenderTarget2D*         TextureTarget;                           // 0x0100 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class USkeletalMeshComponent*         SkeletalMeshComp;                        // 0x0108
(0x0008) [0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)
int32_t                               MaterialIndex;                           // 0x0110 (0x0004)
[0x0000000000000000]
int32_t                               ForceLOD;                                // 0x0114 (0x0004)
[0x0000000000000000]
int32_t                               HitMaskCullDistance;                     // 0x0118 (0x0004)
[0x0000000000000000]
float                                 FadingStartTimeSinceHit;                 // 0x011C (0x0004)
[0x0000000000000000]
float                                 FadingPercentage;                        // 0x0120 (0x0004)
[0x0000000000000000]
float                                 FadingDurationTime;                      // 0x0124 (0x0004)
[0x0000000000000000]
float                                 FadingIntervalTime;                      // 0x0128 (0x0004)
[0x0000000000000000]

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCapture2DHitMaskComponent");
}

return uClassPointer;
};

void SetFadingStartTimeSinceHit(float InFadingStartTimeSinceHit);
void SetCaptureParameters(struct FVector InMaskPosition, float InMaskRadius, struct FVector
InStartupPosition, unsigned long bOnlyWhenFacing);
void SetCaptureTargetTexture(class UTextureRenderTarget2D* InTextureTarget);
};

// Class Engine.SceneCaptureCubeMapComponent
// 0x001C (0x0100 - 0x011C)
class USceneCaptureCubeMapComponent : public USceneCaptureComponent
{
public:
class UTextureRenderTargetCube*          TextureTarget;                   // 0x0100
(0x0008) [0x0000000000000001] (CPF_Edit)
float                          NearPlane;                   // 0x0108 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FarPlane;                   // 0x010C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                     WorldLocation;                // 0x0110 (0x000C)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCaptureCubeMapComponent");
}

return uClassPointer;
};

};

// Class Engine.SceneCapturePortalComponent
// 0x0018 (0x0100 - 0x0118)
class USceneCapturePortalComponent : public USceneCaptureComponent
{
public:
class UTextureRenderTarget2D*          TextureTarget;                   // 0x0100 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
```

```
float                               ScaleFOV;                           // 0x0108 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class AActor*                       ViewDestination;                    // 0x0110 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCapturePortalComponent");
}

return uClassPointer;
};

void SetCaptureParameters(class UTextureRenderTarget2D* NewTextureTarget, float
NewScaleFOV, class AActor* NewViewDest);
};

// Class Engine.SceneCaptureReflectComponent
// 0x000C (0x0100 - 0x010C)
class USceneCaptureReflectComponent : public USceneCaptureComponent
{
public:
class UTextureRenderTarget2D*       TextureTarget;                      // 0x0100 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                               ScaleFOV;                           // 0x0108 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SceneCaptureReflectComponent");
}

return uClassPointer;
};

};

// Class Engine.WindDirectionalSourceComponent
// 0x001B (0x009D - 0x00B8)
class UWindDirectionalSourceComponent : public UActorComponent
{
public:
struct FPointer                     SceneProxy;                         // 0x00A0 (0x0008)
[0x0000000001003002] (CPF_Const | CPF_Native | CPF_Transient)
```

```
float                              Strength;                        // 0x00A8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                              Phase;                           // 0x00AC (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                              Frequency;                         // 0x00B0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                              Speed;                           // 0x00B4 (0x0004)
[0x0000000200000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WindDirectionalSourceComponent");
}

return uClassPointer;
};

};

// Class Engine.WindPointSourceComponent
// 0x000C (0x00B8 - 0x00C4)
class UWindPointSourceComponent : public UWindDirectionalSourceComponent
{
public:
class UDrawSphereComponent*              PreviewRadiusComponent;               // 0x00B8
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
float                              Radius;                          // 0x00C0 (0x0004)
[0x0000000200000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WindPointSourceComponent");
}

return uClassPointer;
};

};

// Class Engine.ActorFactory
// 0x003C (0x0060 - 0x009C)
class UActorFactory : public UObject
{
```

```
public:
class UClass*                          GameplayActorClass;                     // 0x0060 (0x0008)
[0x0000000000000000]
class FString                          MenuName;                               // 0x0068 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                MenuPriority;                           // 0x0078 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                AlternateMenuPriority;                  // 0x007C (0x0004)
[0x0000000020000000]  CPF_Deprecated)
class FString                          NewActorClassName;                      // 0x0080 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                          NewActorClass;                          // 0x0090 (0x0008)
[0x0000000000000000]
unsigned long                          bPlaceable : 1;                         // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                          bShowInEditorQuickMenu : 1;             // 0x0098 (0x0004)
[0x0000000000000000] [0x00000002]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactory");
}

return uClassPointer;
};

void eventPostCreateActor(class AActor* NewActor, class USeqAct_ActorFactory*
ActorFactoryData);
};

// Class Engine.ActorFactoryActor
// 0x000C (0x009C - 0x00A8)
class UActorFactoryActor : public UActorFactory
{
public:
class UClass*                          ActorClass;                             // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryActor");
}

return uClassPointer;
```

```
};

};

// Class Engine.ActorFactoryAI
// 0x0028 (0x009C - 0x00C4)
class UActorFactoryAI : public UActorFactory
{
public:
class UClass*                          ControllerClass;                       // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UClass*                          PawnClass;                             // 0x00A8 (0x0008)
[0x0000000000000001] (CPF_Edit)
class FString                          PawnName;                              // 0x00B0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                                TeamIndex;                             // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryAI");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryAmbientSound
// 0x000C (0x009C - 0x00A8)
class UActorFactoryAmbientSound : public UActorFactory
{
public:
class USoundCue*                       AmbientSoundCue;                       // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryAmbientSound");
}

return uClassPointer;
};
```

```
};

// Class Engine.ActorFactoryAmbientSoundMovable
// 0x0000 (0x00A8 - 0x00A8)
class UActorFactoryAmbientSoundMovable : public UActorFactoryAmbientSound
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryAmbientSoundMovable");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryAmbientSoundSimple
// 0x000C (0x009C - 0x00A8)
class UActorFactoryAmbientSoundSimple : public UActorFactory
{
public:
class USoundNodeWave*                    SoundNodeWave;                         // 0x00A0
(0x0008) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryAmbientSoundSimple");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryAmbientSoundNonLoop
// 0x0000 (0x00A8 - 0x00A8)
class UActorFactoryAmbientSoundNonLoop : public UActorFactoryAmbientSoundSimple
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryAmbientSoundNonLoop");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryAmbientSoundSimpleToggleable
// 0x0000 (0x00A8 - 0x00A8)
class UActorFactoryAmbientSoundSimpleToggleable : public
UActorFactoryAmbientSoundSimple
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ActorFactoryAmbientSoundSimpleToggleable");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryAmbientSoundNonLoopingToggleable
// 0x0000 (0x00A8 - 0x00A8)
class UActorFactoryAmbientSoundNonLoopingToggleable : public
UActorFactoryAmbientSoundSimpleToggleable
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ActorFactoryAmbientSoundNonLoopingToggleable");
}
return uClassPointer;
};

};

// Class Engine.ActorFactoryApexDestructible
// 0x001C (0x009C - 0x00B8)
class UActorFactoryApexDestructible : public UActorFactory
{
public:
unsigned long                    bStartAwake : 1;                 // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
uint8_t                          RBChannel;                       // 0x00A4 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRBCollisionChannelContainer          CollideWithChannels;               // 0x00A8
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
class UApexDestructibleAsset*          DestructibleAsset;               // 0x00B0
(0x0008) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryApexDestructible");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryArchetype
// 0x000C (0x009C - 0x00A8)
class UActorFactoryArchetype : public UActorFactory
{
public:
class AActor*                    ArchetypeActor;                  // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryArchetype");
}

return uClassPointer;
};
```

```
};

// Class Engine.ActorFactoryCoverLink
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryCoverLink : public UActorFactory
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryCoverLink");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryDominantDirectionalLight
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryDominantDirectionalLight : public UActorFactory
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryDominantDirectionalLight");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryDominantDirectionalLightMovable
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryDominantDirectionalLightMovable : public UActorFactory
{
public:

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ActorFactoryDominantDirectionalLightMovable");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryDynamicSM
// 0x001D (0x009C - 0x00B9)
class UActorFactoryDynamicSM : public UActorFactory
{
public:
class UStaticMesh*                    StaticMesh;                     // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                      DrawScale3D;                     // 0x00A8 (0x000C)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bNoEncroachCheck : 1;                 // 0x00B4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bNotifyRigidBodyCollision : 1;             // 0x00B4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       bBlockRigidBody : 1;                 // 0x00B4 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                       bUseCompartment : 1;                 // 0x00B4 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                       bCastDynamicShadow : 1;                // 0x00B4 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
uint8_t                           CollisionType;                    // 0x00B8 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryDynamicSM");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryMover
// 0x0007 (0x00B9 - 0x00C0)
class UActorFactoryMover : public UActorFactoryDynamicSM
{
```

```
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryMover");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryRigidBody
// 0x0033 (0x00B9 - 0x00EC)
class UActorFactoryRigidBody : public UActorFactoryDynamicSM
{
public:
unsigned long                    bStartAwake : 1;                    // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bDamageAppliesImpulse : 1;          // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bLocalSpaceInitialVelocity : 1;     // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    bEnableStayUprightSpring : 1;       // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FVector                   InitialVelocity;                    // 0x00C4 (0x000C)
[0x0000000000000001] (CPF_Edit)
class UDistributionVector*            AdditionalVelocity;            // 0x00D0 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)
class UDistributionVector*            InitialAngularVelocity;        // 0x00D8 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)
uint8_t                          RBChannel;                          // 0x00E0 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                            StayUprightTorqueFactor;            // 0x00E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            StayUprightMaxTorque;               // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryRigidBody");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryEmitter
// 0x000C (0x009C - 0x00A8)
class UActorFactoryEmitter : public UActorFactory
{
public:
class UParticleSystem*                    ParticleSystem;                        // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryEmitter");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryFracturedStaticMesh
// 0x0018 (0x009C - 0x00B4)
class UActorFactoryFracturedStaticMesh : public UActorFactory
{
public:
class UFracturedStaticMesh*                FracturedStaticMesh;                  // 0x00A0
(0x0008) [0x0000000000000001] (CPF_Edit)
struct FVector                             DrawScale3D;                          // 0x00A8 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryFracturedStaticMesh");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryLensFlare
// 0x000C (0x009C - 0x00A8)
```

```
class UActorFactoryLensFlare : public UActorFactory
{
public:
class ULensFlare*                               LensFlareObject;                               // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryLensFlare");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryLight
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryLight : public UActorFactory
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryLight");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryPathNode
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryPathNode : public UActorFactory
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryPathNode");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryPhysicsAsset
// 0x0030 (0x009C - 0x00CC)
class UActorFactoryPhysicsAsset : public UActorFactory
{
public:
class UPhysicsAsset*                    PhysicsAsset;                      // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USkeletalMesh*                    SkeletalMesh;                      // 0x00A8 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bStartAwake : 1;                    // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bDamageAppliesImpulse : 1;               // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       bNotifyRigidBodyCollision : 1;            // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                       bUseCompartment : 1;                 // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                       bCastDynamicShadow : 1;               // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
struct FVector                      InitialVelocity;                   // 0x00B4 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                      DrawScale3D;                      // 0x00C0 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryPhysicsAsset");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryPlayerStart
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryPlayerStart : public UActorFactory
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryPlayerStart");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryPylon
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryPylon : public UActorFactory
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryPylon");
}

return uClassPointer;
};

};

// Class Engine.ActorFactorySkeletalMesh
// 0x001C (0x009C - 0x00B8)
class UActorFactorySkeletalMesh : public UActorFactory
{
public:
class USkeletalMesh*                    SkeletalMesh;                    // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UAnimSet*                         AnimSet;                         // 0x00A8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                            AnimSequenceName;                // 0x00B0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactorySkeletalMesh");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryStaticMesh
// 0x0018 (0x009C - 0x00B4)
class UActorFactoryStaticMesh : public UActorFactory
{
public:
class UStaticMesh*                    StaticMesh;                        // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                        DrawScale3D;                       // 0x00A8 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryStaticMesh");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryTrigger
// 0x0004 (0x009C - 0x00A0)
class UActorFactoryTrigger : public UActorFactory
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryTrigger");
}

return uClassPointer;
};

};
```

```
// Class Engine.AkBank
// 0x0010 (0x0060 - 0x0070)
class UAkBank : public UObject
{
public:
unsigned long                    AutoLoad : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    GenerateDefinition : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FBankLoadState                    LoadState;                    // 0x0064 (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AkBank");
}

return uClassPointer;
};

};

// Class Engine.AkBaseSoundObject
// 0x0000 (0x0060 - 0x0060)
class UAkBaseSoundObject : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AkBaseSoundObject");
}

return uClassPointer;
};

};

// Class Engine.AkEvent
// 0x0008 (0x0060 - 0x0068)
class UAkEvent : public UAkBaseSoundObject
{
public:
```

```
class UAkBank*                              RequiredBank;                              // 0x0060 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AkEvent");
}

return uClassPointer;
};

};

// Class Engine.SoundCue
// 0x00B0 (0x0060 - 0x0110)
class USoundCue : public UAkBaseSoundObject
{
public:
struct FName                               SoundClass;                              // 0x0060 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  SoundClassName;                          // 0x0068 (0x0001)
[0x0000000000000000]
unsigned long                            bDebug : 1;                             // 0x006C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
class USoundNode*                             FirstNode;                              // 0x0070 (0x0008)
[0x0000000000000000]
uint8_t                                 UnknownData00[0x50];                    // 0x0078 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.SoundCue.EditorData
float                                   MaxAudibleDistance;                     // 0x00C8 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                   VolumeMultiplier;                     // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   PitchMultiplier;                     // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   Duration;                          // 0x00D4 (0x0004)
[0x0000000000000000]
class UFaceFXAnimSet*                        FaceFXAnimSetRef;                        // 0x00D8
(0x0008) [0x0000000000000001] (CPF_Edit)
class FString                             FaceFXGroupName;                        // 0x00E0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                             FaceFXAnimName;                         // 0x00F0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                                  MaxConcurrentPlayCount;                  // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                  CurrentPlayCount;                     // 0x0104 (0x0004)
[0x0000000000202002] (CPF_Const | CPF_Transient)
struct FName                               SoundGroup;                              // 0x0108 (0x0008)
[0x0000000020000000]  CPF_Deprecated)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundCue");
}

return uClassPointer;
};

float GetCueDuration();
};

// Class Engine.ArchetypePool
// 0x0050 (0x0070 - 0x00C0)
class UArchetypePool : public UComponent
{
public:
struct FMultiMap_Mirror                    ObjectArchetypeMap;                    // 0x0070 (0x0050)
[0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ArchetypePool");
}

return uClassPointer;
};

class AActor* SpawnClass(class UClass* InClass, class AActor* SpawnOwner, struct FVector
SpawnLocation, struct FRotator SpawnRotation, unsigned long bNoCollisionFail);
class AActor* SpawnArchetype(class AActor* InArchetype, class AActor* SpawnOwner, struct
FVector SpawnLocation, struct FRotator SpawnRotation, unsigned long bNoCollisionFail);
class UObject* InstanceClass(class UClass* InClass);
class UObject* InstanceArchetype(class UObject* InArchetype);
};

// Class Engine.BookMark
// 0x0028 (0x0060 - 0x0088)
class UBookMark : public UObject
{
public:
struct FVector                          Location;                          // 0x0060 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FRotator                         Rotation;                          // 0x006C (0x000C)
[0x0000000000000001] (CPF_Edit)
```

```
TArray<class FString>                    HiddenLevels;                       // 0x0078 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.BookMark");
}

return uClassPointer;
};

};

// Class Engine.BookMark2D
// 0x000C (0x0060 - 0x006C)
class UBookMark2D : public UObject
{
public:
float                                    Zoom2D;                             // 0x0060 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FIntPoint                         Location;                           // 0x0064 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.BookMark2D");
}

return uClassPointer;
};

};

// Class Engine.KismetBookMark
// 0x0014 (0x006C - 0x0080)
class UKismetBookMark : public UBookMark2D
{
public:
class FString                            BookMarkSequencePathName;                // 0x0070
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.KismetBookMark");
}

return uClassPointer;
};

};

// Class Engine.Canvas
// 0x0068 (0x0060 - 0x00C8)
class UCanvas : public UObject
{
public:
class UFont*                        Font;                           // 0x0060 (0x0008)
[0x0000000000000000]
float                           OrgX;                           // 0x0068 (0x0004)
[0x0000000000000000]
float                           OrgY;                           // 0x006C (0x0004)
[0x0000000000000000]
float                           ClipX;                          // 0x0070 (0x0004)
[0x0000000000000000]
float                           ClipY;                          // 0x0074 (0x0004)
[0x0000000000000000]
float                           CurX;                           // 0x0078 (0x0004)
[0x0000000000000002] (CPF_Const)
float                           CurY;                           // 0x007C (0x0004)
[0x0000000000000002] (CPF_Const)
float                           CurZ;                           // 0x0080 (0x0004)
[0x0000000000000002] (CPF_Const)
float                           CurYL;                          // 0x0084 (0x0004)
[0x0000000000000000]
struct FColor                       DrawColor;                      // 0x0088 (0x0004)
[0x0000000000000000]
unsigned long                       bCenter : 1;                    // 0x008C (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                       bNoSmooth : 1;                    // 0x008C (0x0004)
[0x0000000000000000] [0x00000002]
int32_t                         SizeX;                          // 0x0090 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         SizeY;                          // 0x0094 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FPointer                     Canvas;                         // 0x0098 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                     SceneView;                        // 0x00A0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                         UnknownData00[0x8];                 // 0x00A8 (0x0008) MISSED
OFFSET
struct FPlane                       ColorModulate;                    // 0x00B0 (0x0010)
[0x0000000000000000]
class UTexture2D*                       DefaultTexture;                   // 0x00C0 (0x0008)
```

```
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Canvas");
}

return uClassPointer;
};

void DrawDebugGraph(class FString Title, float ValueX, float ValueY, float UL_X, float UL_Y, float
W, float H, struct FVector2D RangeX, struct FVector2D RangeY);
void DrawTextureDoubleLineW(struct FVector StartPoint, struct FVector EndPoint, float Perc,
float Spacing, float Width, struct FColor LineColor, struct FColor AltLineColor, class UTexture*
Tex, float U, float V, float UL, float VL);
void DrawTextureLineW(struct FVector StartPoint, struct FVector EndPoint, float Perc, float Width,
struct FColor LineColor, class UTexture* LineTexture, float U, float V, float UL, float VL);
void Draw2DLine(float X1, float Y1, float X2, float Y2, struct FColor LineColor);
void SetDrawColorStruct(struct FColor C);
void SetDrawColor(uint8_t R, uint8_t G, uint8_t B, uint8_t A);
void DrawBox(float Width, float Height);
void DrawRect(float RectX, float RectY, class UTexture* Tex);
void DrawPixel(int32_t X, int32_t Y, struct FLinearColor PixelColor, uint8_t BlendMode);
void DrawIcon(struct FCanvasIcon Icon, float X, float Y, float Scale);
void DrawScaledIcon(struct FCanvasIcon Icon, float X, float Y, struct FVector Scale);
struct FCanvasIcon MakeIcon(class UTexture* Texture, float U, float V, float UL, float VL);
void DrawBlendedTile(class UTexture* Tex, float XL, float YL, float U, float V, float UL, float VL,
uint8_t Blend);
void DrawTextureW(class UTexture* Tex, float Scale);
void PopMaskRegion();
void PushMaskRegion(float X, float Y, float XL, float YL);
void SetClip(float X, float Y);
void SetOrigin(float X, float Y);
void SetPos(float PosX, float PosY, float PosZ);
class UFont* GetDefaultCanvasFont();
void eventReset(unsigned long bKeepOrigin);
void Push3DTransform(struct FVector Translation, struct FRotator Rotation, float FOV);
void PopTransform();
void PushTranslationMatrix(struct FVector TranslationVector);
void DeProject(struct FVector2D ScreenPos, struct FVector& WorldOrigin, struct FVector&
WorldDirection);
struct FVector Project(struct FVector Location);
void DrawTextW(class FString Text, unsigned long CR, float XScale, float YScale, struct
FFontRenderInfo& RenderInfo);
void TextSize(class FString String, float XScale, float YScale, float& XL, float& YL);
void StrLen(class FString String, float& XL, float& YL);
static struct FFontRenderInfo CreateFontRenderInfo(unsigned long bClipText, unsigned long
bEnableShadow, struct FLinearColor GlowColor, struct FVector2D GlowOuterRadius, struct
FVector2D GlowInnerRadius);
```

```
void DrawTris(class UTexture* Tex, TArray<struct FCanvasUVTri> Triangles, struct FColor
InColor);
void DrawTileStretched(class UTexture* Tex, float XL, float YL, float U, float V, float UL, float VL,
struct FLinearColor LColor, unsigned long bStretchHorizontally, unsigned long bStretchVertically,
float ScalingFactor);
void DrawTimer(class UTexture* Tex, float StartTime, float TotalTime, float XL, float YL, float U,
float V, float UL, float VL, struct FLinearColor LColor, uint8_t Blend);
void DrawRotatedMaterialTile(class UMaterialInterface* Mat, struct FRotator Rotation, float XL,
float YL, float U, float V, float UL, float VL, float AnchorX, float AnchorY);
void DrawRotatedTile(class UTexture* Tex, struct FRotator Rotation, float XL, float YL, float U,
float V, float UL, float VL, float AnchorX, float AnchorY);
void DrawMaterialTile(class UMaterialInterface* Mat, float XL, float YL, float U, float V, float UL,
float VL, unsigned long bClipTile);
void PreOptimizeDrawTiles(int32_t Num, class UTexture* Tex, uint8_t Blend);
void DrawTile(class UTexture* Tex, float XL, float YL, float U, float V, float UL, float VL, struct
FLinearColor LColor, unsigned long ClipTile, uint8_t Blend);
};

// Class Engine.Channel
// 0x0040 (0x0060 - 0x00A0)
class UChannel : public UObject
{
public:
uint8_t                 UnknownData00[0x40];             // 0x0060 (0x0040)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Channel");
}

return uClassPointer;
};

};

// Class Engine.ActorChannel
// 0x0080 (0x00A0 - 0x0120)
class UActorChannel : public UChannel
{
public:
uint8_t                 UnknownData00[0x80];             // 0x00A0 (0x0080)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorChannel");
}

return uClassPointer;
};

};

// Class Engine.ControlChannel
// 0x0018 (0x00A0 - 0x00B8)
class UControlChannel : public UChannel
{
public:
uint8_t                         UnknownData00[0x18];                // 0x00A0 (0x0018)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ControlChannel");
}

return uClassPointer;
};

};

// Class Engine.FileChannel
// 0x0228 (0x00A0 - 0x02C8)
class UFileChannel : public UChannel
{
public:
uint8_t                         UnknownData00[0x228];               // 0x00A0 (0x0228)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FileChannel");
}

return uClassPointer;
};
```

```
};

// Class Engine.VoiceChannel
// 0x0010 (0x00A0 - 0x00B0)
class UVoiceChannel : public UChannel
{
public:
uint8_t                         UnknownData00[0x10];                    // 0x00A0 (0x0010)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.VoiceChannel");
}

return uClassPointer;
};

};

// Class Engine.Controller
// 0x020C (0x0268 - 0x0474)
class AController : public AActor
{
public:
struct FPointer                        VfTable_IInterface_NavigationHandle;        // 0x0268
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UObjectProvider*                      ObjectProvider;                      // 0x0270 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UGroupComponent_ORS*                  RegistryGroup;                          // 0x0278
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class APawn*                           Pawn;                            // 0x0280 (0x0008)
[0x0000000104000020] (CPF_Net | CPF_EditInline)
class APlayerReplicationInfo*                 PlayerReplicationInfo;                 // 0x0288
(0x0008) [0x0000000104000020] (CPF_Net | CPF_EditInline)
int32_t                              PlayerNum;                          // 0x0290 (0x0004)
[0x0000000000000002] (CPF_Const)
class AController*                        NextController;                       // 0x0298 (0x0008)
[0x0000000000000002] (CPF_Const)
unsigned long                          bIsPlayer : 1;                         // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                          bGodMode : 1;                          // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                          bSoaking : 1;                          // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                          bSlowerZAcquire : 1;                      // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                          bNotifyPostLanded : 1;                     // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000010]
```

```
unsigned long                          bNotifyApex : 1;                    // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                          bOverrideSearchStart : 1;                // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                          bAdvancedTactics : 1;                // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                          bCanDoSpecial : 1;                 // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                          bAdjusting : 1;                   // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                          bPreparingMove : 1;                  // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                          bForceStrafe : 1;                  // 0x02A0 (0x0004)
[0x0000000000000000] [0x00000800]
unsigned long                          bLOSflag : 1;                    // 0x02A0 (0x0004)
[0x0000000000000002] [0x00001000] (CPF_Const)
unsigned long                          bSkipExtraLOSChecks : 1;                // 0x02A0 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                          bNotifyFallingHitWall : 1;               // 0x02A0 (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                          bEarlyOutOfSighTestsForSameType : 1;        // 0x02A0
(0x0004) [0x0000000000000000] [0x00008000]
unsigned long                          bPreciseDestination : 1;               // 0x02A0 (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                          bSeeFriendly : 1;                  // 0x02A0 (0x0004)
[0x0000000000000000] [0x00020000]
unsigned long                          bUsingPathLanes : 1;                 // 0x02A0 (0x0004)
[0x0000000000000000] [0x00040000]
uint8_t                           bFire;                        // 0x02A4 (0x0001)
[0x0000000000000004] (CPF_Input)
uint8_t                           bAltFire;                       // 0x02A5 (0x0001)
[0x0000000000000004] (CPF_Input)
float                            MinHitWall;                       // 0x02A8 (0x0004)
[0x0000000000000000]
class UClass*                         NavigationHandleClass;                  // 0x02B0 (0x0008)
[0x0000000000000000]
class UNavigationHandle*                       NavigationHandle;                   // 0x02B8 (0x0008)
[0x0000000004000000] (CPF_EditInline)
struct FVector                         OverrideSearchStart;                 // 0x02C0 (0x000C)
[0x0000000000000000]
float                            MoveTimer;                        // 0x02CC (0x0004)
[0x0000000000000000]
class AActor*                         MoveTarget;                      // 0x02D0 (0x0008)
[0x0000000000000000]
struct FBasedPosition                       DestinationPosition;                  // 0x02D8 (0x0038)
[0x0000000000000000]
struct FBasedPosition                       FocalPosition;                    // 0x0310 (0x0038)
[0x0000000000000000]
class AActor*                         Focus;                         // 0x0348 (0x0008)
[0x0000000000000000]
class AActor*                         GoalList[0x4];                   // 0x0350 (0x0020)
[0x0000000000000000]
struct FBasedPosition                       AdjustPosition;                    // 0x0370 (0x0038)
[0x0000000000000000]
```

```
class ANavigationPoint*                    StartSpot;                         // 0x03A8 (0x0008)
[0x0000000000000000]
TArray<class ANavigationPoint*>              RouteCache;                          // 0x03B0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class UReachSpec*                          CurrentPath;                        // 0x03C0 (0x0008)
[0x0000000000000000]
class UReachSpec*                          NextRoutePath;                       // 0x03C8 (0x0008)
[0x0000000000000000]
struct FVector                         CurrentPathDir;                      // 0x03D0 (0x000C)
[0x0000000000000000]
class AActor*                          RouteGoal;                          // 0x03E0 (0x0008)
[0x0000000000000000]
float                            RouteDist;                          // 0x03E8 (0x0004)
[0x0000000000000000]
float                            LastRouteFind;                        // 0x03EC (0x0004)
[0x0000000000000000]
class AInterpActor*                        PendingMover;                        // 0x03F0 (0x0008)
[0x0000000000000000]
class AActor*                          FailedMoveTarget;                     // 0x03F8 (0x0008)
[0x0000000000000000]
int32_t                           MoveFailureCount;                     // 0x0400 (0x0004)
[0x0000000000000000]
float                            GroundPitchTime;                      // 0x0404 (0x0004)
[0x0000000000000000]
class APawn*                           ShotTarget;                         // 0x0408 (0x0008)
[0x0000000000000000]
class AActor*                          LastFailedReach;                      // 0x0410 (0x0008)
[0x0000000000000002] (CPF_Const)
float                            FailedReachTime;                      // 0x0418 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FVector                         FailedReachLocation;                   // 0x041C (0x000C)
[0x0000000000000002] (CPF_Const)
float                            SightCounter;                        // 0x0428 (0x0004)
[0x0000000000000000]
float                            SightCounterInterval;                  // 0x042C (0x0004)
[0x0000000000000000]
float                            InUseNodeCostMultiplier;                // 0x0430 (0x0004)
[0x0000000000000000]
int32_t                           HighJumpNodeCostModifier;               // 0x0434 (0x0004)
[0x0000000000000000]
float                            MaxMoveTowardPawnTargetTime;             // 0x0438 (0x0004)
[0x0000000000000000]
class APawn*                           Enemy;                            // 0x0440 (0x0008)
[0x0000000000000000]
TArray<struct FVisiblePortalInfo>             VisiblePortals;                       // 0x0448 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                            LaneOffset;                         // 0x0458 (0x0004)
[0x0000000000000000]
struct FRotator                         OldBasedRotation;                     // 0x045C (0x000C)
[0x0000000000000002] (CPF_Const)
struct FVector                         NavMeshPath_SearchExtent_Modifier;          // 0x0468
(0x000C) [0x0000000000000000]

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Controller");
}

return uClassPointer;
};

bool eventGeneratePathToLocation(struct FVector Goal, float WithinDistance, unsigned long bAllowPartialPath);
bool eventGeneratePathToActor(class AActor* Goal, float WithinDistance, unsigned long bAllowPartialPath);
void eventInterpolationFinished(class USeqAct_Interp* InterpAction);
void eventInterpolationStarted(class USeqAct_Interp* InterpAction, class UInterpGroupInst* GroupInst);
void InitNavigationHandle();
void ReadyForLift();
void eventCurrentLevelUnloaded();
bool eventIsInCombat(unsigned long bForceCheck);
bool eventIsSpectating();
void OnToggleHidden(class USeqAct_ToggleHidden* Action);
bool NotifyCoverClaimViolation(class AController* NewClaim, class ACoverLink* Link, int32_t SlotIdx);
void eventNotifyCoverAdjusted();
void NotifyCoverDisabled(class ACoverLink* Link, int32_t SlotIdx, unsigned long bAdjacentIdx);
void OnSetVelocity(class USeqAct_SetVelocity* Action);
void OnSetPhysics(class USeqAct_SetPhysics* Action);
void OnToggleGodMode(class USeqAct_ToggleGodMode* inAction);
void OnAttachToActor(class USeqAct_AttachToActor* Action);
void OnTeleport(class USeqAct_Teleport* Action);
bool IsDead();
class FString GetHumanReadableName();
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void StopLatentExecution();
bool InLatentExecution(int32_t LatentActionNumber);
void eventReachedPreciseDestination();
void eventNotifyMissedJump();
void eventNotifyJumpApex();
bool eventNotifyBump(class AActor* Other, struct FVector HitNormal);
void eventNotifyFallingHitWall(struct FVector HitNormal, class AActor* Wall);
bool eventNotifyHitWall(struct FVector HitNormal, class AActor* Wall);
bool eventNotifyLanded(struct FVector HitNormal, class AActor* FloorActor);
bool eventNotifyHeadVolumeChange(class APhysicsVolume* NewVolume);
void eventNotifyPhysicsVolumeChange(class APhysicsVolume* NewVolume);
bool LandingShake();
void eventGetActorEyesViewPoint(struct FVector& out_Location, struct FRotator& out_Rotation);
bool eventHandlePathObstruction(class AActor* BlockedBy);
void UnderLift(class ALiftCenter* Lift);
bool eventMoverFinished();
void WaitForMover(class AInterpActor* M);
```

```
bool eventAllowDetourTo(class ANavigationPoint* N);
void eventMayFall(unsigned long bFloor, struct FVector FloorNormal);
void eventLongFall();
void WaitForLanding(float waitDuration);
bool PickWallAdjust(struct FVector HitNormal);
void eventMoveUnreachable(struct FVector AttemptedDest, class AActor* AttemptedTarget);
bool ActorReachable(class AActor* anActor);
bool PointReachable(struct FVector aPoint);
class AActor* FindPathToIntercept(class APawn* P, class AActor* InRouteGoal, unsigned long bWeightDetours, int32_t MaxPathLength, unsigned long bReturnPartial);
class ANavigationPoint* FindRandomDest();
class AActor* FindPathTowardNearest(class UClass* GoalClass, unsigned long bWeightDetours, int32_t MaxPathLength, unsigned long bReturnPartial);
class AActor* FindPathToward(class AActor* anActor, unsigned long bWeightDetours, int32_t MaxPathLength, unsigned long bReturnPartial);
class AActor* FindPathTo(struct FVector aPoint, int32_t MaxPathLength, unsigned long bReturnPartial);
void FinishRotation();
void eventSetupSpecialPathAbilities();
void MoveToward(class AActor* NewTarget, class AActor* ViewFocus, float DestinationOffset, unsigned long bUseStrafing, unsigned long bShouldWalk);
void MoveToDirectNonPathPos(struct FVector NewDestination, class AActor* ViewFocus, float DestinationOffset, unsigned long bShouldWalk);
void MoveTo(struct FVector NewDestination, class AActor* ViewFocus, float DestinationOffset, unsigned long bShouldWalk);
void eventEnemyNotVisible();
void eventSeeMonster(class APawn* Seen);
void eventSeePlayer(class APawn* Seen);
void eventHearNoise(float Loudness, class AActor* NoiseMaker, struct FName NoiseType);
bool CanSeeByPoints(struct FVector ViewLocation, struct FVector TestLocation, struct FRotator ViewRotation);
bool CanSee(class APawn* Other);
bool LineOfSightTo(class AActor* Other, struct FVector chkLocation, unsigned long bTryAlternateTargetLoc);
void RoundHasEnded(class AActor* EndRoundFocus);
void GameHasEnded(class AActor* EndGameFocus, unsigned long bIsWinner);
void SetCharacter(class FString inCharacter);
void ServerGivePawn();
void ServerRestartPlayer();
uint8_t GetTeamNum();
void InitPlayerReplicationInfo();
void EnemyJustTeleported();
bool BeyondFogDistance(struct FVector ViewPoint, struct FVector OtherPoint);
void Restart();
void CleanupPRI();
void eventDestroyed();
void eventNotifyPostLanded();
bool GamePlayEndedState();
void eventUnPossess();
void eventPossess(class APawn* inPawn);
void OnPossess(class USeqAct_Possess* inAction);
void eventReplicatedEvent(struct FName VarName);
void ClientSetRotation(struct FRotator NewRotation, unsigned long bResetCamera);
void ClientSetLocation(struct FVector NewLocation, struct FRotator NewRotation);
void Reset();
void eventPostBeginPlay();
void eventSetSkelControlScale(struct FName SkelControlName, float Scale);
void eventSetMorphWeight(struct FName MorphNodeName, float MorphWeight);
void eventStopActorFaceFXAnim();
bool eventPlayActorFaceFXAnim(class UFaceFXAnimSet* AnimSet, class FString GroupName,
class FString SeqName, class USoundCue* SoundCueToPlay, class UAkEvent* AkEventToPlay);
void eventFinishAnimControl(class UInterpGroup* InInterpGroup);
void eventSetAnimPosition(struct FName SlotName, int32_t ChannelIndex, struct FName
InAnimSeqName, float InPosition, unsigned long bFireNotifies, unsigned long bLooping, unsigned
long bEnableRootMotion);
void eventBeginAnimControl(class UInterpGroup* InInterpGroup);
void eventNotifyPathChanged();
struct FVector GetAdjustLocation();
void SetAdjustLocation(struct FVector NewLoc, unsigned long bAdjust, unsigned long
bOffsetFromBase);
struct FVector GetDestinationPosition();
void SetDestinationPosition(struct FVector Dest, unsigned long bOffsetFromBase);
struct FVector GetFocalPoint();
void SetFocalPoint(struct FVector FP, unsigned long bOffsetFromBase);
void RouteCache_RemoveIndex(int32_t InIndex, int32_t Count);
void RouteCache_RemoveItem(class ANavigationPoint* Nav);
void RouteCache_InsertItem(class ANavigationPoint* Nav, int32_t Idx);
void RouteCache_AddItem(class ANavigationPoint* Nav);
void RouteCache_Empty();
bool IsLocalController();
bool IsLocalPlayerController();
void eventConstruct();
};

// Class Engine.PlayerController
// 0x035C (0x0474 - 0x07D0)
class APlayerController : public AController
{
public:
class UPlayer*                    Player;                         // 0x0478 (0x0008)
[0x0000000000000002] (CPF_Const)
class ACamera*                     PlayerCamera;                       // 0x0480 (0x0008)
[0x0000000004000001] (CPF_Edit | CPF_EditInline)
class UClass*                     CameraClass;                      // 0x0488 (0x0008)
[0x0000000000000002] (CPF_Const)
unsigned long                     bFrozen : 1;                      // 0x0490 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                     bPressedJump : 1;                     // 0x0490 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                     bDoubleJump : 1;                      // 0x0490 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                     bUpdatePosition : 1;                  // 0x0490 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                     bUpdating : 1;                     // 0x0490 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                     bCheatFlying : 1;                   // 0x0490 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                     bCameraPositionLocked : 1;              // 0x0490 (0x0004)
```

```
[0x0000000000000000] [0x00000040]
unsigned long                    bShortConnectTimeOut : 1;            // 0x0490 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                    bPendingDestroy : 1;                 // 0x0490 (0x0004)
[0x0000000000000002] [0x00000100] (CPF_Const)
unsigned long                    bWasSpeedHack : 1;                   // 0x0490 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                    bWasSaturated : 1;                   // 0x0490 (0x0004)
[0x0000000000000002] [0x00000400] (CPF_Const)
unsigned long                    bAimingHelp : 1;                     // 0x0490 (0x0004)
[0x0000000000044000] [0x00000800] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bCameraCut : 1;                      // 0x0490 (0x0004)
[0x0000000000002000] [0x00001000] (CPF_Transient)
unsigned long                    bClientSimulatingViewTarget : 1;     // 0x0490 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                    bHasVoiceHandshakeCompleted : 1;     // 0x0490
(0x0004) [0x0000000000000000] [0x00004000]
unsigned long                    bIsChatBanned : 1;                   // 0x0490 (0x0004)
[0x0001000000000000] [0x00008000]
unsigned long                    bIsSocialBanned : 1;                 // 0x0490 (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                    bCinematicMode : 1;                  // 0x0490 (0x0004)
[0x0000000000000000] [0x00020000]
unsigned long                    bInteractiveMode : 1;                // 0x0490 (0x0004)
[0x0000000000000000] [0x00040000]
unsigned long                    bCinemaDisableInputMove : 1;         // 0x0490 (0x0004)
[0x0000000000000000] [0x00080000]
unsigned long                    bCinemaDisableInputLook : 1;         // 0x0490 (0x0004)
[0x0000000000000000] [0x00100000]
unsigned long                    bIgnoreNetworkMessages : 1;          // 0x0490 (0x0004)
[0x0000000000000000] [0x00200000]
unsigned long                    bShowKismetDrawText : 1;             // 0x0490 (0x0004)
[0x0000000000004000] [0x00400000] (CPF_Config)
unsigned long                    bReplicateAllPawns : 1;              // 0x0490 (0x0004)
[0x0000000000000000] [0x00800000]
unsigned long                    bIsUsingStreamingVolumes : 1;        // 0x0490 (0x0004)
[0x0000000000000000] [0x01000000]
unsigned long                    bIsExternalUIOpen : 1;               // 0x0490 (0x0004)
[0x0000000000000000] [0x02000000]
unsigned long                    bIsControllerConnected : 1;          // 0x0490 (0x0004)
[0x0000000000000000] [0x04000000]
unsigned long                    bCheckSoundOcclusion : 1;            // 0x0490 (0x0004)
[0x0000000000000000] [0x08000000]
unsigned long                    bDebugCameraAnims : 1;               // 0x0490 (0x0004)
[0x0000000000000000] [0x10000000]
unsigned long                    bBlockCameraAnimsFromOverridingPostProcess : 1;//
0x0490 (0x0004) [0x0000000000000000] [0x20000000]
unsigned long                    bLogHearSoundOverflow : 1;           // 0x0490 (0x0004)
[0x0000000000044000] [0x40000000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bCheckRelevancyThroughPortals : 1;   // 0x0490
(0x0004) [0x0000000000044000] [0x80000000] (CPF_Config | CPF_GlobalConfig)
unsigned long                    bDebugClientAdjustPosition : 1;      // 0x0494 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                        MaxResponseTime;                         // 0x0498 (0x0004)
```

```
[0x0000000000000000]
float                          WaitDelay;                         // 0x049C (0x0004)
[0x0000000000000000]
class APawn*                        AcknowledgedPawn;                        // 0x04A0 (0x0008)
[0x0000000000000000]
uint8_t                        DoubleClickDir;                         // 0x04A8 (0x0001)
[0x0000000000000000]
uint8_t                        bIgnoreMoveInput;                         // 0x04A9 (0x0001)
[0x0000000000000000]
uint8_t                        bIgnoreLookInput;                         // 0x04AA (0x0001)
[0x0000000000000000]
uint8_t                        bRun;                            // 0x04AB (0x0001)
[0x0000000000000004] (CPF_Input)
uint8_t                        bDuck;                            // 0x04AC (0x0001)
[0x0000000000000004] (CPF_Input)
uint8_t                        NetPlayerIndex;                         // 0x04AD (0x0001)
[0x0000000000200002] (CPF_Const)
class AActor*                       ViewTarget;                         // 0x04B0 (0x0008)
[0x0000000000000002] (CPF_Const)
class APlayerReplicationInfo*                RealViewTarget;                         // 0x04B8 (0x0008)
[0x0000000000000000]
class UInterpTrackInstDirector*               ControllingDirTrackInst;                    // 0x04C0
(0x0008) [0x0000000000002000] (CPF_Transient)
float                          FOVAngle;                          // 0x04C8 (0x0004)
[0x0000000000000000]
float                          DesiredFOV;                          // 0x04CC (0x0004)
[0x0000000000000000]
float                          DefaultFOV;                          // 0x04D0 (0x0004)
[0x0000000000000000]
float                          LODDistanceFactor;                      // 0x04D4 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FRotator                     TargetViewRotation;                        // 0x04D8 (0x000C)
[0x0000000000000020] (CPF_Net)
float                          TargetEyeHeight;                       // 0x04E4 (0x0004)
[0x0000000000000020] (CPF_Net)
struct FRotator                     BlendedTargetViewRotation;                   // 0x04E8 (0x000C)
[0x0000000000000000]
class AHUD*                         myHUD;                             // 0x04F8 (0x0008)
[0x0000000000000000]
class AHUD*                         mySecondaryHUD;                         // 0x0500 (0x0008)
[0x0000000000000000]
class UClass*                       SavedMoveClass;                         // 0x0508 (0x0008)
[0x0000000000000000]
class USavedMove*                       SavedMoves;                          // 0x0510 (0x0008)
[0x0000000000000000]
class USavedMove*                       FreeMoves;                          // 0x0518 (0x0008)
[0x0000000000000000]
class USavedMove*                       PendingMove;                          // 0x0520 (0x0008)
[0x0000000000000000]
struct FVector                      LastAckedAccel;                         // 0x0528 (0x000C)
[0x0000000000000000]
float                          CurrentTimeStamp;                       // 0x0534 (0x0004)
[0x0000000000000000]
float                          LastUpdateTime;                         // 0x0538 (0x0004)
```

```
[0x0000000000000000]
float                          ServerTimeStamp;                    // 0x053C (0x0004)
[0x0000000000000000]
float                          TimeMargin;                      // 0x0540 (0x0004)
[0x0000000000000000]
float                          ClientUpdateTime;                   // 0x0544 (0x0004)
[0x0000000000000000]
float                          MaxTimeMargin;                      // 0x0548 (0x0004)
[0x0000000000000000]
float                          LastActiveTime;                   // 0x054C (0x0004)
[0x0000000000000000]
float                          DynamicPingThreshold;                  // 0x0550 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                          LastPingUpdate;                   // 0x0554 (0x0004)
[0x0000000000000000]
float                          LastSpeedHackLog;                    // 0x0558 (0x0004)
[0x0000000000000000]
struct FClientAdjustment                PendingAdjustment;                  // 0x0560 (0x0038)
[0x0000000000000000]
int32_t                         GroundPitch;                     // 0x0598 (0x0004)
[0x0000000000000000]
class UCheatManager*                 CheatManager;                      // 0x05A0 (0x0008)
[0x0000000000002000] (CPF_Transient)
class UClass*                      CheatClass;                        // 0x05A8 (0x0008)
[0x0000000000000000]
class UPlayerInput*                    PlayerInput;                       // 0x05B0 (0x0008)
[0x0000000004002001] (CPF_Edit | CPF_Transient | CPF_EditInline)
class UClass*                      InputClass;                        // 0x05B8 (0x0008)
[0x0000000000000000]
struct FVector                     FailedPathStart;                    // 0x05C0 (0x000C)
[0x0000000000000002] (CPF_Const)
class UCylinderComponent*                CylinderComponent;                  // 0x05D0
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class FString                      ForceFeedbackManagerClassName;           // 0x05D8
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UForceFeedbackManager*              ForceFeedbackManager;                // 0x05E8
(0x0008) [0x0000000000002000] (CPF_Transient)
TArray<class UInteraction*>               Interactions;                   // 0x05F0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FUniqueNetId>               VoiceMuteList;                  // 0x0600 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FUniqueNetId>               GameplayVoiceMuteList;               // 0x0610
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FUniqueNetId>               VoicePacketFilter;               // 0x0620 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FConnectedPeerInfo>            ConnectedPeers;                    // 0x0630
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FUniqueNetId>               BestNextHostPeers;                 // 0x0640
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class UOnlineGameSearch*                 MigratedSearchToJoin;               // 0x0650
(0x0008) [0x0000000000000000]
class UOnlineSubsystem*                OnlineSub;                    // 0x0658 (0x0008)
[0x0000000000000000]
class UOnlineVoiceInterface*              VoiceInterface_Object;              // 0x0660
```

```
(0x0008) [0x0000000000000000]
class UOnlineVoiceInterface*                VoiceInterface_Interface;               // 0x0668
(0x0008) [0x0000000000000000]
class UUIDataStore_OnlinePlayerData*            OnlinePlayerData;                   // 0x0670
(0x0008) [0x0000000000000000]
float                               InteractDistance;                   // 0x0678 (0x0004)
[0x0000000000004000] (CPF_Config)
struct FName                            DelayedJoinSessionName;                 // 0x067C (0x0008)
[0x0000000000000000]
TArray<struct FInputMatchRequest>               InputRequests;                      // 0x0688
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
float                               LastBroadcastTime;                      // 0x0698 (0x0004)
[0x0000000000000000]
class FString                           LastBroadcastString[0x4];               // 0x06A0 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FName>                        PendingMapChangeLevelNames;             // 0x06E0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class ACoverReplicator*                     MyCoverReplicator;                  // 0x06F0 (0x0008)
[0x0000000000000000]
TArray<struct FDebugTextInfo>                   DebugTextList;                      // 0x06F8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                               SpectatorCameraSpeed;                   // 0x0708 (0x0004)
[0x0000000000000000]
class UNetConnection*                       PendingSwapConnection;                  // 0x0710
(0x0008) [0x0000000000200002] (CPF_Const)
float                               MinRespawnDelay;                        // 0x0718 (0x0004)
[0x0000000000000000]
int32_t                             MaxConcurrentHearSounds;                // 0x071C (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
TArray<class UAudioComponent*>                  HearSoundActiveComponents;              //
0x0720 (0x0010) [0x0000000004480008] (CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
TArray<class UAudioComponent*>                  HearSoundPoolComponents;                //
0x0730 (0x0010) [0x0000000004480008] (CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
TArray<class AActor*>                       HiddenActors;                       // 0x0740 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                               LastSpectatorStateSynchTime;            // 0x0750 (0x0004)
[0x0000000000000000]
struct FScriptDelegate                      __EventConnectionTimedOut__Delegate;        // 0x0758
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                      __EventDisconnected__Delegate;          // 0x0770
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                      __OnMissingPeersUnregistered__Delegate;     // 0x0788
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                      __CanUnpause__Delegate;                 // 0x07A0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                      __InputMatchDelegate__Delegate;         // 0x07B8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlayerController");
}

return uClassPointer;
};

bool LimitSpectatorVelocity();
void Jump();
void FindGoodView();
void eventSetMatineeConstantCameraAnim(unsigned long bOn, uint8_t Type, float Rate);
bool eventNotifyDisconnect(class FString Command);
void ReceivedGameClass(class UClass* GameClass);
void LogOutBugItAIGoToLogFile(class FString InScreenShotDesc, class FString InGoString, class
FString InLocString);
void LogOutBugItGoToLogFile(class FString InScreenShotDesc, class FString InGoString, class
FString InLocString);
void DisableDebugAI();
void eventOnEngineInitialTick();
void BugItStringCreator(struct FVector& ViewLocation, struct FRotator& ViewRotation, class
FString& GoString, class FString& LocString);
void BugItAI(class FString ScreenShotDescription);
void LogLoc();
void BugIt(class FString ScreenShotDescription);
void BugItWorker(struct FVector TheLocation, struct FRotator TheRotation);
void BugItGoString(class FString TheLocation, class FString TheRotation);
void BugItGo(float X, float Y, float Z, int32_t Pitch, int32_t Yaw, int32_t Roll);
bool HasPeerConnection(struct FUniqueNetId& PeerNetId);
void OnSetSoundMode(class USeqAct_SetSoundMode* Action);
void eventClientSpawnCameraLensEffect(class UClass* LensEffectEmitterClass);
void DebugCameraAnims();
void eventClientStopCameraAnim(class UCameraAnim* AnimToStop, unsigned long
bImmediate);
void eventClientPlayCameraAnim(class UCameraAnim* AnimToPlay, float Scale, float Rate, float
BlendInTime, float BlendOutTime, unsigned long bLoop, unsigned long bRandomStartTime,
uint8_t Space, struct FRotator CustomPlaySpace);
void OnCameraShake(class USeqAct_CameraShake* inAction);
void ClientStopCameraShake(class UCameraShake* Shake);
void ClientPlayCameraShake(class UCameraShake* Shake, float Scale, unsigned long
bTryForceFeedback, uint8_t PlaySpace, struct FRotator UserPlaySpaceRot);
void DoForceFeedbackForScreenShake(class UCameraShake* ShakeData, float ShakeScale);
void InputMatchDelegate();
void Sentinel_PostAcquireTravelTheWorldPoints();
void Sentinel_PreAcquireTravelTheWorldPoints();
void Sentinel_SetupForGamebasedTravelTheWorld();
void OnFlyThroughHasEnded(class USeqAct_FlyThroughHasEnded* inAction);
bool eventGetAchievementProgression(int32_t AchievementId, float& CurrentValue, float&
MaxValue);
static class FString GetPartyGameTypeName();
static class FString GetPartyMapName();
bool IsPartyLeader();
void eventClientAddTextureStreamingLoc(struct FVector InLoc, float Duration, unsigned long
bOverrideLocation);
void eventClientPrestreamTextures(class AActor* ForcedActor, float ForceDuration, unsigned
long bEnableStreaming, int32_t CinematicTextureGroups);
void eventClientSetForceMipLevelsToBeResident(class UMaterialInterface* Material, float
ForceDuration, int32_t CinematicTextureGroups);
void ClientControlMovieTexture(class UTextureMovie* MovieTexture, uint8_t Mode);
int32_t GetSplitscreenPlayerCount();
class APlayerReplicationInfo* GetSplitscreenPlayerByIndex(int32_t PlayerIndex);
bool IsSplitscreenPlayer(int32_t& out_SplitscreenPlayerIndex);
bool IsPrimaryPlayer();
void ClientNotifyPartyHostLeaving(struct FUniqueNetId PartyHostPlayerId);
void ClientReturnToParty(struct FUniqueNetId RequestingPlayerId);
void OnJoinTravelToSessionComplete(struct FName SessionName, unsigned long
bWasSuccessful);
void PreJoinUpdateGameSettings(struct FName SessionName, class UOnlineGameSettings*
GameSettings);
void ClientTravelToSession(struct FName SessionName, class UClass* SearchClass, uint8_t
PlatformSpecificInfo);
void PathClear();
void PathChild(int32_t Cnt);
void PathStep(int32_t Cnt);
void eventSoakPause(class APawn* P);
void IncrementNumberOfMatchesPlayed();
bool CanViewUserCreatedContent();
void ClientEndOnlineGame();
void ClientStartOnlineGame();
void OnRegisterHostStatGuidComplete(unsigned long bWasSuccessful);
void ClientRegisterHostStatGuid(class FString StatGuid);
void eventRemoveAllDebugStrings();
void eventRemoveDebugText(class AActor* SrcActor);
void eventAddDebugText(class FString DebugText, class AActor* SrcActor, float Duration, struct
FVector Offset, struct FVector DesiredOffset, struct FColor TextColor, unsigned long
bSkipOverwriteCheck, unsigned long bAbsoluteLocation, unsigned long bKeepAttachedToActor,
class UFont* InFont, struct FVector2D InTextScale);
void DrawDebugTextList(class UCanvas* Canvas, float RenderDelta);
void OnDestroy(class USeqAct_Destroy* Action);
void ClientStartNetworkedVoice();
void ClientStopNetworkedVoice();
void ClientWriteLeaderboardStats(class UClass* OnlineStatsWriteClass, unsigned long
bIsIncomplete);
void ClientWriteOnlinePlayerScores(int32_t LeaderboardId);
void NotifyNotEnoughSpaceInInvite();
void NotifyNotAllPlayersCanJoinInvite();
void NotifyInviteFailed();
void OnInviteJoinComplete(struct FName SessionName, unsigned long bWasSuccessful);
class FString ModifyClientURL(class FString URL);
void OnDestroyForInviteComplete(struct FName SessionName, unsigned long bWasSuccessful);
void OnEndForInviteComplete(struct FName SessionName, unsigned long bWasSuccessful);
void ClearInviteDelegates();
bool CanAllPlayersPlayOnline();
bool InviteHasEnoughSpace(class UOnlineGameSettings* InviteSettings);
void OnGameInviteAccepted(class FString ErrorString, struct FOnlineGameSearchResult&
InviteResult);
bool IsShowingSubtitles();
```

```
void SetShowSubtitles(unsigned long bValue);
void eventNotifyDirectorControl(unsigned long bNowControlling, class USeqAct_Interp* CurrentMatinee);
void eventServerUnmutePlayer(struct FUniqueNetId PlayerNetId);
void eventServerMutePlayer(struct FUniqueNetId PlayerNetId);
void GameplayUnmutePlayer(struct FUniqueNetId PlayerNetId);
void GameplayMutePlayer(struct FUniqueNetId PlayerNetId);
void eventClientUnmutePlayer(struct FUniqueNetId PlayerNetId);
void eventClientMutePlayer(struct FUniqueNetId PlayerNetId, unsigned long bAddToMuteList);
void ClientVoiceHandshakeComplete();
static class APlayerController* GetPlayerControllerFromNetId(struct FUniqueNetId PlayerNetId);
void ClientSetOnlineStatus();
void SeamlessTravelFrom(class APlayerController* OldPC);
void SeamlessTravelTo(class APlayerController* NewPC);
void eventGetSeamlessTravelActorList(unsigned long bToEntry, TArray<class AActor*>& ActorList);
bool IsPlayerMuted(struct FUniqueNetId& Sender);
class UUIInteraction* GetUIController();
void SaveActorConfig(struct FName actorName);
void SaveClassConfig(class FString ClassName);
void ClientWaitForLevelsVisible(class USeqAct_WaitForLevelsVisible* inAction);
void eventClientSetBlockOnAsyncLoading();
void eventClientFlushLevelStreaming();
void eventClientCancelPendingMapChange();
void eventClientCommitMapChange();
void DelayedPrepareMapChange();
void eventClientPrepareMapChange(struct FName LevelName, unsigned long bFirst, unsigned long bLast);
void eventServerUpdateLevelVisibility(struct FName PackageName, unsigned long bIsVisible);
void ClientUpdateLevelStreamingStatus(struct FName PackageName, unsigned long bNewShouldBeLoaded, unsigned long bNewShouldBeVisible, unsigned long bNewShouldBlockOnLoad);
void eventLevelStreamingStatusChanged(class ULevelStreaming* LevelObject, unsigned long bNewShouldBeLoaded, unsigned long bNewShouldBeVisible, unsigned long bNewShouldBlockOnLoad);
void eventClientForceGarbageCollection();
void OnConsoleCommand(class USeqAct_ConsoleCommand* inAction);
void ResetPlayerMovementInput();
bool eventIsLookInputIgnored();
void IgnoreLookInput(unsigned long bNewLookInput);
bool eventIsMoveInputIgnored();
void IgnoreMoveInput(unsigned long bNewMoveInput);
void ClientSetCinematicMode(unsigned long bInCinematicMode, unsigned long bAffectsMovement, unsigned long bAffectsTurning, unsigned long bAffectsHUD);
void SetCinematicMode(unsigned long bInCinematicMode, unsigned long bHidePlayer, unsigned long bAffectsHUD, unsigned long bAffectsMovement, unsigned long bAffectsTurning, unsigned long bAffectsButtons);
void OnToggleCinematicMode(class USeqAct_ToggleCinematicMode* Action);
bool IsForceFeedbackAllowed();
void eventClientStopForceFeedbackWaveform(class UForceFeedbackWaveform* FFWaveform);
void eventClientPlayForceFeedbackWaveform(class UForceFeedbackWaveform* FFWaveform, class AActor* FFWaveformInstigator);
void eventPlayRumble(class UAnimNotify_Rumble* TheAnimNotify);
void OnForceFeedback(class USeqAct_ForceFeedback* Action);
void ShowGameState();
void ShowPlayerState();
void ServerRemoteEvent(struct FName EventName);
void RE(struct FName EventName);
void RemoteEvent(struct FName EventName);
void ListCE();
void ListConsoleEvents();
void CE(struct FName EventName);
void CauseEvent(struct FName EventName);
void ServerCauseEvent(struct FName EventName);
void OnToggleHUD(class USeqAct_ToggleHUD* inAction);
void OnSetCameraTarget(class USeqAct_SetCameraTarget* inAction);
void ClientClearKismetText(struct FVector2D MessageOffset);
void ClientDrawKismetText(struct FKismetDrawTextInfo DrawTextInfo, float DisplayTime);
void OnDrawTextW(class USeqAct_DrawText* inAction);
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void ClientIgnoreLookInput(unsigned long bIgnore);
void ClientIgnoreMoveInput(unsigned long bIgnore);
void OnToggleInput(class USeqAct_ToggleInput* inAction);
void eventAdjustHUDRenderSize(int32_t FullScreenSizeX, int32_t FullScreenSizeY, int32_t& X,
int32_t& Y, int32_t& SizeX, int32_t& SizeY);
void DrawHUD(class AHUD* H);
bool CanRestartPlayer();
void ViewAPlayer(int32_t Dir);
class APlayerReplicationInfo* GetNextViewablePlayer(int32_t Dir);
bool IsSpectating();
void AdjustPlayerWalkingMoveAccel(struct FVector& newAccel);
void CheckJumpOrDuck();
struct FRotator eventLimitViewRotation(struct FRotator ViewRotation, float ViewPitchMin, float
ViewPitchMax);
void ProcessViewRotation(float DeltaTime, struct FRotator DeltaRot, struct FRotator&
out_ViewRotation);
void UpdateRotation(float DeltaTime);
void ViewShake(float DeltaTime);
bool IsAimingAt(class AActor* ATarget, float Epsilon);
void GetPlayerViewPoint(struct FVector& out_Location, struct FRotator& out_Rotation);
void eventSpawnPlayerCamera();
void ServerVerifyViewTarget();
class AActor* GetViewTarget();
void eventClientSetViewTarget(class AActor* A, struct FViewTargetTransitionParams
TransitionParams);
void SetViewTargetWithBlend(class AActor* NewViewTarget, float BlendTime, uint8_t BlendFunc,
float BlendExp, unsigned long bLockOutgoing);
void SetViewTarget(class AActor* NewViewTarget, struct FViewTargetTransitionParams
TransitionParams);
bool IsLocalController();
bool IsLocalPlayerController();
float GetFOVAngle();
void AdjustFOV(float DeltaTime);
bool eventNotifyLanded(struct FVector HitNormal, class AActor* FloorActor);
void eventCameraLookAtFinished(class USeqAct_CameraLookAt* Action);
bool AimingHelp(unsigned long bInstantHit);
void PlayerMove(float DeltaTime);
void eventPlayerTick(float DeltaTime);
void ClientGameEnded(class AActor* EndGameFocus, unsigned long bIsWinner);
void GameHasEnded(class AActor* EndGameFocus, unsigned long bIsWinner);
void ClientRestart(class APawn* NewPawn);
void EnterStartState();
void ForceSingleNetUpdateFor(class AActor* Target);
bool HasClientLoadedCurrentWorld();
void eventNotifyLoadedWorld(struct FName WorldPackageName, unsigned long bFinalDest);
void eventServerNotifyLoadedWorld(struct FName WorldPackageName);
void Restart();
void eventClientSetProgressMessage(uint8_t MessageType, class FString Message, class
FString Title, unsigned long bIgnoreFutureNetworkMessages);
void SwitchLevel(class FString URL);
void Suicide();
bool TriggerInteracted();
void GetTriggerUseList(float interactDistanceToCheck, float crosshairDist, float minDot, unsigned
long bUsuableOnly, TArray<class ATrigger*>& out_useList);
void UTrace();
void eventConditionalPause(unsigned long bDesiredPauseState);
void Pause();
bool IsPaused();
bool SetPause(unsigned long bPause, struct FScriptDelegate CanUnpauseDelegate);
bool CanUnpause();
void LocalTravel(class FString URL);
void RestartLevel();
void Speech(struct FName Type, int32_t Index, class FString Callsign);
void HandleWalking();
void ReplicateMove(float DeltaTime, struct FVector newAccel, uint8_t DoubleClickMove, struct
FRotator DeltaRot);
int32_t CompressAccel(int32_t C);
class USavedMove* GetFreeMove();
void ClientUpdatePosition();
void ClearAckedMoves();
void ServerUpdatePing(int32_t NewPing);
void UpdateStateFromAdjustment(struct FName NewState);
void LongClientAdjustPosition(float TimeStamp, struct FName NewState, uint8_t newPhysics,
float NewLocX, float NewLocY, float NewLocZ, float NewVelX, float NewVelY, float NewVelZ,
class AActor* NewBase, float NewFloorX, float NewFloorY, float NewFloorZ);
bool SkipPositionUpdateForRM();
void UpdatePing(float DeltaTime);
void ClientAdjustPosition(float TimeStamp, struct FName NewState, uint8_t newPhysics, float
NewLocX, float NewLocY, float NewLocZ, float NewVelX, float NewVelY, float NewVelZ, class
AActor* NewBase);
void ClientAckGoodMove(float TimeStamp);
void ShortClientAdjustPosition(float TimeStamp, struct FName NewState, uint8_t newPhysics,
float NewLocX, float NewLocY, float NewLocZ, class AActor* NewBase);
void VeryShortClientAdjustPosition(float TimeStamp, float NewLocX, float NewLocY, float
NewLocZ, class AActor* NewBase);
void MoveAutonomous(float DeltaTime, uint8_t CompressedFlags, struct FVector newAccel,
struct FRotator DeltaRot);
void ProcessMove(float DeltaTime, struct FVector newAccel, uint8_t DoubleClickMove, struct
FRotator DeltaRot);
void ProcessDrive(float InForward, float InStrafe, float InUp, unsigned long InJump);
void eventSendClientAdjustment();
void ServerMoveHandleClientError(float TimeStamp, struct FVector Accel, struct FVector
ClientLoc);
float GetServerMoveDeltaTime(float TimeStamp);
void ForceDeathUpdate();
bool UsingFirstPersonCamera();
void eventClientSetCameraFade(unsigned long bEnableFading, struct FColor FadeColor, struct FVector2D FadeAlpha, float FadeTime, unsigned long bFadeAudio);
void eventResetCameraMode();
void SetCameraMode(struct FName NewCamMode);
void ClientSetCameraMode(struct FName NewCamMode);
bool eventPreClientTravel(class FString PendingURL, uint8_t TravelType, unsigned long bIsSeamlessTravel);
void TeamSay(class FString msg);
void ClientAdminMessage(class FString msg);
void Say(class FString msg);
bool AllowTextMessage(class FString msg);
void Mutate(class FString MutateString);
void FOV(float F);
void ResetFOV();
void SetFOV(float NewFOV);
void FixFOV();
void eventDestroyed();
void CleanupPawn();
void eventClearOnlineDelegates();
void OnPartyMembersInfoChanged(class FString PlayerName, struct FUniqueNetId PlayerID, int32_t CustomData1, int32_t CustomData2, int32_t CustomData3, int32_t CustomData4);
void OnPartyMemberListChanged(unsigned long bJoinedOrLeft, class FString PlayerName, struct FUniqueNetId PlayerID);
void RegisterOnlineDelegates();
void PlayBeepSound();
void eventTeamMessage(class APlayerReplicationInfo* PRI, class FString S, struct FName Type, float MsgLifeTime);
void SpeakTTS(class FString S, class APlayerReplicationInfo* PRI);
class USoundCue* CreateTTSSoundCue(class FString StrToSpeak, class APlayerReplicationInfo* PRI);
bool AllowTTSMessageFrom(class APlayerReplicationInfo* PRI);
bool CanCommunicate();
void eventClientMessage(class FString S, struct FName Type, float MsgLifeTime);
void ClientPlayActorFaceFXAnim(class AActor* SourceActor, class UFaceFXAnimSet* AnimSet, class FString GroupName, class FString SeqName, class USoundCue* SoundCueToPlay, class UAkEvent* AkEventToPlay);
void eventKismet_ClientStopSound(class USoundCue* ASound, class AActor* SourceActor, float FadeOutTime);
void eventKismet_ClientPlaySound(class USoundCue* ASound, class AActor* SourceActor, float VolumeMultiplier, float PitchMultiplier, float FadeInTime, unsigned long bSuppressSubtitles, unsigned long bSuppressSpatialization);
bool IsClosestLocalPlayerToActor(class AActor* TheActor);
void eventWwiseClientHearSound(class UAkEvent* ASound, class AActor* SourceActor, struct FVector SourceLocation, unsigned long bStopWhenOwnerDestroyed, unsigned long bIsOccluded);
void eventClientHearSound(class USoundCue* ASound, class AActor* SourceActor, struct FVector SourceLocation, unsigned long bStopWhenOwnerDestroyed, unsigned long bIsOccluded);
class UAudioComponent* GetPooledAudioComponent(class USoundCue* ASound, class AActor* SourceActor, unsigned long bStopWhenOwnerDestroyed, unsigned long bUseLocation, struct
FVector SourceLocation);
void HearSoundFinished(class UAudioComponent* AC);
void eventClientPlaySound(class USoundCue* ASound);
void ClientSetSecondaryHUD(class UClass* newHUDType);
void ClientSetHUD(class UClass* newHUDType);
void eventUnPossess();
void ServerAcknowledgePossession(class APawn* P);
void AcknowledgePossession(class APawn* P);
void eventPossess(class APawn* aPawn);
void GivePawn(class APawn* NewPawn);
void AskForPawn();
void ClientGotoState(struct FName NewState, struct FName NewLabel);
void SetTiltActive(unsigned long bActive);
bool IsMouseAvailable();
bool IsKeyboardAvailable();
void SetUseTiltForwardAndBack(unsigned long bActive);
void SetOnlyUseControllerTiltInput(unsigned long bActive);
void SetControllerTiltActive(unsigned long bActive);
bool IsControllerTiltActive();
void SetRumbleScale(float ScaleBy);
void ReloadProfileSettings();
void UnregisterStandardPlayerDataStores();
void UnregisterPlayerDataStores();
void RegisterStandardPlayerDataStores();
void RegisterCustomPlayerDataStores();
void RegisterPlayerDataStores();
void ClientInitializeDataStores();
void eventInitInputSystem();
void SetPlayerInput(class UClass* NewInputClass);
class UOnlineSubsystem* GetOnlineSubsystem();
void UpdatePrimaryPlayerORS();
void PostControllerIdChange();
void PreControllerIdChange();
void CleanOutSavedMoves();
void ClientReset();
void Reset();
void SetPlayerCamera(class ACamera* NewCamera);
void SetHUD(class AHUD* NewHUD);
void SpawnDefaultHUD();
void EnableCheats();
void AddCheats(unsigned long bForce);
void ServerGivePawn();
void ServerShortTimeout();
void ResetTimeMargin();
void eventPreRender(class UCanvas* Canvas);
void OnJoinMigratedGame(struct FName SessionName, unsigned long bWasSuccessful);
void PeerDesignatedAsClient(struct FName SessionName);
void OnUnregisterPlayerCompleteForJoinMigrate(struct FName SessionName, struct
FUniqueNetId PlayerID, unsigned long bWasSuccessful);
void eventPeerReceivedMigratedSession(struct FUniqueNetId FromPeerNetId, struct FName
SessionName, class UClass* SearchClass, uint8_t PlatformSpecificInfo);
void TellPeerToTravelToSession(struct FUniqueNetId ToPeerNetId, struct FName SessionName,
class UClass* SearchClass, uint8_t PlatformSpecificInfo, int32_t PlatformSpecificInfoSize);
void TellPeerToTravel(struct FUniqueNetId ToPeerNetId);
void PeerTravelAsHost(float TravelCountdownTimer, class FString URL);
class FString GetNewPeerHostURL();
void PeerDesignatedAsHost(struct FName SessionName);
class UClass* GetCurrentSearchClass();
void OnHostMigratedOnlineGame(struct FName SessionName, unsigned long bWasSuccessful);
void OnUnregisterPlayerCompleteForHostMigrate(struct FName SessionName, struct FUniqueNetId PlayerID, unsigned long bWasSuccessful);
bool RemoveMissingPeersFromSession(struct FName SessionName, struct FScriptDelegate UnregisterDelegate);
class APlayerReplicationInfo* GetPRIFromNetId(struct FUniqueNetId PlayerID);
void OnMissingPeersUnregistered(struct FName SessionName, struct FUniqueNetId PlayerID, unsigned long bWasSuccessful);
void GetRegisteredPlayersInSession(struct FName SessionName, TArray<struct FUniqueNetId>& OutRegisteredPlayers);
void NotifyHostMigrationStarted();
bool eventMigrateNewHost();
bool IsBestHostPeer(struct FUniqueNetId PeerNetId);
void eventNotifyPeerDisconnectHost(struct FUniqueNetId PeerNetId);
void ClientUpdateBestNextHosts(struct FUniqueNetId SortedNextHosts, uint8_t NumEntries);
void eventRemovePeer(struct FUniqueNetId PeerNetId);
void eventAddPeer(struct FUniqueNetId PeerNetId, uint8_t NatType);
int32_t FindConnectedPeerIndex(struct FUniqueNetId PeerNetId);
void eventReceivedPlayer();
void eventPostBeginPlay();
class ACoverReplicator* SpawnCoverReplicator();
bool CanUnpauseControllerConnected();
void OnControllerChanged(int32_t ControllerId, unsigned long bIsConnected);
bool CanUnpauseExternalUI();
void OnExternalUIChanged(unsigned long bIsOpening);
void ForceClearUnpauseDelegates();
void DisableActorHeadTracking(class AActor* TargetActor);
void EnableActorHeadTracking(class AActor* TargetActor, struct FName TrackControllerName, class UClass* ActorClassesToLookAt, unsigned long bLookAtPawns, float MinLookAtTime, float MaxLookAtTime, float MaxInterestTime, float LookAtActorRadius, struct FName TargetBoneNames);
void eventFellOutOfWorld();
void CleanUpAudioComponents();
int32_t FindStairRotation(float DeltaTime);
bool CheckSpeedHack(float DeltaTime);
void eventServerProcessConvolve(class FString C, int32_t H);
void eventClientConvolve(class FString C, int32_t H);
void SetAudioGroupVolume(struct FName GroupName, float Volume);
void SetAllowMatureLanguage(unsigned long bAllowMatureLanguge);
class FString PasteFromClipboard();
void CopyToClipboard(class FString Text);
void UpdateURL(class FString NewOption, class FString NewValue, unsigned long bSave1Default);
void eventClientTravel(class FString URL, uint8_t TravelType, unsigned long bSeamless, struct FGuid MapPackageGuid);
class FString ConsoleCommand(class FString Command, unsigned long bWriteToLog);
class FString GetServerNetworkAddress();
class FString GetPlayerNetworkAddress();
void EventDisconnected(class APlayerController* PC);
void EventConnectionTimedOut(class APlayerController* PC);
```

```
};

// Class Engine.CheatManager
// 0x0020 (0x0060 - 0x0080)
class UCheatManager : public UObject
{
public:
class FString                              ViewingFrom;                              // 0x0060 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                              OwnCamera;                                // 0x0070 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CheatManager");
}

return uClassPointer;
};

void AnalyticsEndSession();
void AnalyticsStartSession();
void GetAnalyticsUserId();
void SetAnalyticsUserId(class FString UserId);
void SendAnalyticsCachedEvents();
void SendAnalyticsCurrencyGivenEvent(class FString GameCurrencyType, int32_t
GameCurrencyAmount);
void SendAnalyticsCurrencyPurchaseEvent(class FString GameCurrencyType, int32_t
GameCurrencyAmount, class FString RealCurrencyType, float RealMoneyCost, class FString
PaymentProvider);
void SendAnalyticsItemPurchaseEvent(class FString ItemID, class FString Currency, int32_t
PerItemCost, int32_t ItemQuantity);
void SendAnalyticsUserAttributeEvent(class FString AttributeName, class FString
AttributeValue);
void SendAnalyticsEvent(class FString EventName, class FString AttributeName, class FString
AttributeValue);
void OnRequestComplete(class UHttpRequestInterface* OriginalRequest, class
UHttpResponseInterface* Response, unsigned long bDidSucceed);
void TestHttp(class FString Verb, class FString Payload, class FString URL, unsigned long
bSendParallelRequest);
void OnDeleteUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void DebugDeleteUserFile(class FString UserId, class FString Filename);
void OnReadUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void DebugReadUserFile(class FString UserId, class FString Filename);
void OnWriteUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void DebugWriteUserFile(class FString UserId, class FString Filename);
void OnEnumerateUserFilesComplete(unsigned long bWasSuccessful, class FString UserId);
void DebugQueryUserFiles(class FString UserId);
void OnReceivedLocalNotificationDebug(unsigned long bWasAppActive, struct FNotificationInfo& Notification);
void DebugNotification(class FString MessageBody, int32_t SecondsFromNow);
void DrawLocationXYZ(float X, float Y, float Z);
void DrawLocation(struct FVector Loc);
void ToggleUsingHighestMips();
void DumpCoverStats();
void DebugEmsDownload();
void DebugDeleteTitleFiles();
void OnSaveComplete(unsigned long bWasSuccessful, class FString Filename);
void DebugSaveTitleFile(class FString Filename);
void OnLoadComplete(unsigned long bWasSuccessful, class FString Filename);
void OnDownloadComplete(unsigned long bWasSuccessful, class FString Filename);
void DebugDownloadTitleFile(class FString Filename, unsigned long bFromCache);
void DebugIniLocPatcher();
void ToggleAILogging();
void VerifyNavMeshCoverRefs();
void PrintNavMeshObstacles();
void PrintAllPathObjectEdges();
void NavMeshVerification(float interval);
void DrawUnsupportingEdges(class FString PawnClassName);
void VerifyNavMeshObjects();
void LogParticleActivateSystemCalls(unsigned long bShouldLog);
void LogPlaySoundCalls(unsigned long bShouldLog);
void InitCheatManager();
void VerbosePathDebug();
void TestPylonConnectivity();
void TestNavMeshPath(unsigned long bDrawPath);
void SetOnlineDebugLevel(int32_t DebugLevel);
void TestLevel();
void StreamLevelOut(struct FName PackageName);
void OnlyLoadLevel(struct FName PackageName);
void StreamLevelIn(struct FName PackageName);
void SetLevelStreamingStatus(struct FName PackageName, unsigned long bShouldBeLoaded, unsigned long bShouldBeVisible);
void ViewClass(class UClass* aClass);
void ViewBot();
void ViewActor(struct FName actorName);
void ViewPlayer(class FString S);
void ViewSelf(unsigned long bQuiet);
void RememberSpot();
void SuspendAI();
void PlayersOnly();
void Summon(class FString ClassName);
void Avatar(struct FName ClassName);
void KillPawns();
void KillAllPawns(class UClass* aClass);
void KillAll(class UClass* aClass);
void SetSpeed(float F);
void SetGravity(float F);
void SetJumpZ(float F);
void Slomo(float T);
void God();
void Ghost();
void Walk();
void Fly();
void EndPath();
void ChangeSize(float F);
void Teleport();
void KillViewedActor();
void WriteToLog(class FString Param);
void FreezeFrame(float Delay);
void ListDynamicActors();
void DebugPause();
void EditAIByTrace();
void DebugAI(struct FName Category);
void FXStop(class UClass* aClass);
void FXPlay(class UClass* aClass, class FString FXAnimPath);
};

// Class Engine.Client
// 0x0018 (0x0060 - 0x0078)
class UClient : public UObject
{
public:
uint8_t                              UnknownData00[0x8];                    // 0x0060 (0x0008) MISSED
OFFSET
float                                MinDesiredFrameRate;                    // 0x0068 (0x0004)
[0x0000000000004000] (CPF_Config)
float                                DisplayGamma;                        // 0x006C (0x0004)
[0x0000000000004000] (CPF_Config)
float                                InitialButtonRepeatDelay;                // 0x0070 (0x0004)
[0x0000000000004000] (CPF_Config)
float                                ButtonRepeatDelay;                     // 0x0074 (0x0004)
[0x0000000000004000] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Client");
}

return uClassPointer;
};

};

// Class Engine.ClipPadEntry
// 0x0020 (0x0060 - 0x0080)
class UClipPadEntry : public UObject
{
public:
```

```
class FString                              Title;                                  // 0x0060 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                              Text;                                   // 0x0070 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ClipPadEntry");
}

return uClassPointer;
};

};

// Class Engine.CloudSaveSystem
// 0x00A0 (0x0060 - 0x0100)
class UCloudSaveSystem : public UObject
{
public:
class UCloudSaveSystemKVSInterface*             KeyValueStore_Object;                      // 0x0060
(0x0008) [0x0000000000002000] (CPF_Transient)
class UCloudSaveSystemKVSInterface*             KeyValueStore_Interface;                   // 0x0068
(0x0008) [0x0000000000002000] (CPF_Transient)
class UCloudSaveSystemDataBlobStoreInterface*     DataBlobStore_Object;                        //
0x0070 (0x0008) [0x0000000000002000] (CPF_Transient)
class UCloudSaveSystemDataBlobStoreInterface*     DataBlobStore_Interface;                     //
0x0078 (0x0008) [0x0000000000002000] (CPF_Transient)
TArray<struct FGetSaveDataCallbackStruct>        OnGetSaveDataCallbacks;                    //
0x0080 (0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FSetSaveDataCallbackStruct>        OnSetSaveDataCallbacks;                    //
0x0090 (0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FScriptDelegate                          DeleteSaveDataCallback;                  // 0x00A0 (0x0018)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                                      ActiveSlotForDelete;                      // 0x00B8 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FSaveSlotOperation>               ActiveSaveSlotOperations;                  // 0x00C0
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FScriptDelegate                          __OnGetSaveDataCallback__Delegate;           // 0x00D0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                          __SaveSystemCallback__Delegate;             // 0x00E8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CloudSaveSystem");
}

return uClassPointer;
};

bool GetKeyValue(int32_t SaveSlot, class FString KeyName, uint8_t Type, struct
FPlatformInterfaceDelegateResult& Value);
bool GetDataStoreIDAndBlobNameForSaveSlot(int32_t SaveSlot, class FString& DataStoreID,
class FString& DataBlobName);
bool SetKeyValue(int32_t SaveSlot, class FString KeyName, struct FPlatformInterfaceData&
Value);
bool InternalSetSaveSlotKeyValues(int32_t SaveSlot, class FString DataStoreID, class FString
SaveDataBlobName);
bool SetSaveSlotKeyValues(class FString DataStoreID, class FString SaveDataBlobName,
int32_t& SaveSlot);
void OnDeleteSaveDataComplete(unsigned long bWasSucessfull, class FString StorageID, class
FString BlobName, class FString Error);
bool DeleteSaveData(int32_t SaveSlot, struct FScriptDelegate InDeleteSaveDataCallback);
void OnSetSaveDataComplete(unsigned long bWasSucessfull, class FString StorageID, class
FString BlobName, class FString Error);
void SetSaveData(int32_t SaveSlot, struct FScriptDelegate InSetSaveDataCallback,
TArray<uint8_t>& SaveDataBlob);
void OnGetSaveDataComplete(unsigned long bWasSuccessful, class FString StorageID, class
FString BlobName, class FString Error, TArray<uint8_t>& DataBlob);
void GetSaveData(int32_t SaveSlot, struct FScriptDelegate OnGetSaveDataCallback);
void Init(class UCloudSaveSystemKVSInterface* InKeyValueStore, class
UCloudSaveSystemDataBlobStoreInterface* InDataBlobStore, int32_t VersionNumber);
class UObject* DeserializeObject(class UClass* ObjectClass, uint8_t VersionSupport, int32_t
DataVersion, TArray<uint8_t>& Data);
void SerializeObject(class UObject* ObjectToSerialize, int32_t DataVersion, TArray<uint8_t>&
Data);
bool AreAnySlotOperationsActive();
bool IsDeleteOperationActive();
bool IsOperationActiveForSlot(int32_t SlotIndex);
bool WriteNumSaveSlots(int32_t NumSaveSlots);
int32_t DoesSaveSlotKeyValueDataAlreadyExist(class FString DataStoreID, class FString
DataBlobName);
bool GetNumberOfSaveSlots(int32_t& NumSaveSlots);
void SaveSystemCallback(unsigned long bWasSuccessful, int32_t SaveSlot, class FString Error);
void OnGetSaveDataCallback(unsigned long bWasSuccessful, int32_t SaveSlot, class FString
Error, TArray<uint8_t>& DataBlob);
};

// Class Engine.CodecMovie
// 0x0004 (0x0060 - 0x0064)
class UCodecMovie : public UObject
{
public:
float                                   PlaybackDuration;                        // 0x0060 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CodecMovie");
}

return uClassPointer;
};

};

// Class Engine.CodecMovieBink
// 0x0004 (0x0064 - 0x0068)
class UCodecMovieBink : public UCodecMovie
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CodecMovieBink");
}

return uClassPointer;
};

};

// Class Engine.CodecMovieFallback
// 0x0008 (0x0064 - 0x006C)
class UCodecMovieFallback : public UCodecMovie
{
public:
float                                   CurrentTime;                            // 0x0068 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CodecMovieFallback");
}

return uClassPointer;
};

};

// Class Engine.ControllerLayoutStack
// 0x0048 (0x0060 - 0x00A8)
class UControllerLayoutStack : public UObject
{
public:
struct FName                        CurrentLayout;                      // 0x0060 (0x0008)
[0x0001004000002000] (CPF_Transient)
TArray<struct FControllerLayout>            LayoutStack;                       // 0x0068 (0x0010)
[0x0001000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FName                        ControllerLayout_Gameplay;               // 0x0078 (0x0008)
[0x0001000000000002] (CPF_Const)
struct FName                        ControllerLayout_Menu;                  // 0x0080 (0x0008)
[0x0001000000000002] (CPF_Const)
struct FName                        ControllerLayout_ReplayViewer;             // 0x0088 (0x0008)
[0x0001000000000002] (CPF_Const)
struct FName                        ControllerLayout_Spectator;               // 0x0090 (0x0008)
[0x0001000000000002] (CPF_Const)
struct FName                        ControllerLayout_TrainingEditor;            // 0x0098 (0x0008)
[0x0001000000000002] (CPF_Const)
int32_t                         ControllerLayoutPriority_Default;           // 0x00A0 (0x0004)
[0x0001000000000002] (CPF_Const)
int32_t                         ControllerLayoutPriority_Menu;             // 0x00A4 (0x0004)
[0x0001000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ControllerLayoutStack");
}

return uClassPointer;
};

void Empty();
void Pop(int32_t ControllerId, struct FName InControllerLayout);
void Push(int32_t ControllerId, struct FName InControllerLayout, int32_t Priority);
};

// Class Engine.CurveEdPresetCurve
// 0x0020 (0x0060 - 0x0080)
class UCurveEdPresetCurve : public UObject
{
public:
class FString                       CurveName;                        // 0x0060 (0x0010)
[0x0000000000408003] (CPF_Edit | CPF_Const | CPF_Localized | CPF_NeedCtorLink)
TArray<struct FPresetGeneratedPoint>          Points;                          // 0x0070 (0x0010)
```

```
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CurveEdPresetCurve");
}

return uClassPointer;
};

};

// Class Engine.CustomPropertyItemHandler
// 0x0000 (0x0060 - 0x0060)
class UCustomPropertyItemHandler : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CustomPropertyItemHandler");
}

return uClassPointer;
};

};

// Class Engine.DateTime
// 0x0025 (0x0060 - 0x0085)
class UDateTime : public UObject
{
public:
uint64_t                                   EpochTime;                              // 0x0060 (0x0008)
[0x0000004040000000] (CPF_EditInlineNotify)
int32_t                                    Year;                                   // 0x0068 (0x0004)
[0x0000004040000000] (CPF_EditInlineNotify)
int32_t                                    Month;                                  // 0x006C (0x0004)
[0x0000004040000000] (CPF_EditInlineNotify)
int32_t                                    Day;                                    // 0x0070 (0x0004)
[0x0000004040000000] (CPF_EditInlineNotify)
int32_t                                    WeekDay;                                // 0x0074 (0x0004)
[0x0000004040000000] (CPF_EditInlineNotify)
int32_t                                    Hour;                                   // 0x0078 (0x0004)
```

```
[0x0000004040000000] (CPF_EditInlineNotify)
int32_t                                 Minute;                                 // 0x007C (0x0004)
[0x0000004040000000] (CPF_EditInlineNotify)
int32_t                                 Second;                                 // 0x0080 (0x0004)
[0x0000004040000000] (CPF_EditInlineNotify)
uint8_t                                 TimeZone;                               // 0x0084 (0x0001)
[0x0000004040000000] (CPF_EditInlineNotify)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DateTime");
}

return uClassPointer;
};

class UDateTime* AddSeconds(int32_t Amount);
class UDateTime* AddMinutes(int32_t Amount);
class UDateTime* AddHours(int32_t Amount);
class UDateTime* AddDays(int32_t Amount);
class UDateTime* AddMonths(int32_t Amount);
class UDateTime* AddYears(int32_t Amount);
class UDateTime* ToTimeZone(uint8_t InTimeZone);
class UDateTime* ToUTC();
class UDateTime* ToLocal();
class FString ToString();
struct FDateTimeStruct ToStruct();
static class UDateTime* FromStruct(struct FDateTimeStruct Data);
static class UDateTime* FromDateTime(int32_t InYear, int32_t InMonth, int32_t InDay, int32_t
InHour, int32_t InMinute, int32_t InSecond, uint8_t InTimeZone);
static class UDateTime* FromString(class FString TimeStamp, uint8_t InTimeZone);
static class UDateTime* FromISO8601(class FString TimeStamp);
static class UDateTime* FromEpochTime(uint64_t InEpochTime);
static class UDateTime* Now();
static uint64_t EpochNow();
};

// Class Engine.DistributionFloatConstant
// 0x0008 (0x007C - 0x0084)
class UDistributionFloatConstant : public UDistributionFloat
{
public:
float                                   Constant;                               // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatConstant");
}

return uClassPointer;
};

};

// Class Engine.DistributionFloatParameterBase
// 0x001D (0x0084 - 0x00A1)
class UDistributionFloatParameterBase : public UDistributionFloatConstant
{
public:
struct FName                        ParameterName;                    // 0x0088 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                               MinInput;                         // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               MaxInput;                         // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               MinOutput;                        // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               MaxOutput;                        // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                             ParamMode;                        // 0x00A0 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatParameterBase");
}

return uClassPointer;
};

};

// Class Engine.DistributionFloatConstantCurve
// 0x001C (0x007C - 0x0098)
class UDistributionFloatConstantCurve : public UDistributionFloat
{
public:
struct FInterpCurveFloat            ConstantCurve;                    // 0x0080 (0x0018)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatConstantCurve");
}

return uClassPointer;
};

};

// Class Engine.DistributionFloatUniform
// 0x000C (0x007C - 0x0088)
class UDistributionFloatUniform : public UDistributionFloat
{
public:
float                                    Min;                                    // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    Max;                                    // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatUniform");
}

return uClassPointer;
};

};

// Class Engine.DistributionFloatUniformCurve
// 0x001C (0x007C - 0x0098)
class UDistributionFloatUniformCurve : public UDistributionFloat
{
public:
struct FInterpCurveVector2D                    ConstantCurve;                              // 0x0080 (0x0018)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatUniformCurve");
}

return uClassPointer;
};

};

// Class Engine.DistributionFloatUniformRange
// 0x0018 (0x007C - 0x0094)
class UDistributionFloatUniformRange : public UDistributionFloat
{
public:
float                         MaxHigh;                     // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         MaxLow;                      // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         MinHigh;                     // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         MinLow;                      // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                      bMirrorMaxMin : 1;                // 0x0090 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatUniformRange");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorConstant
// 0x0015 (0x007C - 0x0091)
class UDistributionVectorConstant : public UDistributionVector
{
public:
struct FVector                     Constant;                     // 0x0080 (0x000C)
[0x0000000000000001] (CPF_Edit)
unsigned long                      bLockAxes : 1;                  // 0x008C (0x0004)
[0x0000000000000000] [0x00000001]
uint8_t                         LockedAxes;                   // 0x0090 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorConstant");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorParameterBase
// 0x0042 (0x0091 - 0x00D3)
class UDistributionVectorParameterBase : public UDistributionVectorConstant
{
public:
struct FName                              ParameterName;                          // 0x0098 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                            MinInput;                              // 0x00A0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                            MaxInput;                              // 0x00AC (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                            MinOutput;                             // 0x00B8 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                            MaxOutput;                             // 0x00C4 (0x000C)
[0x0000000000000001] (CPF_Edit)
uint8_t                                ParamModes[0x3];                          // 0x00D0 (0x0003)
[0x0000000000080009] (CPF_Edit | CPF_ExportObject | CPF_Component)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorParameterBase");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorConstantCurveBase
// 0x0021 (0x007C - 0x009D)
class UDistributionVectorConstantCurveBase : public UDistributionVector
{
public:
struct FInterpCurveVector                 ConstantCurve;                          // 0x0080 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                             bLockAxes : 1;                          // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]
uint8_t                                LockedAxes;                               // 0x009C (0x0001)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorConstantCurveBase");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorConstantCurve
// 0x001B (0x009D - 0x00B8)
class UDistributionVectorConstantCurve : public UDistributionVectorConstantCurveBase
{
public:
struct FInterpCurveVector                    Curve;                              // 0x00A0 (0x0018)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorConstantCurve");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorUniform
// 0x0024 (0x007C - 0x00A0)
class UDistributionVectorUniform : public UDistributionVector
{
public:
struct FVector                               Max;                                // 0x0080 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                               Min;                                // 0x008C (0x000C)
[0x0000000000000001] (CPF_Edit)
unsigned long                                bLockAxes : 1;                      // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                                bUseExtremes : 1;                   // 0x0098 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
uint8_t                                   LockedAxes;                            // 0x009C (0x0001)
```

```
[0x0000000000000001] (CPF_Edit)
uint8_t                              MirrorFlags[0x3];                         // 0x009D (0x0003)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorUniform");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorUniformCurve
// 0x0025 (0x007C - 0x00A1)
class UDistributionVectorUniformCurve : public UDistributionVector
{
public:
struct FInterpCurveTwoVectors                  ConstantCurve;                            // 0x0080 (0x0018)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                        bLockAxes1 : 1;                           // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                        bLockAxes2 : 1;                           // 0x0098 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                        bUseExtremes : 1;                          // 0x0098 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
uint8_t                              LockedAxes[0x2];                          // 0x009C (0x0002)
[0x0000000000000001] (CPF_Edit)
uint8_t                              MirrorFlags[0x3];                         // 0x009E (0x0003)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorUniformCurve");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorUniformRange
// 0x0034 (0x007C - 0x00B0)
```

```
class UDistributionVectorUniformRange : public UDistributionVector
{
public:
struct FVector                          MaxHigh;                              // 0x0080 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          MaxLow;                               // 0x008C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          MinHigh;                             // 0x0098 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          MinLow;                              // 0x00A4 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorUniformRange");
}

return uClassPointer;
};

};

// Class Engine.Download
// 0x0A50 (0x0060 - 0x0AB0)
class UDownload : public UObject
{
public:
uint8_t                             UnknownData00[0xA50];                     // 0x0060 (0x0A50)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Download");
}

return uClassPointer;
};

};

// Class Engine.ChannelDownload
// 0x0008 (0x0AB0 - 0x0AB8)
class UChannelDownload : public UDownload
{
```

```
public:
uint8_t                         UnknownData00[0x8];                 // 0x0AB0 (0x0008) MISSED
OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ChannelDownload");
}

return uClassPointer;
};

};

// Class Engine.EdCoordSystem
// 0x0050 (0x0060 - 0x00B0)
class UEdCoordSystem : public UObject
{
public:
struct FMatrix                          M;                          // 0x0060 (0x0040)
[0x0000000000000001] (CPF_Edit)
class FString                           Desc;                       // 0x00A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EdCoordSystem");
}

return uClassPointer;
};

};

// Class Engine.EditorLinkSelectionInterface
// 0x0000 (0x0060 - 0x0060)
class UEditorLinkSelectionInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EditorLinkSelectionInterface");
}

return uClassPointer;
};

};

// Class Engine.EngineShare
// 0x0000 (0x0060 - 0x0060)
class UEngineShare : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EngineShare");
}

return uClassPointer;
};

void eventInit();
static class FString GetPsyVersionNumber();
static class UEngineShare* GetInstance(class UClass* Type);
};

// Class Engine.EngineTypes
// 0x0000 (0x0060 - 0x0060)
class UEngineTypes : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EngineTypes");
}

return uClassPointer;
};
```

```
};

// Class Engine.FaceFXAnimSet
// 0x0054 (0x0060 - 0x00B4)
class UFaceFXAnimSet : public UObject
{
public:
class UFaceFXAsset*                    DefaultFaceFXAsset;                   // 0x0060 (0x0008)
[0x0000000800000003] (CPF_Edit | CPF_Const)
struct FPointer                        InternalFaceFXAnimSet;                // 0x0068 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<uint8_t>                        RawFaceFXAnimSetBytes;                // 0x0070 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<uint8_t>                        RawFaceFXMiniSessionBytes;            // 0x0080 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<class USoundCue*>               ReferencedSoundCues;                  // 0x0090
(0x0010) [0x0000001800400000] (CPF_NeedCtorLink)
TArray<class UAkEvent*>                ReferencedAkEvents;                   // 0x00A0
(0x0010) [0x0000001800400000] (CPF_NeedCtorLink)
int32_t                                NumLoadErrors;                        // 0x00B0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FaceFXAnimSet");
}

return uClassPointer;
};

};

// Class Engine.FaceFXAsset
// 0x0074 (0x0060 - 0x00D4)
class UFaceFXAsset : public UObject
{
public:
class USkeletalMesh*                   DefaultSkelMesh;                      // 0x0060 (0x0008)
[0x0000000800000002] (CPF_Const)
struct FPointer                        FaceFXActor;                          // 0x0068 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<uint8_t>                        RawFaceFXActorBytes;                  // 0x0070 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<uint8_t>                        RawFaceFXSessionBytes;                // 0x0080 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<class UMorphTargetSet*>         PreviewMorphSets;                     // 0x0090
(0x0010) [0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UFaceFXAnimSet*>          MountedFaceFXAnimSets;                // 0x00A0
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
```

```
TArray<class USoundCue*>                    ReferencedSoundCues;                    // 0x00B0
(0x0010) [0x0000001800400000] (CPF_NeedCtorLink)
TArray<class UAkEvent*>                     ReferencedAkEvents;                     // 0x00C0
(0x0010) [0x0000001800400000] (CPF_NeedCtorLink)
int32_t                                     NumLoadErrors;                          // 0x00D0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FaceFXAsset");
}

return uClassPointer;
};

void UnmountFaceFXAnimSet(class UFaceFXAnimSet* AnimSet);
void MountFaceFXAnimSet(class UFaceFXAnimSet* AnimSet);
};

// Class Engine.FeaturePrivilegeErrors
// 0x0030 (0x0080 - 0x00B0)
class UFeaturePrivilegeErrors : public UErrorList
{
public:
class UErrorType*                           FPE_AgeRestriction;                     // 0x0080 (0x0008)
[0x0000000000000002] (CPF_Const)
class UErrorType*                           FPE_ParentalLock;                       // 0x0088 (0x0008)
[0x0000000000000002] (CPF_Const)
class UErrorType*                           FPE_OnlineAccount;                      // 0x0090 (0x0008)
[0x0000000000000002] (CPF_Const)
class UErrorType*                           FPE_PremiumService;                     // 0x0098 (0x0008)
[0x0000000000000002] (CPF_Const)
class UErrorType*                           FPE_TooManyLocalUsers;                  // 0x00A0 (0x0008)
[0x0000000000000002] (CPF_Const)
class UErrorType*                           FPE_Unknown;                            // 0x00A8 (0x0008)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FeaturePrivilegeErrors");
}

return uClassPointer;
};
```

```
};

// Class Engine.Font
// 0x015C (0x0060 - 0x01BC)
class UFont : public UObject
{
public:
TArray<struct FFontCharacter>                    Characters;                              // 0x0060 (0x0010)
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UTexture2D*>                       Textures;                                // 0x0070 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                                UnknownData00[0x50];                    // 0x0080 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.Font.CharRemap
int32_t                                IsRemapped;                             // 0x00D0 (0x0004)
[0x0000000000000000]
float                                  EmScale;                               // 0x00D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  Ascent;                                // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  Descent;                               // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  Leading;                               // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                Kerning;                               // 0x00E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FFontImportOptionsData                    ImportOptions;                           // 0x00E8 (0x00B0)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                                NumCharacters;                          // 0x0198 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<int32_t>                                MaxCharHeight;                         // 0x01A0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                                  ScalingFactor;                          // 0x01B0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                              bMapUppercaseToLowercase : 1;              // 0x01B4
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                              bMapLowercaseToUppercase : 1;              // 0x01B4
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
int32_t                                VerticalOffsetOverride;                   // 0x01B8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Font");
}

return uClassPointer;
};
```

```
void GetStringHeightAndWidth(class FString& InString, int32_t& Height, int32_t& Width);
float GetMaxCharHeight();
float GetAuthoredViewportHeight(float ViewportHeight);
float GetScalingFactor(float HeightTest);
int32_t GetResolutionPageIndex(float HeightTest);
};

// Class Engine.MultiFont
// 0x0014 (0x01BC - 0x01D0)
class UMultiFont : public UFont
{
public:
TArray<float>                    ResolutionTestTable;              // 0x01C0 (0x0010)
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MultiFont");
}

return uClassPointer;
};

int32_t GetResolutionTestTableIndex(float HeightTest);
};

// Class Engine.FontImportOptions
// 0x00B0 (0x0060 - 0x0110)
class UFontImportOptions : public UObject
{
public:
struct FFontImportOptionsData          Data;                          // 0x0060 (0x00B0)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FontImportOptions");
}

return uClassPointer;
};

};
```

```
// Class Engine.ForceFeedbackManager
// 0x0038 (0x0060 - 0x0098)
class UForceFeedbackManager : public UObject
{
public:
unsigned long                          bAllowsForceFeedback : 1;              // 0x0060 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                          bIsPaused : 1;                         // 0x0060 (0x0004)
[0x0000000000000000] [0x00000002]
class UForceFeedbackWaveform*                    FFWaveform;                            // 0x0068
(0x0008) [0x0000000000000000]
int32_t                            CurrentSample;                         // 0x0070 (0x0004)
[0x0000000000000000]
float                              ElapsedTime;                           // 0x0074 (0x0004)
[0x0000000000000000]
float                              ScaleAllWaveformsBy;                   // 0x0078 (0x0004)
[0x0000000000000000]
class AActor*                          WaveformInstigator;                    // 0x0080 (0x0008)
[0x0000000000000000]
int32_t                            ShakeLeft;                             // 0x0088 (0x0004)
[0x0000000000000000]
int32_t                            ShakeRight;                            // 0x008C (0x0004)
[0x0000000000000000]
float                              EnableTime;                            // 0x0090 (0x0004)
[0x0000000000000000]
float                              EnabledTimeRemaining;                  // 0x0094 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForceFeedbackManager");
}

return uClassPointer;
};

void PauseWaveform(unsigned long bPause);
void StopForceFeedbackWaveform(class UForceFeedbackWaveform* WaveForm);
void PlayForceFeedbackWaveform(class UForceFeedbackWaveform* WaveForm, class AActor*
WaveInstigator);
};

// Class Engine.ForceFeedbackWaveform
// 0x0020 (0x0060 - 0x0080)
class UForceFeedbackWaveform : public UObject
{
public:
unsigned long                          bIsLooping : 1;                        // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
TArray<struct FWaveformSample>          Samples;                        // 0x0068 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                   WaveformFalloffStartDistance;          // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MaxWaveformDistance;                   // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForceFeedbackWaveform");
}

return uClassPointer;
};

};

// Class Engine.GameplayEvents
// 0x01B0 (0x0060 - 0x0210)
class UGameplayEvents : public UObject
{
public:
struct FPointer                         Archive;                        // 0x0060 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
class FString                           StatsFileName;                  // 0x0068 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FGameplayEventsHeader            Header;                         // 0x0078 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FGameSessionInformation          CurrentSessionInfo;                    // 0x00B0
(0x00D0) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FPlayerInformation>       PlayerList;                     // 0x0180 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FTeamInformation>         TeamList;                       // 0x0190 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FGameplayEventMetaData>       SupportedEvents;                   // 0x01A0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FWeaponClassEventData>        WeaponClassArray;                  // 0x01B0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FDamageClassEventData>        DamageClassArray;                  // 0x01C0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FProjectileClassEventData>    ProjectileClassArray;              // 0x01D0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FPawnClassEventData>          PawnClassArray;                    // 0x01E0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<class FString>                   ActorArray;                     // 0x01F0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class FString>                   SoundCueArray;                  // 0x0200 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameplayEvents");
}

return uClassPointer;
};

class FString eventGetFilename();
void CloseStatsFile();
bool OpenStatsFile(class FString Filename);
};

// Class Engine.GameplayEventsReader
// 0x0010 (0x0210 - 0x0220)
class UGameplayEventsReader : public UGameplayEvents
{
public:
TArray<class UGameplayEventsHandler*>            RegisteredHandlers;                          // 0x0210
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameplayEventsReader");
}

return uClassPointer;
};

float GetSessionDuration();
float GetSessionEnd();
float GetSessionStart();
class FString GetSessionTimestamp();
int32_t GetPlatform();
int32_t GetTitleID();
class FString GetSessionID();
void ProcessStreamEnd();
void ProcessStream();
void ProcessStreamStart();
void eventUnregisterHandler(class UGameplayEventsHandler* ExistingHandler);
void eventRegisterHandler(class UGameplayEventsHandler* NewHandler);
bool SerializeHeader();
void CloseStatsFile();
bool OpenStatsFile(class FString Filename);
```

```
};

// Class Engine.GameplayEventsWriterBase
// 0x0008 (0x0210 - 0x0218)
class UGameplayEventsWriterBase : public UGameplayEvents
{
public:
class AGameInfo*                    Game;                         // 0x0210 (0x0008)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameplayEventsWriterBase");
}

return uClassPointer;
};

int32_t RecordCoverLinkFireLinks(class ACoverLink* Link, class AController* Player);
void RecordAIPathFail(class AController* AI, class FString Reason, struct FVector Dest);
void LogSystemPollEvents();
void LogPlayerPlayerEvent(int32_t EventID, class AController* Player, class AController* Target);
void LogAllPlayerPositionsEvent(int32_t EventID);
void LogPlayerLoginChange(int32_t EventID, class AController* Player, class FString PlayerName,
struct FUniqueNetId PlayerID, unsigned long bSplitScreen);
void LogPlayerSpawnEvent(int32_t EventID, class AController* Player, class UClass* PawnClass,
int32_t TeamID);
void LogPlayerStringEvent(int32_t EventID, class AController* Player, class FString EventString);
void LogPlayerFloatEvent(int32_t EventID, class AController* Player, float Value);
void LogPlayerIntEvent(int32_t EventID, class AController* Player, int32_t Value);
void LogTeamStringEvent(int32_t EventID, class ATeamInfo* Team, class FString Value);
void LogTeamFloatEvent(int32_t EventID, class ATeamInfo* Team, float Value);
void LogTeamIntEvent(int32_t EventID, class ATeamInfo* Team, int32_t Value);
void LogGamePositionEvent(int32_t EventID, float Value, struct FVector& Position);
void LogGameFloatEvent(int32_t EventID, float Value);
void LogGameStringEvent(int32_t EventID, class FString Value);
void LogGameIntEvent(int32_t EventID, int32_t Value);
void EndLogging();
void ResetLogging(float HeartbeatDelta);
void StartLogging(float HeartbeatDelta);
int32_t eventGetPlaylistId();
int32_t eventGetGameTypeId();
void Poll();
void eventStopPolling();
void eventStartPolling(float HearbeatDelta);
bool IsSessionInProgress();
};

// Class Engine.GameplayEventsUploadAnalytics
// 0x0000 (0x0218 - 0x0218)
class UGameplayEventsUploadAnalytics : public UGameplayEventsWriterBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameplayEventsUploadAnalytics");
}

return uClassPointer;
};

int32_t RecordCoverLinkFireLinks(class ACoverLink* Link, class AController* Player);
void RecordAIPathFail(class AController* AI, class FString Reason, struct FVector Dest);
class UGenericParamListStatEntry* GetGenericParamListEntry();
void LogPlayerPlayerEvent(int32_t EventID, class AController* Player, class AController* Target);
void LogAllPlayerPositionsEvent(int32_t EventID);
void LogPlayerLoginChange(int32_t EventID, class AController* Player, class FString PlayerName,
struct FUniqueNetId PlayerID, unsigned long bSplitScreen);
void LogPlayerSpawnEvent(int32_t EventID, class AController* Player, class UClass* PawnClass,
int32_t TeamID);
void LogPlayerStringEvent(int32_t EventID, class AController* Player, class FString EventString);
void LogPlayerFloatEvent(int32_t EventID, class AController* Player, float Value);
void LogPlayerIntEvent(int32_t EventID, class AController* Player, int32_t Value);
void LogTeamStringEvent(int32_t EventID, class ATeamInfo* Team, class FString Value);
void LogTeamFloatEvent(int32_t EventID, class ATeamInfo* Team, float Value);
void LogTeamIntEvent(int32_t EventID, class ATeamInfo* Team, int32_t Value);
void LogGamePositionEvent(int32_t EventID, float Value, struct FVector& Position);
void LogGameFloatEvent(int32_t EventID, float Value);
void LogGameStringEvent(int32_t EventID, class FString Value);
void LogGameIntEvent(int32_t EventID, int32_t Value);
void EndLogging();
void ResetLogging(float HeartbeatDelta);
void StartLogging(float HeartbeatDelta);
};

// Class Engine.GameplayEventsWriter
// 0x0000 (0x0218 - 0x0218)
class UGameplayEventsWriter : public UGameplayEventsWriterBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.GameplayEventsWriter");
}

return uClassPointer;
};

int32_t RecordCoverLinkFireLinks(class ACoverLink* Link, class AController* Player);
void RecordAIPathFail(class AController* AI, class FString Reason, struct FVector Dest);
class UGenericParamListStatEntry* GetGenericParamListEntry();
void LogSystemPollEvents();
void LogPlayerPlayerEvent(int32_t EventID, class AController* Player, class AController* Target);
void LogAllPlayerPositionsEvent(int32_t EventID);
void LogPlayerLoginChange(int32_t EventID, class AController* Player, class FString PlayerName,
struct FUniqueNetId PlayerID, unsigned long bSplitScreen);
void LogPlayerSpawnEvent(int32_t EventID, class AController* Player, class UClass* PawnClass,
int32_t TeamID);
void LogPlayerStringEvent(int32_t EventID, class AController* Player, class FString EventString);
void LogPlayerFloatEvent(int32_t EventID, class AController* Player, float Value);
void LogPlayerIntEvent(int32_t EventID, class AController* Player, int32_t Value);
void LogTeamStringEvent(int32_t EventID, class ATeamInfo* Team, class FString Value);
void LogTeamFloatEvent(int32_t EventID, class ATeamInfo* Team, float Value);
void LogTeamIntEvent(int32_t EventID, class ATeamInfo* Team, int32_t Value);
void LogGamePositionEvent(int32_t EventID, float Value, struct FVector& Position);
void LogGameFloatEvent(int32_t EventID, float Value);
void LogGameStringEvent(int32_t EventID, class FString Value);
void LogGameIntEvent(int32_t EventID, int32_t Value);
void EndLogging();
void ResetLogging(float HeartbeatDelta);
void StartLogging(float HeartbeatDelta);
bool SerializeFooter();
bool SerializeHeader();
void CloseStatsFile();
bool OpenStatsFile(class FString Filename);
int32_t ResolvePlayerIndex(class AController* Player);
};

// Class Engine.GameplayEventsHandler
// 0x0028 (0x0060 - 0x0088)
class UGameplayEventsHandler : public UObject
{
public:
TArray<int32_t>                    EventIDFilter;                 // 0x0060 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<struct FGameStatGroup>                GroupFilter;                  // 0x0070 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UGameplayEventsReader*                 Reader;                       // 0x0080 (0x0008)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameplayEventsHandler");
}

return uClassPointer;
};

void RemoveFilter(int32_t EventID);
void AddFilter(int32_t EventID);
void eventResolveGroupFilters();
void eventPostProcessStream();
void eventPreProcessStream();
void SetReader(class UGameplayEventsReader* NewReader);
};

// Class Engine.GenericParamListStatEntry
// 0x0010 (0x0060 - 0x0070)
class UGenericParamListStatEntry : public UObject
{
public:
struct FPointer                    StatEvent;                    // 0x0060 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)
class UGameplayEventsWriter*            Writer;                       // 0x0068 (0x0008)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GenericParamListStatEntry");
}

return uClassPointer;
};

void CommitToDisk();
bool GetString(struct FName ParamName, class FString& out_string);
bool GetVector(struct FName ParamName, struct FVector& out_vector);
bool GetInt(struct FName ParamName, int32_t& out_int);
bool GetFloat(struct FName ParamName, float& out_Float);
void AddString(struct FName ParamName, class FString Value);
void AddVector(struct FName ParamName, struct FVector Value);
void AddInt(struct FName ParamName, int32_t Value);
void AddFloat(struct FName ParamName, float Value);
};

// Class Engine.GuidCache
// 0x0060 (0x0060 - 0x00C0)
class UGuidCache : public UObject
{
```

```
public:
uint8_t                         UnknownData00[0x60];                // 0x0060 (0x0060)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GuidCache");
}

return uClassPointer;
};

};

// Class Engine.HttpBaseInterface
// 0x0000 (0x0060 - 0x0060)
class UHttpBaseInterface : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HttpBaseInterface");
}

return uClassPointer;
};

void GetContent(TArray<uint8_t>& Content);
class FString GetURL();
int32_t GetContentLength();
class FString GetContentType();
class FString GetURLParameter(class FString ParameterName);
TArray<class FString> GetHeaders();
class FString GetHeader(class FString HeaderName);
};

// Class Engine.HttpRequestInterface
// 0x0018 (0x0060 - 0x0078)
class UHttpRequestInterface : public UHttpBaseInterface
{
public:
struct FScriptDelegate                  __OnProcessRequestComplete__Delegate;        // 0x0060
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HttpRequestInterface");
}

return uClassPointer;
};

class UHttpRequestInterface* SetProcessRequestCompleteDelegate(struct FScriptDelegate
ProcessRequestCompleteDelegate);
void OnProcessRequestComplete(class UHttpRequestInterface* OriginalRequest, class
UHttpResponseInterface* InHttpResponse, unsigned long bDidSucceed);
bool ProcessRequest();
class UHttpRequestInterface* SetHeader(class FString HeaderName, class FString
HeaderValue);
class UHttpRequestInterface* SetContentAsString(class FString ContentString);
class UHttpRequestInterface* SetContent(TArray<uint8_t>& ContentPayload);
class UHttpRequestInterface* SetURL(class FString URL);
class UHttpRequestInterface* SetVerb(class FString Verb);
class FString GetVerb();
};

// Class Engine.HttpResponseInterface
// 0x0000 (0x0060 - 0x0060)
class UHttpResponseInterface : public UHttpBaseInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HttpResponseInterface");
}

return uClassPointer;
};

bool IsContentCompressed();
class FString GetContentAsString();
int32_t GetResponseCode();
};

// Class Engine.IniLocPatcher
// 0x0078 (0x0060 - 0x00D8)
```

```
class UIniLocPatcher : public UObject
{
public:
TArray<struct FIniLocFileEntry>                  Files;                                    // 0x0060 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
unsigned long                               bRequestEmsFileList : 1;                  // 0x0070 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
int32_t                                   MaxCachedFileAge;                         // 0x0074 (0x0004)
[0x0000000000004000] (CPF_Config)
class UOnlineTitleFileInterface*                 TitleFileInterface_Object;                  // 0x0078
(0x0008) [0x0000000000002000] (CPF_Transient)
class UOnlineTitleFileInterface*                 TitleFileInterface_Interface;               // 0x0080
(0x0008) [0x0000000000002000] (CPF_Transient)
class UOnlineTitleFileCacheInterface*            TitleFileCacheInterface_Object;              // 0x0088
(0x0008) [0x0000000000002000] (CPF_Transient)
class UOnlineTitleFileCacheInterface*            TitleFileCacheInterface_Interface;           // 0x0090
(0x0008) [0x0000000000002000] (CPF_Transient)
TArray<struct FScriptDelegate>                   ReadTitleFileCompleteDelegates;              // 0x0098
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                           __OnReadTitleFileComplete__Delegate;          // 0x00A8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                           __OnAllTitleFilesCompleted__Delegate;         // 0x00C0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.IniLocPatcher");
}

return uClassPointer;
};

class FString UpdateLocFileName(class FString Filename);
void ClearCachedFiles();
void ClearReadFileDelegate(struct FScriptDelegate ReadTitleFileCompleteDelegate);
void AddReadFileDelegate(struct FScriptDelegate ReadTitleFileCompleteDelegate);
void AddFileToDownload(class FString Filename);
void ProcessIniLocFile(class FString Filename, unsigned long bIsUnicode, TArray<uint8_t>&
FileData);
void CheckForAllFilesComplete();
void TriggerDownloadCompleteDelegates(unsigned long bSuccess, class FString Filename);
void OnFileCacheSaveComplete(unsigned long bWasSuccessful, class FString Filename);
void OnFileCacheLoadComplete(unsigned long bWasSuccessful, class FString Filename);
void OnDownloadFileComplete(unsigned long bWasSuccessful, class FString Filename);
void StartLoadingFiles();
void OnRequestTitleFileListComplete(unsigned long bWasSuccessful, TArray<class FString>
ResultStr);
void DownloadFiles();
void Init();
void OnAllTitleFilesCompleted();
void OnReadTitleFileComplete(unsigned long bWasSuccessful, class FString Filename);
};

// Class Engine.Interface_NavigationHandle
// 0x0000 (0x0060 - 0x0060)
class UInterface_NavigationHandle : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Interface_NavigationHandle");
}

return uClassPointer;
};

void eventNotifyPathChanged();
};

// Class Engine.Interface_Speaker
// 0x0000 (0x0060 - 0x0060)
class UInterface_Speaker : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Interface_Speaker");
}

return uClassPointer;
};

void eventSpeak(class USoundCue* Cue);
};

// Class Engine.InterpCurveEdSetup
// 0x0014 (0x0060 - 0x0074)
class UInterpCurveEdSetup : public UObject
{
public:
TArray<struct FCurveEdTab>                    Tabs;                                   // 0x0060 (0x0010)
```

```
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                              ActiveTab;                              // 0x0070 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpCurveEdSetup");
}

return uClassPointer;
};

};

// Class Engine.InterpTrack
// 0x0064 (0x0060 - 0x00C4)
class UInterpTrack : public UObject
{
public:
struct FPointer                          VfTable_FInterpEdInputInterface;             // 0x0060 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                          CurveEdVTable;                          // 0x0068 (0x0008)
[0x0000000000801000] (CPF_Native | CPF_NoExport)
TArray<class UInterpTrack*>                    SubTracks;                         // 0x0070 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FSubTrackGroup>                    SubTrackGroups;                         // 0x0080
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<struct FSupportedSubTrackInfo>             SupportedSubTracks;                         // 0x0090
(0x0010) [0x0000000800402000] (CPF_Transient | CPF_NeedCtorLink)
class UClass*                            TrackInstClass;                          // 0x00A0 (0x0008)
[0x0000000000000000]
uint8_t                              ActiveCondition;                          // 0x00A8 (0x0001)
[0x0000000000000001] (CPF_Edit)
class FString                           TrackTitle;                          // 0x00B0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                            bOnePerGroup : 1;                          // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                            bDirGroupOnly : 1;                          // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                            bDisableTrack : 1;                          // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                            bIsAnimControlTrack : 1;                          // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                            bSubTrackOnly : 1;                          // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                            bVisible : 1;                          // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000020] (CPF_Transient)
unsigned long                            bIsSelected : 1;                          // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000040] (CPF_Transient)
```

```
unsigned long                          bIsRecording : 1;                          // 0x00C0 (0x0004)
[0x0000000000002000] [0x00000080] (CPF_Transient)
unsigned long                          bIsCollapsed : 1;                          // 0x00C0 (0x0004)
[0x0000000800000000] [0x00000100]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrack");
}

return uClassPointer;
};

};

// Class Engine.JsonObject
// 0x00C4 (0x0060 - 0x0124)
class UJsonObject : public UObject
{
public:
struct FMap_Mirror                          ValueMap;                              // 0x0060 (0x0050)
[0x0000000000001000] (CPF_Native)
struct FMap_Mirror                          ObjectMap;                             // 0x00B0 (0x0050)
[0x0000000000001000] (CPF_Native)
TArray<class FString>                        ValueArray;                           // 0x0100 (0x0010)
[0x0000000000001000] (CPF_Native)
TArray<class UJsonObject*>                        ObjectArray;                         // 0x0110 (0x0010)
[0x0000000000001000] (CPF_Native)
unsigned long                          bArray : 1;                          // 0x0120 (0x0004)
[0x0000000000001000] [0x00000001] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.JsonObject");
}

return uClassPointer;
};

static class UJsonObject* DecodeJson(class FString Str);
static class FString EncodeJson(class UJsonObject* Root);
void SetBoolValue(class FString Key, unsigned long Value);
void SetFloatValue(class FString Key, float Value);
void SetIntValue(class FString Key, int32_t Value);
void SetStringValue(class FString Key, class FString Value);
void SetObject(class FString Key, class UJsonObject* Object);
bool GetBoolValue(class FString Key);
float GetFloatValue(class FString Key);
int32_t GetIntValue(class FString Key);
bool HasKey(class FString Key);
class FString GetStringValue(class FString Key);
class UJsonObject* GetObjectW(class FString Key);
};

// Class Engine.KMeshProps
// 0x0060 (0x0060 - 0x00C0)
class UKMeshProps : public UObject
{
public:
struct FVector                          COMNudge;                         // 0x0060 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FKAggregateGeom                  AggGeom;                          // 0x0070 (0x0050)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.KMeshProps");
}

return uClassPointer;
};

};

// Class Engine.LevelBase
// 0x0078 (0x0060 - 0x00D8)
class ULevelBase : public UObject
{
public:
uint8_t                                 UnknownData00[0x78];              // 0x0060 (0x0078)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelBase");
}

return uClassPointer;
};

};

// Class Engine.Level
// 0x0428 (0x00D8 - 0x0500)
class ULevel : public ULevelBase
{
public:
uint8_t                              UnknownData00[0x120];                    // 0x00D8 (0x0120)
MISSED OFFSET
float                                LightmapTotalSize;                       // 0x01F8 (0x0004)
[0x0000000000020002] (CPF_Const | CPF_EditConst)
float                                ShadowmapTotalSize;                      // 0x01FC (0x0004)
[0x0000000000020002] (CPF_Const | CPF_EditConst)
uint8_t                              UnknownData01[0x300];                    // 0x0200 (0x0300)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Level");
}

return uClassPointer;
};

};

// Class Engine.PendingLevel
// 0x0050 (0x00D8 - 0x0128)
class UPendingLevel : public ULevelBase
{
public:
uint8_t                              UnknownData00[0x50];                     // 0x00D8 (0x0050)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PendingLevel");
}

return uClassPointer;
};
```

```
};

// Class Engine.DemoPlayPendingLevel
// 0x0000 (0x0128 - 0x0128)
class UDemoPlayPendingLevel : public UPendingLevel
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DemoPlayPendingLevel");
}

return uClassPointer;
};

};

// Class Engine.NetPendingLevel
// 0x0000 (0x0128 - 0x0128)
class UNetPendingLevel : public UPendingLevel
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NetPendingLevel");
}

return uClassPointer;
};

};

// Class Engine.LevelStreaming
// 0x00B4 (0x0060 - 0x0114)
class ULevelStreaming : public UObject
{
public:
struct FName                    PackageName;                    // 0x0060 (0x0008)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
class ULevel*                   LoadedLevel;                    // 0x0068 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FVector                  Offset;                         // 0x0070 (0x000C)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FMatrix                          LevelTransform;                         // 0x0080 (0x0040)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FVector                          OldOffset;                              // 0x00C0 (0x000C)
[0x0000000000000002] (CPF_Const)
unsigned long                           bIsVisible : 1;                         // 0x00CC (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                           bHasLoadRequestPending : 1;             // 0x00CC (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                           bHasUnloadRequestPending : 1;           // 0x00CC
(0x0004) [0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                           bShouldBeVisibleInEditor : 1;           // 0x00CC (0x0004)
[0x0000000800000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                           bBoundingBoxVisible : 1;                // 0x00CC (0x0004)
[0x0000000000000002] [0x00000010] (CPF_Const)
unsigned long                           bLocked : 1;                            // 0x00CC (0x0004)
[0x0000000000000003] [0x00000020] (CPF_Edit | CPF_Const)
unsigned long                           bIsFullyStatic : 1;                     // 0x00CC (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                           bShouldBeLoaded : 1;                    // 0x00CC (0x0004)
[0x0000000000002002] [0x00000080] (CPF_Const | CPF_Transient)
unsigned long                           bShouldBeVisible : 1;                   // 0x00CC (0x0004)
[0x0000000000002002] [0x00000100] (CPF_Const | CPF_Transient)
unsigned long                           bShouldBlockOnLoad : 1;                 // 0x00CC (0x0004)
[0x0000000000002000] [0x00000200] (CPF_Transient)
unsigned long                           bDrawOnLevelStatusMap : 1;              // 0x00CC (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                           bIsRequestingUnloadAndRemoval : 1;      // 0x00CC
(0x0004) [0x0000000000002002] [0x00000800] (CPF_Const | CPF_Transient)
struct FColor                           DrawColor;                              // 0x00D0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
TArray<class ALevelStreamingVolume*>            EditorStreamingVolumes;                 //
0x00D8 (0x0010) [0x0000000000420003] (CPF_Edit | CPF_Const | CPF_EditConst |
CPF_NeedCtorLink)
float                                   MinTimeBetweenVolumeUnloadRequests;     // 0x00E8
(0x0004) [0x0000000000000001] (CPF_Edit)
float                                   LastVolumeUnloadRequestTime;            // 0x00EC (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<class FString>                   Keywords;                               // 0x00F0 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
class ALevelGridVolume*                 EditorGridVolume;                       // 0x0100 (0x0008)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                                 GridPosition[0x3];                      // 0x0108 (0x000C)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelStreaming");
}
return uClassPointer;
};

};

// Class Engine.LevelStreamingAlwaysLoaded
// 0x0008 (0x0114 - 0x011C)
class ULevelStreamingAlwaysLoaded : public ULevelStreaming
{
public:
unsigned long                          bIsProceduralBuildingLODLevel : 1;          // 0x0118 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelStreamingAlwaysLoaded");
}

return uClassPointer;
};

};

// Class Engine.LevelStreamingDistance
// 0x0014 (0x0114 - 0x0128)
class ULevelStreamingDistance : public ULevelStreaming
{
public:
struct FVector                         Origin;                               // 0x0118 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                            MaxDistance;                          // 0x0124 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelStreamingDistance");
}

return uClassPointer;
};

};
```

```
// Class Engine.LevelStreamingKismet
// 0x0004 (0x0114 - 0x0118)
class ULevelStreamingKismet : public ULevelStreaming
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelStreamingKismet");
}

return uClassPointer;
};

};

// Class Engine.LevelStreamingPersistent
// 0x0004 (0x0114 - 0x0118)
class ULevelStreamingPersistent : public ULevelStreaming
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LevelStreamingPersistent");
}

return uClassPointer;
};

};

// Class Engine.LightmappedSurfaceCollection
// 0x0018 (0x0060 - 0x0078)
class ULightmappedSurfaceCollection : public UObject
{
public:
class UModel*                    SourceModel;                    // 0x0060 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<int32_t>                  Surfaces;                     // 0x0068 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightmappedSurfaceCollection");
}

return uClassPointer;
};

};

// Class Engine.LightmassLevelSettings
// 0x0030 (0x0060 - 0x0090)
class ULightmassLevelSettings : public UObject
{
public:
int32_t                                    NumIndirectLightingBounces;                 // 0x0060 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                              EnvironmentColor;                           // 0x0064 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      EnvironmentIntensity;                       // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      EmissiveBoost;                              // 0x006C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      DiffuseBoost;                               // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      SpecularBoost;                              // 0x0074 (0x0004)
[0x0000000000000000]
unsigned long                              bUseAmbientOcclusion : 1;                   // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                              bVisualizeAmbientOcclusion : 1;             // 0x0078 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                                      DirectIlluminationOcclusionFraction;        // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      IndirectIlluminationOcclusionFraction;      // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      OcclusionExponent;                          // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      FullyOccludedSamplesFraction;               // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      MaxOcclusionDistance;                       // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightmassLevelSettings");
}
return uClassPointer;
};

};

// Class Engine.LightmassPrimitiveSettingsObject
// 0x001C (0x0060 - 0x007C)
class ULightmassPrimitiveSettingsObject : public UObject
{
public:
struct FLightmassPrimitiveSettings          LightmassSettings;                    // 0x0060
(0x001C) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightmassPrimitiveSettingsObject");
}

return uClassPointer;
};

};

// Class Engine.LinkedAccountDetails
// 0x0000 (0x0060 - 0x0060)
class ULinkedAccountDetails : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LinkedAccountDetails");
}

return uClassPointer;
};

};

// Class Engine.MapInfo
// 0x0000 (0x0060 - 0x0060)
class UMapInfo : public UObject
{
```

```
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MapInfo");
}

return uClassPointer;
};

};

// Class Engine.Model
// 0x0D50 (0x0060 - 0x0DB0)
class UModel : public UObject
{
public:
uint8_t                                UnknownData00[0xD50];                  // 0x0060 (0x0D50)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Model");
}

return uClassPointer;
};

};

// Class Engine.MusicTrackDataStructures
// 0x0000 (0x0060 - 0x0060)
class UMusicTrackDataStructures : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MusicTrackDataStructures");
}

return uClassPointer;
};

};

// Class Engine.NavigationMeshBase
// 0x0340 (0x0060 - 0x03A0)
class UNavigationMeshBase : public UObject
{
public:
uint8_t                        UnknownData00[0x340];                // 0x0060 (0x0340)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavigationMeshBase");
}

return uClassPointer;
};

};

// Class Engine.NetDriver
// 0x01A0 (0x0068 - 0x0208)
class UNetDriver : public USubsystem
{
public:
uint8_t                        UnknownData00[0x3C];                 // 0x0068 (0x003C)
MISSED OFFSET
float                          ConnectionTimeout;                   // 0x00A4 (0x0004)
[0x0000000000004000] (CPF_Config)
float                          KeepAliveTime;                       // 0x00A8 (0x0004)
[0x0000000000004000] (CPF_Config)
float                          RelevantTimeout;                     // 0x00AC (0x0004)
[0x0000000000004000] (CPF_Config)
float                          SpawnPrioritySeconds;                // 0x00B0 (0x0004)
[0x0000000000004000] (CPF_Config)
float                          ServerTravelPause;                   // 0x00B4 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                        MinClientRate;                       // 0x00B8 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                        MaxClientRate;                       // 0x00BC (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                        MedianClientRate;                    // 0x00C0 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                        MinReplicationRate;                  // 0x00C4 (0x0004)
```

```
[0x0000000000004000] (CPF_Config)
int32_t                                  MaxReplicationRate;                        // 0x00C8 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                  MedianReplicationRate;                    // 0x00CC (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                  NetServerMaxTickRate;                     // 0x00D0 (0x0004)
[0x0000000000004000] (CPF_Config)
unsigned long                            bClampListenServerTickRate : 1;           // 0x00D4 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
unsigned long                            AllowDownloads : 1;                       // 0x00D8 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
unsigned long                            AllowPeerConnections : 1;                 // 0x00DC (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
unsigned long                            AllowPeerVoice : 1;                       // 0x00E0 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
uint8_t                                  UnknownData01[0x24];                      // 0x00E4 (0x0024)
MISSED OFFSET
int32_t                                  MaxDownloadSize;                          // 0x0108 (0x0004)
[0x0000000000004000] (CPF_Config)
TArray<class FString>                    DownloadManagers;                         // 0x0110 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
uint8_t                                  UnknownData02[0xA8];                      // 0x0120 (0x00A8)
MISSED OFFSET
class FString                            NetConnectionClassName;                   // 0x01C8 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
uint8_t                                  UnknownData03[0x30];                      // 0x01D8 (0x0030)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NetDriver");
}

return uClassPointer;
};

};

// Class Engine.DemoRecDriver
// 0x00F8 (0x0208 - 0x0300)
class UDemoRecDriver : public UNetDriver
{
public:
uint8_t                                  UnknownData00[0x30];                      // 0x0208 (0x0030)
MISSED OFFSET
class FString                            DemoSpectatorClass;                       // 0x0238 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
uint8_t                                  UnknownData01[0xA0];                      // 0x0248 (0x00A0)
MISSED OFFSET
```

```
int32_t                                    MaxRewindPoints;                            // 0x02E8 (0x0004)
[0x0000000000004000] (CPF_Config)
float                                     RewindPointInterval;                        // 0x02F0 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                    NumRecentRewindPoints;                      // 0x02F4 (0x0004)
[0x0000000000004000] (CPF_Config)
uint8_t                                   UnknownData02[0x8];                         // 0x02F8 (0x0008) MISSED
OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DemoRecDriver");
}

return uClassPointer;
};

};

// Class Engine.NetworkEncryptionKey
// 0x0040 (0x0060 - 0x00A0)
class UNetworkEncryptionKey : public UObject
{
public:
TArray<uint8_t>                            Key;                                        // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                            IV;                                         // 0x0070 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                            HMACKey;                                    // 0x0080 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                            SessionId;                                  // 0x0090 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NetworkEncryptionKey");
}

return uClassPointer;
};

static class UNetworkEncryptionKey* Generate();
};
```

```
// Class Engine.ObjectReferencer
// 0x0010 (0x0060 - 0x0070)
class UObjectReferencer : public UObject
{
public:
TArray<class UObject*>                    ReferencedObjects;                    // 0x0060 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ObjectReferencer");
}

return uClassPointer;
};

};

// Class Engine.OnlineAuthInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineAuthInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineAuthInterface");
}

return uClassPointer;
};

class UAsyncTask* RequestAccountAuthorization(struct FUniqueNetId PlayerID, struct
FScriptDelegate Callback);
void OnAccountAuthorization(class FString Token);
void OnLoginChanged(unsigned long bLoggedIn);
bool RequestMtxCode(struct FUniqueNetId PlayerID, struct FScriptDelegate Callback);
bool RequestAuthTicket(struct FUniqueNetId PlayerID, struct FScriptDelegate Callback);
bool RequiresAuthTicket();
void OnReceivedAuthCode(unsigned long bSuccess, class FString Code);
bool GetServerAddr(struct FIpAddr& OutServerIP, int32_t& OutServerPort);
bool GetServerUniqueId(struct FUniqueNetId& OutServerUID);
bool FindLocalServerAuthSession(class UPlayer* ClientConnection, struct FLocalAuthSession&
OutSessionInfo);
```

```
bool FindServerAuthSession(class UPlayer* ServerConnection, struct FAuthSession&
OutSessionInfo);
bool FindLocalClientAuthSession(class UPlayer* ServerConnection, struct FLocalAuthSession&
OutSessionInfo);
bool FindClientAuthSession(class UPlayer* ClientConnection, struct FAuthSession&
OutSessionInfo);
void AllLocalServerAuthSessions(struct FLocalAuthSession& OutSessionInfo);
void AllServerAuthSessions(struct FAuthSession& OutSessionInfo);
void AllLocalClientAuthSessions(struct FLocalAuthSession& OutSessionInfo);
void AllClientAuthSessions(struct FAuthSession& OutSessionInfo);
void EndAllRemoteServerAuthSessions();
void EndAllLocalServerAuthSessions();
void EndRemoteServerAuthSession(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP);
void EndLocalServerAuthSession(struct FUniqueNetId ClientUID, struct FIpAddr ClientIP);
bool VerifyServerAuthSession(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP, int32_t
AuthTicketUID);
bool CreateServerAuthSession(struct FUniqueNetId ClientUID, struct FIpAddr ClientIP, int32_t
ClientPort, int32_t& OutAuthTicketUID);
void EndAllRemoteClientAuthSessions();
void EndAllLocalClientAuthSessions();
void EndRemoteClientAuthSession(struct FUniqueNetId ClientUID, struct FIpAddr ClientIP);
void EndLocalClientAuthSession(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP, int32_t
ServerPort);
bool VerifyClientAuthSession(struct FUniqueNetId ClientUID, struct FIpAddr ClientIP, int32_t
ClientPort, int32_t AuthTicketUID);
bool CreateClientAuthSession(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP, int32_t
ServerPort, unsigned long bSecure, int32_t& OutAuthTicketUID);
bool SendServerAuthRetryRequest();
bool SendClientAuthEndSessionRequest(class UPlayer* ClientConnection);
bool SendServerAuthResponse(class UPlayer* ClientConnection, int32_t AuthTicketUID);
bool SendClientAuthResponse(int32_t AuthTicketUID);
bool SendServerAuthRequest(struct FUniqueNetId ServerUID);
bool SendClientAuthRequest(class UPlayer* ClientConnection, struct FUniqueNetId ClientUID);
void ClearServerConnectionCloseDelegate(struct FScriptDelegate
ServerConnectionCloseDelegate);
void AddServerConnectionCloseDelegate(struct FScriptDelegate
ServerConnectionCloseDelegate);
void OnServerConnectionClose(class UPlayer* ServerConnection);
void ClearClientConnectionCloseDelegate(struct FScriptDelegate
ClientConnectionCloseDelegate);
void AddClientConnectionCloseDelegate(struct FScriptDelegate
ClientConnectionCloseDelegate);
void OnClientConnectionClose(class UPlayer* ClientConnection);
void ClearServerAuthRetryRequestDelegate(struct FScriptDelegate
ServerAuthRetryRequestDelegate);
void AddServerAuthRetryRequestDelegate(struct FScriptDelegate
ServerAuthRetryRequestDelegate);
void OnServerAuthRetryRequest(class UPlayer* ClientConnection);
void ClearClientAuthEndSessionRequestDelegate(struct FScriptDelegate
ClientAuthEndSessionRequestDelegate);
void AddClientAuthEndSessionRequestDelegate(struct FScriptDelegate
ClientAuthEndSessionRequestDelegate);
void OnClientAuthEndSessionRequest(class UPlayer* ServerConnection);
void ClearServerAuthCompleteDelegate(struct FScriptDelegate ServerAuthCompleteDelegate);
void AddServerAuthCompleteDelegate(struct FScriptDelegate ServerAuthCompleteDelegate);
void OnServerAuthComplete(unsigned long bSuccess, struct FUniqueNetId ServerUID, class
UPlayer* ServerConnection, class FString ExtraInfo);
void ClearClientAuthCompleteDelegate(struct FScriptDelegate ClientAuthCompleteDelegate);
void AddClientAuthCompleteDelegate(struct FScriptDelegate ClientAuthCompleteDelegate);
void OnClientAuthComplete(unsigned long bSuccess, struct FUniqueNetId ClientUID, class
UPlayer* ClientConnection, class FString ExtraInfo);
void ClearServerAuthResponseDelegate(struct FScriptDelegate ServerAuthResponseDelegate);
void AddServerAuthResponseDelegate(struct FScriptDelegate ServerAuthResponseDelegate);
void OnServerAuthResponse(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP, int32_t
AuthTicketUID);
void ClearClientAuthResponseDelegate(struct FScriptDelegate ClientAuthResponseDelegate);
void AddClientAuthResponseDelegate(struct FScriptDelegate ClientAuthResponseDelegate);
void OnClientAuthResponse(struct FUniqueNetId ClientUID, struct FIpAddr ClientIP, int32_t
AuthTicketUID);
void ClearServerAuthRequestDelegate(struct FScriptDelegate ServerAuthRequestDelegate);
void AddServerAuthRequestDelegate(struct FScriptDelegate ServerAuthRequestDelegate);
void OnServerAuthRequest(class UPlayer* ClientConnection, struct FUniqueNetId ClientUID,
struct FIpAddr ClientIP, int32_t ClientPort);
void ClearClientAuthRequestDelegate(struct FScriptDelegate ClientAuthRequestDelegate);
void AddClientAuthRequestDelegate(struct FScriptDelegate ClientAuthRequestDelegate);
void OnClientAuthRequest(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP, int32_t
ServerPort, unsigned long bSecure);
void ClearAuthReadyDelegate(struct FScriptDelegate AuthReadyDelegate);
void AddAuthReadyDelegate(struct FScriptDelegate AuthReadyDelegate);
void OnAuthReady();
bool IsReady();
};

// Class Engine.OnlineEventTracker
// 0x0000 (0x0060 - 0x0060)
class UOnlineEventTracker : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineEventTracker");
}

return uClassPointer;
};

bool eventRaiseEvent(class FString EventName, TArray<class FString> EventParams);
void eventShutDown();
void eventInit();
};

// Class Engine.OnlineFriendsInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineFriendsInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineFriendsInterface");
}

return uClassPointer;
};

bool GetActivePlatformId(uint8_t LocalUserNum, struct FUniqueNetId AccountId, struct
FUniqueNetId& PlatformId);
bool RequestLinkedAccounts(uint8_t LocalUserNum, TArray<struct FUniqueNetId> AccountIds,
struct FScriptDelegate Callback);
void OnReceivedLinkedAccount(unsigned long bSuccess, TArray<struct FLinkedAccountData>
LinkedAccountData);
};

// Class Engine.OnlineLobbySettings
// 0x0000 (0x0060 - 0x0060)
class UOnlineLobbySettings : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineLobbySettings");
}

return uClassPointer;
};

};

// Class Engine.OnlineMatchmakingStats
// 0x0000 (0x0060 - 0x0060)
class UOnlineMatchmakingStats : public UObject
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineMatchmakingStats");
}

return uClassPointer;
};

void StopTimer(struct FMMStats_Timer& Timer);
void StartTimer(struct FMMStats_Timer& Timer);
};

// Class Engine.OnlinePersistentAuthInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlinePersistentAuthInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlinePersistentAuthInterface");
}

return uClassPointer;
};

bool AuthWithNintendoAccountToken(uint8_t LocalUserNum, class FString&
NintendoAccountToken);
uint64_t GetTimeUntilAuthExpiration(uint8_t LocalUserNum);
class UDateTime* GetAuthExpirationTimestamp(uint8_t LocalUserNum);
bool LaunchAccountPortal(uint8_t LocalUserNum);
class FString GetClientSecret();
class FString GetClientID();
class FString GetClientCredentials();
class FString GetContinuanceToken(uint8_t LocalUserNum);
bool UseRefreshToken(uint8_t LocalUserNum, class FString RefreshToken);
class FString GetRefreshToken(uint8_t LocalUserNum);
void ClearUnderageUserDetectedDelegate(struct FScriptDelegate Callback);
void AddUnderageUserDetectedDelegate(struct FScriptDelegate Callback);
void OnUnderageUserDetected(uint8_t LocalUserNum, class FString ParentalConsentURL);
bool RequestPinGrantCode(uint8_t LocalUserNum);
void ClearRequestPinGrantCodeDelegate(uint8_t LocalUserNum, struct FScriptDelegate
Callback);
void AddRequestPinGrantCodeDelegate(uint8_t LocalUserNum, struct FScriptDelegate Callback);
void OnReceievedPinGrantCode(uint8_t Result, uint8_t LocalUserNum, class FString Code, class
FString URL, int32_t SecondsUntilExpiration);
};

// Class Engine.OnlinePlayerStorage
// 0x0038 (0x0060 - 0x0098)
class UOnlinePlayerStorage : public UObject
{
public:
int32_t                         VersionNumber;                      // 0x0060 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         VersionSettingsId;                  // 0x0064 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         SaveCountSettingId;                  // 0x0068 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<struct FOnlineProfileSetting>           ProfileSettings;                        // 0x0070 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FSettingsPropertyPropertyMetaData>   ProfileMappings;                          //
0x0080 (0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                         AsyncState;                         // 0x0090 (0x0001)
[0x0000000000000002] (CPF_Const)
int32_t                         DeviceID;                          // 0x0094 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlinePlayerStorage");
}

return uClassPointer;
};

void SetDefaultVersionNumber();
int32_t GetVersionNumber();
void AppendVersionToSettings();
void eventSetToDefaults();
void AddSettingFloat(int32_t SettingId);
void AddSettingInt(int32_t SettingId);
bool GetRangedProfileSettingValue(int32_t ProfileId, float& OutValue);
bool SetRangedProfileSettingValue(int32_t ProfileId, float NewValue);
bool GetProfileSettingRange(int32_t ProfileId, float& OutMinValue, float& OutMaxValue, float&
RangeIncrement, uint8_t& bFormatAsInt);
static bool GetProfileSettingMappingIds(int32_t ProfileId, TArray<int32_t>& Ids);
bool GetProfileSettingMappingType(int32_t ProfileId, uint8_t& OutType);
bool SetProfileSettingValueFloat(int32_t ProfileSettingId, float Value);
bool SetProfileSettingValueInt(int32_t ProfileSettingId, int32_t Value);
bool SetProfileSettingValueId(int32_t ProfileSettingId, int32_t Value);
bool GetProfileSettingValueFloat(int32_t ProfileSettingId, float& Value);
bool GetProfileSettingValueInt(int32_t ProfileSettingId, int32_t& Value);
bool GetProfileSettingValueFromListIndex(int32_t ProfileSettingId, int32_t ListIndex, int32_t&
Value);
bool GetProfileSettingValueId(int32_t ProfileSettingId, int32_t& ValueId, int32_t& ListIndex);
bool SetProfileSettingValue(int32_t ProfileSettingId, class FString& NewValue);
bool SetProfileSettingValueByName(struct FName ProfileSettingName, class FString&
NewValue);
bool GetProfileSettingValueByName(struct FName ProfileSettingName, class FString& Value);
bool GetProfileSettingValues(int32_t ProfileSettingId, TArray<struct FName>& Values);
struct FName GetProfileSettingValueName(int32_t ProfileSettingId);
bool GetProfileSettingValue(int32_t ProfileSettingId, int32_t ValueMapID, class FString& Value);
bool IsProfileSettingIdMapped(int32_t ProfileSettingId);
static int32_t FindDefaultProfileMappingIndexByName(struct FName ProfileSettingName);
int32_t FindProfileMappingIndexByName(struct FName ProfileSettingName);
int32_t FindProfileMappingIndex(int32_t ProfileSettingId);
int32_t FindProfileSettingIndex(int32_t ProfileSettingId);
class FString GetProfileSettingColumnHeader(int32_t ProfileSettingId);
struct FName GetProfileSettingName(int32_t ProfileSettingId);
bool GetProfileSettingId(struct FName ProfileSettingName, int32_t& ProfileSettingId);
};

// Class Engine.OnlineProfileSettings
// 0x0030 (0x0098 - 0x00C8)
class UOnlineProfileSettings : public UOnlinePlayerStorage
{
public:
TArray<int32_t>                          ProfileSettingIds;                       // 0x0098 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FOnlineProfileSetting>          DefaultSettings;                         // 0x00A8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FIdToStringMapping>             OwnerMappings;                           // 0x00B8
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineProfileSettings");
}

return uClassPointer;
};

void eventModifyAvailableProfileSettings();
void AppendVersionToReadIds();
void eventSetToDefaults();
bool GetProfileSettingDefaultFloat(int32_t ProfileSettingId, float& DefaultFloat);
bool GetProfileSettingDefaultInt(int32_t ProfileSettingId, int32_t& DefaultInt);
bool GetProfileSettingDefaultId(int32_t ProfileSettingId, int32_t& DefaultId, int32_t& ListIndex);
};

// Class Engine.OnlineStats
// 0x0010 (0x0060 - 0x0070)
```

```
class UOnlineStats : public UObject
{
public:
TArray<struct FStringIdToStringMapping>          ViewIdMappings;                              // 0x0060
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineStats");
}

return uClassPointer;
};

struct FName GetViewName(int32_t ViewId);
bool GetViewId(struct FName ViewName, int32_t& ViewId);
};

// Class Engine.OnlineStatsRead
// 0x0068 (0x0070 - 0x00D8)
class UOnlineStatsRead : public UOnlineStats
{
public:
int32_t                                  ViewId;                                  // 0x0070 (0x0004)
[0x0000000000000000]
int32_t                                  SortColumnId;                            // 0x0074 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<int32_t>                              ColumnIds;                               // 0x0078 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
int32_t                                  TotalRowsInView;                         // 0x0088 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<struct FOnlineStatsRow>                 Rows;                                    // 0x0090 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FColumnMetaData>                 ColumnMappings;                          // 0x00A0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class FString                                ViewName;                                // 0x00B0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
int32_t                                  TitleId;                                 // 0x00C0 (0x0004)
[0x0000000000000002] (CPF_Const)
class FString                                LeaderboardName;                         // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.OnlineStatsRead");
}

return uClassPointer;
};

int32_t GetRankForPlayer(struct FUniqueNetId PlayerID);
void AddPlayer(class FString PlayerName, struct FUniqueNetId PlayerID);
bool IsStatZero(struct FUniqueNetId PlayerID, int32_t StatColumnNo);
bool GetStatValueForPlayerAsString(struct FUniqueNetId PlayerID, int32_t StatColumnNo, class
FString& StatValue);
bool SetFloatStatValueForPlayer(struct FUniqueNetId PlayerID, int32_t StatColumnNo, float
StatValue);
bool GetFloatStatValueForPlayer(struct FUniqueNetId PlayerID, int32_t StatColumnNo, float&
StatValue);
bool SetIntStatValueForPlayer(struct FUniqueNetId PlayerID, int32_t StatColumnNo, int32_t
StatValue);
bool GetIntStatValueForPlayer(struct FUniqueNetId PlayerID, int32_t StatColumnNo, int32_t&
StatValue);
void eventOnReadComplete();
};

// Class Engine.OnlineStatsWrite
// 0x0060 (0x0070 - 0x00D0)
class UOnlineStatsWrite : public UOnlineStats
{
public:
TArray<struct FStringIdToStringMapping>          StatMappings;                          // 0x0070
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FSettingsProperty>               Properties;                          // 0x0080 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<int32_t>                              ViewIds;                              // 0x0090 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<int32_t>                              ArbitratedViewIds;                     // 0x00A0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                    RatingId;                             // 0x00B0 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FScriptDelegate                       __OnStatsWriteComplete__Delegate;           // 0x00B8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineStatsWrite");
}

return uClassPointer;
};

void DecrementIntStat(int32_t StatId, int32_t DecBy);
void DecrementFloatStat(int32_t StatId, float DecBy);
void IncrementIntStat(int32_t StatId, int32_t IncBy);
void IncrementFloatStat(int32_t StatId, float IncBy);
void SetIntStat(int32_t StatId, int32_t Value);
void SetFloatStat(int32_t StatId, float Value);
struct FName GetStatName(int32_t StatId);
bool GetStatId(struct FName StatName, int32_t& StatId);
void OnStatsWriteComplete();
};

// Class Engine.OnlineSubsystem
// 0x02F8 (0x0060 - 0x0358)
class UOnlineSubsystem : public UObject
{
public:
struct FPointer                          VfTable_FTickableObject;                 // 0x0060 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UOnlineAccountInterface*               AccountInterface_Object;                 // 0x0068
(0x0008) [0x0000000000000000]
class UOnlineAccountInterface*               AccountInterface_Interface;               // 0x0070
(0x0008) [0x0000000000000000]
class UOnlinePlayerInterface*               PlayerInterface_Object;                  // 0x0078
(0x0008) [0x0000000000000000]
class UOnlinePlayerInterface*               PlayerInterface_Interface;                // 0x0080
(0x0008) [0x0000000000000000]
class UOnlinePlayerInterfaceEx*              PlayerInterfaceEx_Object;                // 0x0088
(0x0008) [0x0000000000000000]
class UOnlinePlayerInterfaceEx*              PlayerInterfaceEx_Interface;              // 0x0090
(0x0008) [0x0000000000000000]
class UOnlineMarketplaceInterface*             MarketplaceInterface_Object;              // 0x0098
(0x0008) [0x0000000000000000]
class UOnlineMarketplaceInterface*             MarketplaceInterface_Interface;            // 0x00A0
(0x0008) [0x0000000000000000]
class UOnlineSystemInterface*               SystemInterface_Object;                  // 0x00A8
(0x0008) [0x0000000000000000]
class UOnlineSystemInterface*               SystemInterface_Interface;                // 0x00B0
(0x0008) [0x0000000000000000]
class UOnlineGameInterface*                GameInterface_Object;                   // 0x00B8
(0x0008) [0x0000000000000000]
class UOnlineGameInterface*                GameInterface_Interface;                 // 0x00C0
(0x0008) [0x0000000000000000]
class UOnlineContentInterface*               ContentInterface_Object;                 // 0x00C8
(0x0008) [0x0000000000000000]
class UOnlineContentInterface*               ContentInterface_Interface;               // 0x00D0
(0x0008) [0x0000000000000000]
class UOnlineVoiceInterface*               VoiceInterface_Object;                  // 0x00D8
(0x0008) [0x0000000000000000]
class UOnlineVoiceInterface*               VoiceInterface_Interface;                // 0x00E0
(0x0008) [0x0000000000000000]
class UOnlineStatsInterface*               StatsInterface_Object;                  // 0x00E8
(0x0008) [0x0000000000000000]
class UOnlineStatsInterface*               StatsInterface_Interface;                // 0x00F0
(0x0008) [0x0000000000000000]
class UOnlineNewsInterface*                NewsInterface_Object;                   // 0x00F8
```

```
(0x0008) [0x0000000000000000]
class UOnlineNewsInterface*                    NewsInterface_Interface;                    // 0x0100
(0x0008) [0x0000000000000000]
class UOnlinePartyChatInterface*                 PartyChatInterface_Object;                  // 0x0108
(0x0008) [0x0000000000000000]
class UOnlinePartyChatInterface*                 PartyChatInterface_Interface;               // 0x0110
(0x0008) [0x0000000000000000]
class UOnlineTitleFileInterface*                TitleFileInterface_Object;                  // 0x0118
(0x0008) [0x0000000000000000]
class UOnlineTitleFileInterface*                TitleFileInterface_Interface;               // 0x0120
(0x0008) [0x0000000000000000]
class UOnlineTitleFileCacheInterface*             TitleFileCacheInterface_Object;             // 0x0128
(0x0008) [0x0000000000000000]
class UOnlineTitleFileCacheInterface*             TitleFileCacheInterface_Interface;          // 0x0130
(0x0008) [0x0000000000000000]
class UUserCloudFileInterface*                  UserCloudInterface_Object;                  // 0x0138
(0x0008) [0x0000000000000000]
class UUserCloudFileInterface*                  UserCloudInterface_Interface;               // 0x0140
(0x0008) [0x0000000000000000]
class USharedCloudFileInterface*                 SharedCloudInterface_Object;                // 0x0148
(0x0008) [0x0000000000000000]
class USharedCloudFileInterface*                 SharedCloudInterface_Interface;             // 0x0150
(0x0008) [0x0000000000000000]
class UOnlineSocialInterface*                   SocialInterface_Object;                     // 0x0158
(0x0008) [0x0000000000000000]
class UOnlineSocialInterface*                   SocialInterface_Interface;                  // 0x0160
(0x0008) [0x0000000000000000]
class UOnlineAuthInterface*                     AuthInterface_Object;                       // 0x0168
(0x0008) [0x0000000000000000]
class UOnlineAuthInterface*                     AuthInterface_Interface;                    // 0x0170
(0x0008) [0x0000000000000000]
class UOnlinePersistentAuthInterface*             PersistentAuthInterface_Object;             // 0x0178
(0x0008) [0x0000000000000000]
class UOnlinePersistentAuthInterface*             PersistentAuthInterface_Interface;          //
0x0180 (0x0008) [0x0000000000000000]
class UOnlineGameDVRInterface*                  GameDVRInterface_Object;                    // 0x0188
(0x0008) [0x0000000000000000]
class UOnlineGameDVRInterface*                  GameDVRInterface_Interface;                 // 0x0190
(0x0008) [0x0000000000000000]
class UOnlineCommunityContentInterface*            CommunityContentInterface_Object;           //
0x0198 (0x0008) [0x0000000000000000]
class UOnlineCommunityContentInterface*            CommunityContentInterface_Interface;        //
0x01A0 (0x0008) [0x0000000000000000]
class UOnlinePurchaseInterface*                  PurchaseInterface_Object;                   // 0x01A8
(0x0008) [0x0000000000000000]
class UOnlinePurchaseInterface*                  PurchaseInterface_Interface;                // 0x01B0
(0x0008) [0x0000000000000000]
class UOnlineLobbyInterface*                    LobbyInterface_Object;                      // 0x01B8
(0x0008) [0x0000000000000000]
class UOnlineLobbyInterface*                    LobbyInterface_Interface;                   // 0x01C0
(0x0008) [0x0000000000000000]
class UOnlineFriendsInterface*                  FriendsInterface_Object;                    // 0x01C8
(0x0008) [0x0000000000000000]
class UOnlineFriendsInterface*                  FriendsInterface_Interface;                 // 0x01D0
```

```
(0x0008) [0x0000000000000000]
class UOnlineGameClipsInterface*            GameClipsInterface_Object;              // 0x01D8
(0x0008) [0x0000000000000000]
class UOnlineGameClipsInterface*            GameClipsInterface_Interface;           // 0x01E0
(0x0008) [0x0000000000000000]
class UClass*                         SearchClass;                        // 0x01E8 (0x0008)
[0x0000000000000000]
unsigned long                         bSupportsMultiSignin : 1;               // 0x01F0 (0x0004)
[0x0000004000004000] [0x00000001] (CPF_Config)
unsigned long                         bSupportsMultiVoice : 1;                // 0x01F0 (0x0004)
[0x0000004000004000] [0x00000002] (CPF_Config)
unsigned long                         bShowPrivilegeCheckErrors : 1;           // 0x01F0 (0x0004)
[0x0000004000004000] [0x00000004] (CPF_Config)
TArray<class UPlatformAccountSettings*>          AccountSettings;                        // 0x01F8
(0x0010) [0x0000000004480008] (CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)
TArray<class UPlatformBlockListStatus*>          BlockListStatuses;                      // 0x0208
(0x0010) [0x0000000004480008] (CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)
TArray<class UPlatformURL*>                 UnsupportedCorrectiveActionURLs;            // 0x0218
(0x0010) [0x0000000004480008] (CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)
TArray<struct FSteamDLCInfo>                 SteamDLC;                           // 0x0228 (0x0010)
[0x0000000000404001] (CPF_Edit | CPF_Config | CPF_NeedCtorLink)
TArray<struct FPS4DLCInfo>                  PS4DLC;                           // 0x0238 (0x0010)
[0x0000000000404001] (CPF_Edit | CPF_Config | CPF_NeedCtorLink)
TArray<struct FXboxOneDLCInfo>                XboxOneDLC;                           // 0x0248
(0x0010) [0x0000000000404001] (CPF_Edit | CPF_Config | CPF_NeedCtorLink)
TArray<struct FSwitchDLCInfo>                 SwitchDLC;                          // 0x0258 (0x0010)
[0x0000000000404001] (CPF_Edit | CPF_Config | CPF_NeedCtorLink)
TArray<struct FEpicDLCInfo>                 EpicDLC;                          // 0x0268 (0x0010)
[0x0000000000404001] (CPF_Edit | CPF_Config | CPF_NeedCtorLink)
uint8_t                             OnlinePlatformType;                  // 0x0278 (0x0001)
[0x0000004000004000] (CPF_Config)
uint8_t                             CurrentConnectionStatus;                // 0x0279 (0x0001)
[0x0000000000002000] (CPF_Transient)
TArray<class FString>                      OnlineSubsystemNames;                  // 0x0280
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<struct FNamedInterface>               NamedInterfaces;                      // 0x0290
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FNamedInterfaceDef>              NamedInterfaceDefs;                      // 0x02A0
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<struct FNamedSession>                Sessions;                           // 0x02B0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class FString                        IniLocPatcherClassName;                // 0x02C0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UIniLocPatcher*                     Patcher;                            // 0x02D0 (0x0008)
[0x0000000000002000] (CPF_Transient)
float                             AsyncMinCompletionTime;                 // 0x02D8 (0x0004)
[0x0000000000004000] (CPF_Config)
struct FScriptDelegate                     __FeaturePrivilegeLevelUpdated__Delegate;      // 0x02E0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                     __EventPlatformAccountSettingsCreated__Delegate;//
0x02F8 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
```

```
struct FScriptDelegate                    __EventPlatformBlockListStatusCreated__Delegate;//
0x0310 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                    __OnReadOnlineAvatarComplete__Delegate;        // 0x0328
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                    __OnSystemUserControllerPairingChanged__Delegate;//
0x0340 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineSubsystem");
}

return uClassPointer;
};

bool TriggerScreenshot();
static uint8_t OnlineFriendStateFromString(class FString State);
uint8_t GetControllerPlatform(int32_t LocalPlayerNum);
void SetControllerLayout(int32_t ControllerId, struct FName LayoutName);
void RemovePsyNetPartyMember(struct FUniqueNetId PlayerID);
void AddPsyNetPartyMember(struct FUniqueNetId PlayerID);
bool CanSendOfflinePartyInvite(struct FUniqueNetId PlayerID);
static bool UniqueNetIDIsValid(struct FUniqueNetId InID);
class FString CombineURLWithLoginInfo(class FString& InURL);
int32_t GetEnvironmentZone();
static class FString GetPlayerIP(struct FUniqueNetId PlayerID);
bool ShowRestrictionMessage(int32_t ControllerId, uint8_t Privilege);
bool IsOriginalAppOwner();
bool CanPlayersTextChat(struct FUniqueNetId PlayerID, struct FUniqueNetId TargetId);
bool ShowHelpUI(uint8_t LocalUserNum);
bool ResetStats(unsigned long bResetAchievements);
void FinishOnlineGameSession(class FString ServerId);
void StartOnlineGameSession(class FString ServerId);
void ClearSystemUserControllerPairingChangedDelegate(struct FScriptDelegate
PairingChangeDelegate);
void AddSystemUserControllerPairingChangedDelegate(struct FScriptDelegate
PairingChangeDelegate);
void OnSystemUserControllerPairingChanged(int32_t NewLocalUserNum, int32_t
PreviousLocalUserNum);
bool IsAchievementUnlocked(uint8_t LocalUserNum, int32_t AchievementId);
bool IsRichPresenceLocalized();
void UpdateGameProgress(uint8_t LocalPlayerNum, float Progress);
void ClearAccountPickerInput();
void LookForAccountPickerInput();
void AddPlayerToSession(uint8_t ControllerId);
void ClearPrimaryPlayer();
void RemoveUserAssociation(uint8_t ControllerId);
void MapEnd();
void MapStart();
void OnlineMatchEnd();
void OnlineMatchStart(class FString MapName);
void SetSessionDifficultyLevel(int32_t DifficultyLevel);
void SetSessionGameplayModeName(struct FName GameplayModeName);
void SetSessionGameplayMode(int32_t GameplayMode);
void PrintDebugInfo(class UDebugDrawer* Drawer);
static uint8_t GetOnlinePlatformFromName(class FString PlatformName);
static class FString GetNativePlatformName();
static class FString GetPlatformName(uint8_t PlatformType);
static class FString ReplacePlatformServiceName(class FString ReplString);
bool IsEnabled();
void SetPlayedWith(struct FUniqueNetId PlayerNetId);
void ClearOnlineAvatar(struct FUniqueNetId PlayerNetId);
void ClearPendingAvatarDownloads();
bool UseSubSystemAvatar();
void ReadOnlineAvatars(uint8_t Size, struct FScriptDelegate ReadOnlineAvatarCompleteDelegate,
TArray<struct FUniqueNetId>& PlayerNetIds);
void OnReadOnlineAvatarComplete(struct FUniqueNetId PlayerNetId, class UTexture* Avatar,
class FString OnlinePlayerName);
void SetDebugSpewLevel(int32_t DebugSpewLevel);
void DumpVoiceRegistration();
void DumpSessionState();
static void DumpGameSettings(class UOnlineGameSettings* GameSettings);
static int32_t GetNumSupportedLogins();
int32_t GetBuildUniqueId();
struct FUniqueNetId eventGetPlayerUniqueNetIdFromIndex(int32_t UserIndex);
static bool StringToUniqueLobbyId(class FString UniqueNetIdString, struct FUniqueLobbyId&
out_UniqueId);
static class FString UniqueLobbyIdToString(struct FUniqueLobbyId& IdToConvert);
static bool StringToUniqueNetId(class FString UniqueNetIdString, struct FUniqueNetId&
out_UniqueId);
static class FString UniqueNetIdToString(struct FUniqueNetId& IdToConvert);
class UObject* eventGetNamedInterface(struct FName InterfaceName);
void eventSetNamedInterface(struct FName InterfaceName, class UObject* NewInterface);
bool eventSetCommunityContentInterface(class UObject* InCommunityContentInterface);
bool eventSetGameDVRInterface(class UObject* InGameDVRInterface);
bool eventSetSharedCloudInterface(class UObject* InCloudInterface);
bool eventSetUserCloudInterface(class UObject* InCloudInterface);
bool eventSetFriendsInterface(class UObject* InFriendsInterface);
bool eventSetPersistentAuthInterface(class UObject* InPersistentAuthInterface);
bool eventSetAuthInterface(class UObject* InAuthInterface);
bool eventSetSocialInterface(class UObject* InSocialInterface);
bool eventSetTitleFileCacheInterface(class UObject* NewInterface);
bool eventSetTitleFileInterface(class UObject* NewInterface);
bool eventSetPartyChatInterface(class UObject* NewInterface);
bool eventSetNewsInterface(class UObject* NewInterface);
bool eventSetStatsInterface(class UObject* NewInterface);
bool eventSetGameClipsInterface(class UObject* InGameClipsInterface);
bool eventSetVoiceInterface(class UObject* NewInterface);
bool eventSetContentInterface(class UObject* NewInterface);
bool eventSetLobbyInterface(class UObject* InInterface);
bool eventSetPurchaseInterface(class UObject* NewInterface);
bool eventSetGameInterface(class UObject* NewInterface);
bool eventSetSystemInterface(class UObject* NewInterface);
bool eventSetMarketplaceInterface(class UObject* NewInterface);
bool eventSetPlayerInterfaceEx(class UObject* NewInterface);
bool eventSetPlayerInterface(class UObject* NewInterface);
bool eventSetAccountInterface(class UObject* NewInterface);
void eventExit();
bool eventPostInit();
bool eventInit();
class UPlatformURL* GetUnsupportedCorrectiveActionURLForUser(int32_t ControllerId);
class UPlatformBlockListStatus* GetBlockListStatusForUser(int32_t ControllerId);
class UPlatformBlockListStatus* GetBlockListStatusForPrimaryUser();
class UPlatformAccountSettings* GetAccountSettingsForUser(int32_t ControllerId);
class UPlatformAccountSettings* GetAccountSettingsForPrimaryUser();
void EventPlatformBlockListStatusCreated(class UPlatformBlockListStatus* Status, uint8_t
ControllerId);
void EventPlatformAccountSettingsCreated(class UPlatformAccountSettings* PlatformSettings,
uint8_t ControllerId);
void GetFeaturePrivilegeLevel(uint8_t LocalUserNum, uint8_t Privilege, struct FScriptDelegate
Callback);
void FeaturePrivilegeLevelUpdated(uint8_t LocalUserNum, uint8_t Privilege, uint8_t Level, class
UError* FailReason);
};

// Class Engine.ORS
// 0x0000 (0x0060 - 0x0060)
class UORS : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ORS");
}

return uClassPointer;
};

void ClearAllTimers(class UObject* Object);
static void RemoveObjectFromAny(class UObject* Object);
static void DestroyObject(class UObject* Object);
static class UScriptGroup_ORS* CreateGroup(class UObject* GroupOwner);
static class UScriptGroup_ORS* Group(class UObject* GroupMember);
static class UScriptGroup_ORS* GlobalGroup();
static void ShutDown();
static void Init();
static void DisableSubscription(class UClass* SystemClass, struct FName FunctionName);
static void DisableSystem(class UClass* SystemClass);
};

// Class Engine.PackageMapLevel
// 0x0008 (0x0118 - 0x0120)
class UPackageMapLevel : public UPackageMap
{
public:
uint8_t                         UnknownData00[0x8];                 // 0x0118 (0x0008) MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PackageMapLevel");
}

return uClassPointer;
};

};

// Class Engine.PackageMapSeekFree
// 0x0000 (0x0120 - 0x0120)
class UPackageMapSeekFree : public UPackageMapLevel
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PackageMapSeekFree");
}

return uClassPointer;
};

};

// Class Engine.PatchScriptCommandlet
// 0x000C (0x00B4 - 0x00C0)
class UPatchScriptCommandlet : public UCommandlet
{
public:
uint8_t                         UnknownData00[0xC];                 // 0x00B4 (0x000C) MISSED OFFSET

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PatchScriptCommandlet");
}

return uClassPointer;
};

};

// Class Engine.PitchTekSettings
// 0x0138 (0x0060 - 0x0198)
class UPitchTekSettings : public UObject
{
public:
unsigned long                    bEnabled : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bUseInitialColorTexture : 1;               // 0x0060 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bUseInitialDataTexture : 1;               // 0x0060 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    bRenderBallDecalOnContact : 1;              // 0x0060 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bRenderCarPositionDecal : 1;              // 0x0060 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
uint8_t                        ColorTargetSize;                    // 0x0064 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                        DataTargetSize;                    // 0x0065 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                         PitchLengthX;                    // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         PitchLengthY;                    // 0x006C (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                   InitialColorTexture;                 // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                   InitialDataTexture;                  // 0x0078 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FPitchTekClearSettings              ClearSettings;                    // 0x0080 (0x0010)
[0x0000000000000001] (CPF_Edit)
float                         BallContactHeight;                  // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPitchTekTextureDecalSettings            BallSettings;                     // 0x0098 (0x0020)
[0x0000000000000001] (CPF_Edit)
float                         WheelContactHeight;                  // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPitchTekTextureDecalSettings            WheelSettings;                     // 0x00C0
(0x0020) [0x0000000000000001] (CPF_Edit)
float                         CarBodyContactHeight;                 // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPitchTekTextureDecalSettings            CarBodySettings;                    // 0x00E8
(0x0020) [0x0000000000000001] (CPF_Edit)
float                         DemolitionExplosionMaxHeight;              // 0x0108 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
struct FPitchTekTextureDecalSettings            DemolitionSettings;                        // 0x0110
(0x0020) [0x0000000000000001] (CPF_Edit)
struct FPitchTekTextureDecalSettings            GoalExplosionSettings;                     // 0x0130
(0x0020) [0x0000000000000001] (CPF_Edit)
float                                   JumpBlastMaxHeight;                        // 0x0150 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPitchTekTextureDecalSettings            JumpBlastSettings;                         // 0x0158
(0x0020) [0x0000000000000001] (CPF_Edit)
struct FPitchTekTextureDecalSettings            CarPositionSettings;                       // 0x0178
(0x0020) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PitchTekSettings");
}

return uClassPointer;
};

};

// Class Engine.PlatformAccountSettings
// 0x0020 (0x0070 - 0x0090)
class UPlatformAccountSettings : public UComponent
{
public:
unsigned long                           bCrossplayEnabled : 1;                     // 0x0070 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                           bAllowInvites : 1;                         // 0x0070 (0x0004)
[0x0000000000000000] [0x00000002]
uint8_t                                 CrossPlatformChatState;                    // 0x0074 (0x0001)
[0x0000000000000000]
struct FScriptDelegate                  __EventAccountSettingsChanged__Delegate;      // 0x0078
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlatformAccountSettings");
}

return uClassPointer;
};
```

```
void EventAccountSettingsChanged(class UPlatformAccountSettings* AccountSettings);
};

// Class Engine.PlatformBlockListStatus
// 0x0020 (0x0070 - 0x0090)
class UPlatformBlockListStatus : public UComponent
{
public:
uint8_t                              DownloadStatus;                    // 0x0070 (0x0001)
[0x0008004000000000]
struct FScriptDelegate               __DownloadStatus__ChangeNotify;          // 0x0078
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlatformBlockListStatus");
}

return uClassPointer;
};

void __DownloadStatus__ChangeNotifyFunc();
void eventSetStatus(uint8_t InStatus);
};

// Class Engine.PlatformInterfaceBase
// 0x0028 (0x0060 - 0x0088)
class UPlatformInterfaceBase : public UObject
{
public:
TArray<struct FDelegateArray>        AllDelegates;                      // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate               __PlatformInterfaceDelegate__Delegate;     // 0x0070
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlatformInterfaceBase");
}

return uClassPointer;
};

void ClearDelegate(int32_t DelegateType, struct FScriptDelegate InDelegate);
void AddDelegate(int32_t DelegateType, struct FScriptDelegate InDelegate);
static class UInAppMessageBase* GetInAppMessageInterface();
static class UAppNotificationsBase* GetAppNotificationsInterface();
static class UTwitterIntegrationBase* GetTwitterIntegration();
static class UAnalyticEventsBase* GetAnalyticEventsInterface();
static class UMicroTransactionBase* GetMicroTransactionInterface();
static class UInGameAdManager* GetInGameAdManager();
static class UFacebookIntegration* GetFacebookIntegration();
static class UCloudStorageBase* GetLocalStorageInterface();
static class UCloudStorageBase* GetCloudStorageInterface();
void CallDelegates(int32_t DelegateType, struct FPlatformInterfaceDelegateResult&
DelegateResult);
void PlatformInterfaceDelegate(struct FPlatformInterfaceDelegateResult& Result);
};

// Class Engine.MicroTransactionBase
// 0x0030 (0x0088 - 0x00B8)
class UMicroTransactionBase : public UPlatformInterfaceBase
{
public:
TArray<struct FPurchaseInfo>                AvailableProducts;                    // 0x0088 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                           LastError;                        // 0x0098 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                           LastErrorSolution;                    // 0x00A8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MicroTransactionBase");
}

return uClassPointer;
};

int32_t eventGetProductIndex(class FString Identifier);
bool eventBeginPurchase(int32_t Index);
bool eventIsAllowedToMakePurchases();
bool eventQueryForAvailablePurchases();
void eventInit();
};

// Class Engine.MicroTransactionProxy
// 0x0000 (0x00B8 - 0x00B8)
class UMicroTransactionProxy : public UMicroTransactionBase
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MicroTransactionProxy");
}

return uClassPointer;
};

};

// Class Engine.PlatformURL
// 0x0028 (0x0070 - 0x0098)
class UPlatformURL : public UComponent
{
public:
class FString                    URL;                        // 0x0070 (0x0010)
[0x0008004000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                __URL__ChangeNotify;              // 0x0080 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlatformURL");
}

return uClassPointer;
};

void __URL__ChangeNotifyFunc();
void eventSetURL(class FString InURL);
};

// Class Engine.Player
// 0x0058 (0x0060 - 0x00B8)
class UPlayer : public UObject
{
public:
struct FPointer                  VfTable_FExec;                 // 0x0060 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UObjectProvider*               ObjectProvider;                // 0x0068 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UGroupComponent_ORS*                RegistryGroup;                   // 0x0070
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class APlayerController*              Actor;                      // 0x0078 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
```

```
int32_t                          CurrentNetSpeed;                    // 0x0080 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                          ConfiguredInternetSpeed;              // 0x0084 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                          ConfiguredReplicationRate;             // 0x0088 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                           PP_DesaturationMultiplier;              // 0x008C (0x0004)
[0x0000000000004000] (CPF_Config)
float                           PP_HighlightsMultiplier;               // 0x0090 (0x0004)
[0x0000000000004000] (CPF_Config)
float                           PP_MidTonesMultiplier;                 // 0x0094 (0x0004)
[0x0000000000004000] (CPF_Config)
float                           PP_ShadowsMultiplier;                  // 0x0098 (0x0004)
[0x0000000000004000] (CPF_Config)
struct FScriptDelegate                   __EventReceivedController__Delegate;       // 0x00A0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Player");
}

return uClassPointer;
};

void SwitchController(class APlayerController* PC);
void HandleTeamChanged(class APlayerReplicationInfo* PRI);
void HandlePRIRemoved(class APlayerReplicationInfo* PRI);
void HandlePRIAdded(class APlayerReplicationInfo* PRI);
void HandleControllerSet(class UPlayer* PlayerRef);
void eventConstruct();
void EventReceivedController(class UPlayer* PlayerRef);
};

// Class Engine.LocalPlayer
// 0x0418 (0x00B8 - 0x04D0)
class ULocalPlayer : public UPlayer
{
public:
struct FPointer                     VfTable_FObserverInterface;             // 0x00B8 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
int32_t                          ControllerId;                       // 0x00C0 (0x0004)
[0x0000000000000000]
class UGameViewportClient*                ViewportClient;                       // 0x00C8 (0x0008)
[0x0000000000000000]
struct FVector2D                      Origin;                          // 0x00D0 (0x0008)
[0x0000000000000000]
struct FVector2D                      Size;                           // 0x00D8 (0x0008)
[0x0000000000000000]
```

```
class UPostProcessChain*                    PlayerPostProcess;                    // 0x00E0 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<class UPostProcessChain*>            PlayerPostProcessChains;              // 0x00E8
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
unsigned long                               bForceDefaultPostProcessChain : 1;          // 0x00F8
(0x0004) [0x0008000000002000] [0x00000001] (CPF_Transient)
unsigned long                               bWantToResetToMapDefaultPP : 1;             // 0x00F8
(0x0004) [0x0000000000000000] [0x00000002]
unsigned long                               bSentSplitJoin : 1;                   // 0x00F8 (0x0004)
[0x0000000000022002] [0x00000004] (CPF_Const | CPF_Transient | CPF_EditConst)
unsigned long                               bPendingServerAuth : 1;               // 0x00F8 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                               bDrawWorldFullScreen : 1;             // 0x00F8 (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                               bReplayFXDirtied : 1;                 // 0x00F8 (0x0004)
[0x0000000000000000] [0x00000020]
struct FPointer                             ViewState;                            // 0x0100 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                             ViewState2;                           // 0x0108 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FSynchronizedActorVisibilityHistory        ActorVisibilityHistory;               // 0x0110
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FVector                              LastViewLocation;                     // 0x0120 (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FCurrentPostProcessVolumeInfo            CurrentPPInfo;                        // 0x0130
(0x0178) [0x0000000001402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
struct FCurrentPostProcessVolumeInfo            LevelPPInfo;                          // 0x02A8
(0x0178) [0x0000000001402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<struct FPostProcessSettingsOverride>       ActivePPOverrides;                    // 0x0420
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
uint8_t                                     AspectRatioAxisConstraint;            // 0x0430 (0x0001)
[0x0000000000004000] (CPF_Config)
class FString                               LastMap;                              // 0x0438 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class UTranslationContext*                  TagContext;                           // 0x0448 (0x0008)
[0x0000000000000000]
class UOnlineAuthInterface*                 CachedAuthInt_Object;                 // 0x0450
(0x0008) [0x0000000000000000]
class UOnlineAuthInterface*                 CachedAuthInt_Interface;              // 0x0458
(0x0008) [0x0000000000000000]
float                                       ServerAuthTimestamp;                  // 0x0460 (0x0004)
[0x0000000000000000]
int32_t                                     ServerAuthTimeout;                    // 0x0464 (0x0004)
[0x0000000000000000]
int32_t                                     ServerAuthRetryCount;                 // 0x0468 (0x0004)
[0x0000000000000000]
int32_t                                     MaxServerAuthRetryCount;              // 0x046C (0x0004)
[0x0000000000000000]
struct FUniqueNetId                         ServerAuthUID;                        // 0x0470 (0x0048)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                      __bForceDefaultPostProcessChain__ChangeNotify; //
0x04B8 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LocalPlayer");
}

return uClassPointer;
};

void __bForceDefaultPostProcessChain__ChangeNotifyFunc();
void Cleanup(unsigned long bExit);
void eventExit();
void eventNotifyServerConnectionClose();
void eventViewportClosed();
static void StaticOnServerConnectionClose(class UPlayer* ServerConnection);
void OnServerConnectionClose(class UPlayer* ServerConnection);
void ServerAuthFailure();
void eventServerAuthTimedOut();
void OnServerAuthComplete(unsigned long bSuccess, struct FUniqueNetId ServerUID, class
UPlayer* ServerConnection, class FString ExtraInfo);
void ProcessServerAuthResponse(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP, int32_t
AuthTicketUID);
void ProcessClientAuthEndSessionRequest(class UPlayer* ServerConnection);
void ProcessClientAuthRequest(struct FUniqueNetId ServerUID, struct FIpAddr ServerIP, int32_t
ServerPort, unsigned long bSecure);
void eventNotifyServerConnectionOpen();
class FString eventGetNickname();
struct FUniqueNetId eventGetUniqueNetId();
float GetPostProcessPropertyOverride(uint8_t PropertyName);
void SetPostProcessPropertyOverride(uint8_t PropertyName, float PrimarySettingValue, float
SecondarySettingValue);
void ClearActivePostProcessOverride();
void AddActivePostProcessOverride(struct FPostProcessSettings InPostProcessSettings);
struct FVector2D FastProject(struct FVector WorldLoc);
void FastDeProject(struct FVector2D RelativeScreenPos, struct FVector& WorldOrigin, struct
FVector& WorldDirection);
struct FVector Project(struct FVector WorldLoc);
void DeProject(struct FVector2D RelativeScreenPos, struct FVector& WorldOrigin, struct
FVector& WorldDirection);
void TouchPlayerPostProcessChain();
class UPostProcessChain* GetPostProcessChain(int32_t InIndex);
bool RemoveAllPostProcessingChains();
bool RemovePostProcessingChain(int32_t InIndex);
bool InsertPostProcessingChain(class UPostProcessChain* InChain, int32_t InIndex, unsigned
long bInClone);
class UTranslationContext* GetTranslationContext();
void SetControllerId(int32_t NewControllerId);
void ClearPostProcessSettingsOverride(float BlendOutTime);
void OverridePostProcessSettingsCurve(struct FPostProcessSettings OverrideSettings, struct
FInterpCurveFloat& Curve);
void OverridePostProcessSettings(struct FPostProcessSettings OverrideSettings, float
BlendInTime);
bool GetActorVisibility(class AActor* TestActor);
void SendSplitJoin();
bool SpawnPlayActor(class FString URL, class FString& OutError);
};

// Class Engine.NetConnection
// 0xB130 (0x00B8 - 0xB1E8)
class UNetConnection : public UPlayer
{
public:
uint8_t                          UnknownData00[0xB0F0];                // 0x00B8 (0xB0F0)
MISSED OFFSET
TArray<class UChildConnection*>              Children;                            // 0xB1A8 (0x0010)
[0x0000000000602000] (CPF_Transient | CPF_NeedCtorLink)
uint8_t                          UnknownData01[0x1C];                 // 0xB1B8 (0x001C)
MISSED OFFSET
unsigned long                        bUseSessionUID : 1;                  // 0xB1D4 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
uint8_t                          UnknownData02[0x8];                 // 0xB1D8 (0x0008) MISSED
OFFSET
float                            BadConnectionPingThreshold;              // 0xB1E0 (0x0004)
[0x0000000000004000] (CPF_Config)
float                            BadConnectionReceiveTimeThreshold;          // 0xB1E4 (0x0004)
[0x0000000000004000] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NetConnection");
}

return uClassPointer;
};

};

// Class Engine.ChildConnection
// 0x0008 (0xB1E8 - 0xB1F0)
class UChildConnection : public UNetConnection
{
public:
class UNetConnection*                    Parent;                          // 0xB1E8 (0x0008)
[0x0000000000202002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ChildConnection");
}

return uClassPointer;
};

};

// Class Engine.DemoRecConnection
// 0x0000 (0xB1E8 - 0xB1E8)
class UDemoRecConnection : public UNetConnection
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DemoRecConnection");
}

return uClassPointer;
};

};

// Class Engine.Polys
// 0x0018 (0x0060 - 0x0078)
class UPolys : public UObject
{
public:
uint8_t                          UnknownData00[0x18];                    // 0x0060 (0x0018)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Polys");
}

return uClassPointer;
};

};
```

```
// Class Engine.PostProcessChain
// 0x0010 (0x0060 - 0x0070)
class UPostProcessChain : public UObject
{
public:
TArray<class UPostProcessEffect*>                Effects;                                  // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PostProcessChain");
}

return uClassPointer;
};

class UPostProcessEffect* FindPostProcessEffect(struct FName EffectName);
};

// Class Engine.PostProcessEffect
// 0x0025 (0x0060 - 0x0085)
class UPostProcessEffect : public UObject
{
public:
unsigned long                          bShowInEditor : 1;                        // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bShowInGame : 1;                          // 0x0060 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bUseWorldSettings : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bAffectsLightingOnly : 1;                 // 0x0060 (0x0004)
[0x0000000000000000] [0x00000008]
struct FName                           EffectName;                               // 0x0064 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                         NodePosY;                                   // 0x006C (0x0004)
[0x0000000000000000]
int32_t                         NodePosX;                                   // 0x0070 (0x0004)
[0x0000000000000000]
int32_t                         DrawWidth;                                  // 0x0074 (0x0004)
[0x0000000000000000]
int32_t                         DrawHeight;                                 // 0x0078 (0x0004)
[0x0000000000000000]
int32_t                         OutDrawY;                                   // 0x007C (0x0004)
[0x0000000000000000]
int32_t                         InDrawY;                                  // 0x0080 (0x0004)
[0x0000000000000000]
uint8_t                         SceneDPG;                                   // 0x0084 (0x0001)
[0x0000000000000001] (CPF_Edit)
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PostProcessEffect");
}

return uClassPointer;
};

};

// Class Engine.AmbientOcclusionEffect
// 0x005F (0x0085 - 0x00E4)
class UAmbientOcclusionEffect : public UPostProcessEffect
{
public:
struct FLinearColor                  OcclusionColor;                   // 0x0088 (0x0010)
[0x0000000200000001] (CPF_Edit)
float                           OcclusionPower;                   // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           OcclusionScale;                   // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           OcclusionBias;                    // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           MinOcclusion;                     // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                        SSAO2 : 1;                        // 0x00A8 (0x0004)
[0x0000000020000000] [0x00000001]  CPF_Deprecated)
unsigned long                        bAngleBasedSSAO : 1;                   // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                           OcclusionRadius;                  // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           OcclusionAttenuation;                // 0x00B0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
uint8_t                          OcclusionQuality;                  // 0x00B4 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                           OcclusionFadeoutMinDistance;             // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           OcclusionFadeoutMaxDistance;             // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           HaloDistanceThreshold;                // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           HaloDistanceScale;                  // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           HaloOcclusion;                    // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           EdgeDistanceThreshold;                // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           EdgeDistanceScale;                  // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                          FilterDistanceScale;              // 0x00D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         FilterSize;                   // 0x00D8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                          HistoryConvergenceTime;              // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          HistoryWeightConvergenceTime;            // 0x00E0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientOcclusionEffect");
}

return uClassPointer;
};

};

// Class Engine.BlurEffect
// 0x0007 (0x0085 - 0x008C)
class UBlurEffect : public UPostProcessEffect
{
public:
int32_t                         BlurKernelSize;                 // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.BlurEffect");
}

return uClassPointer;
};

};

// Class Engine.DOFEffect
// 0x002F (0x0085 - 0x00B4)
class UDOFEffect : public UPostProcessEffect
{
public:
float                          FalloffExponent;                // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                          BlurKernelSize;                  // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MaxNearBlurAmount;                    // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MinBlurAmount;                     // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MaxFarBlurAmount;                    // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                         FocusType;                     // 0x009C (0x0001)
[0x0000000000000001] (CPF_Edit)
float                          FocusInnerRadius;                  // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FocusDistance;                    // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                          FocusPosition;                    // 0x00A8 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DOFEffect");
}

return uClassPointer;
};

};

// Class Engine.DOFAndBloomEffect
// 0x002C (0x00B4 - 0x00E0)
class UDOFAndBloomEffect : public UDOFEffect
{
public:
float                          BloomScale;                      // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BloomThreshold;                    // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                          BloomTint;                      // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BloomScreenBlendThreshold;                // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          SceneMultiplier;                   // 0x00C8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                          BlurBloomKernelSize;                  // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bEnableReferenceDOF : 1;                 // 0x00D0 (0x0004)
[0x0000000020000000] [0x00000001]  CPF_Deprecated)
uint8_t                         DepthOfFieldType;                  // 0x00D4 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                         DepthOfFieldQuality;                 // 0x00D5 (0x0001)
```

```
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                    BokehTexture;                    // 0x00D8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DOFAndBloomEffect");
}

return uClassPointer;
};

};

// Class Engine.DOFBloomMotionBlurEffect
// 0x0014 (0x00E0 - 0x00F4)
class UDOFBloomMotionBlurEffect : public UDOFAndBloomEffect
{
public:
float                                MaxVelocity;                     // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                MotionBlurAmount;                // 0x00E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                        FullMotionBlur : 1;              // 0x00E8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                CameraRotationThreshold;         // 0x00EC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                CameraTranslationThreshold;      // 0x00F0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DOFBloomMotionBlurEffect");
}

return uClassPointer;
};

};

// Class Engine.UberPostProcessEffect
// 0x00A0 (0x00F4 - 0x0194)
class UUberPostProcessEffect : public UDOFBloomMotionBlurEffect
{
```

```
public:
struct FVector                          SceneShadows;                          // 0x00F8 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          SceneHighLights;                       // 0x0104 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          SceneMidTones;                         // 0x0110 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                              SceneDesaturation;                     // 0x011C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                          SceneColorize;                         // 0x0120 (0x000C)
[0x0000000000000001] (CPF_Edit)
uint8_t                               TonemapperType;                        // 0x012C (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                               PostProcessAAType;                     // 0x012D (0x0001)
[0x0000000000000001] (CPF_Edit)
float                              TonemapperRange;                       // 0x0130 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              TonemapperToeFactor;                   // 0x0134 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              TonemapperScale;                       // 0x0138 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              MotionBlurSoftEdgeKernelSize;          // 0x013C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bEnableImageGrain : 1;                 // 0x0140 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bScaleEffectsWithViewSize : 1;         // 0x0140 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bEnableHDRTonemapper : 1;              // 0x0140 (0x0004)
[0x0000000020000000] [0x00000004]  CPF_Deprecated)
float                              SceneImageGrainScale;                  // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              BloomWeightSmall;                      // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              BloomWeightMedium;                     // 0x014C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              BloomWeightLarge;                      // 0x0150 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              BloomSizeScaleSmall;                   // 0x0154 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              BloomSizeScaleMedium;                  // 0x0158 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              BloomSizeScaleLarge;                   // 0x015C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              EdgeDetectionThreshold;                // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FLUTBlender                          PreviousLUTBlender;                    // 0x0168 (0x0028)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
float                              SceneHDRTonemapperScale;               // 0x0190 (0x0004)
[0x0000000020000000]  CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UberPostProcessEffect");
}

return uClassPointer;
};

};

// Class Engine.DwTriovizImplEffect
// 0x0003 (0x0085 - 0x0088)
class UDwTriovizImplEffect : public UPostProcessEffect
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DwTriovizImplEffect");
}

return uClassPointer;
};

};

// Class Engine.MaterialEffect
// 0x000B (0x0085 - 0x0090)
class UMaterialEffect : public UPostProcessEffect
{
public:
class UMaterialInterface*               Material;                        // 0x0088 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialEffect");
}

return uClassPointer;
};

};
```

```
// Class Engine.MotionBlurEffect
// 0x0017 (0x0085 - 0x009C)
class UMotionBlurEffect : public UPostProcessEffect
{
public:
float                                   MaxVelocity;                            // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MotionBlurAmount;                       // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           FullMotionBlur : 1;                     // 0x0090 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                   CameraRotationThreshold;                // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   CameraTranslationThreshold;             // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MotionBlurEffect");
}

return uClassPointer;
};

};

// Class Engine.PrimitiveComponentFactory
// 0x0004 (0x0060 - 0x0064)
class UPrimitiveComponentFactory : public UObject
{
public:
unsigned long                           CollideActors : 1;                      // 0x0060 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                           BlockActors : 1;                        // 0x0060 (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                           BlockZeroExtent : 1;                    // 0x0060 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                           BlockNonZeroExtent : 1;                 // 0x0060 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                           BlockRigidBody : 1;                     // 0x0060 (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                           HiddenGame : 1;                         // 0x0060 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                           HiddenEditor : 1;                       // 0x0060 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                           CastShadow : 1;                         // 0x0060 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrimitiveComponentFactory");
}

return uClassPointer;
};

};

// Class Engine.MeshComponentFactory
// 0x0014 (0x0064 - 0x0078)
class UMeshComponentFactory : public UPrimitiveComponentFactory
{
public:
TArray<class UMaterialInterface*>                Materials;                              // 0x0068 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MeshComponentFactory");
}

return uClassPointer;
};

};

// Class Engine.StaticMeshComponentFactory
// 0x0008 (0x0078 - 0x0080)
class UStaticMeshComponentFactory : public UMeshComponentFactory
{
public:
class UStaticMesh*                          StaticMesh;                             // 0x0078 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticMeshComponentFactory");
}

return uClassPointer;
};

};

// Class Engine.ProcessReplayCommandlet
// 0x0004 (0x00B4 - 0x00B8)
class UProcessReplayCommandlet : public UCommandlet
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ProcessReplayCommandlet");
}

return uClassPointer;
};

};

// Class Engine.ReachSpec
// 0x0068 (0x0060 - 0x00C8)
class UReachSpec : public UObject
{
public:
struct FPointer                          NavOctreeObject;                          // 0x0060 (0x0008)
[0x0000000000023002] (CPF_Const | CPF_Native | CPF_Transient | CPF_EditConst)
int32_t                                  Distance;                                 // 0x0068 (0x0004)
[0x0000000000000000]
struct FVector                           Direction;                                // 0x006C (0x000C)
[0x0000000000000000]
class ANavigationPoint*                  Start;                                    // 0x0078 (0x0008)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
struct FActorReference                   End;                                      // 0x0080 (0x0018)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                                  CollisionRadius;                          // 0x0098 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                                  CollisionHeight;                          // 0x009C (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                                  reachFlags;                               // 0x00A0 (0x0004)
[0x0000000000000000]
int32_t                                  MaxLandingVelocity;                       // 0x00A4 (0x0004)
[0x0000000000000000]
uint8_t                                  bPruned;                                  // 0x00A8 (0x0001)
[0x0000000000000000]
uint8_t                                  PathColorIndex;                           // 0x00A9 (0x0001)
```

```
[0x0000000000000000]
unsigned long                      bAddToNavigationOctree : 1;             // 0x00AC (0x0004)
[0x0000000000020002] [0x00000001] (CPF_Const | CPF_EditConst)
unsigned long                      bCanCutCorners : 1;                     // 0x00AC (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                      bCheckForObstructions : 1;              // 0x00AC (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                      bSkipPrune : 1;                         // 0x00AC (0x0004)
[0x0000000000000002] [0x00000008] (CPF_Const)
unsigned long                      bDisabled : 1;                          // 0x00AC (0x0004)
[0x0000000000020001] [0x00000010] (CPF_Edit | CPF_EditConst)
TArray<class UClass*>                  PruneSpecList;                      // 0x00B0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class AActor*                      BlockedBy;                              // 0x00C0 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ReachSpec");
}

return uClassPointer;
};

bool IsBlockedFor(class APawn* P);
struct FVector GetDirection();
class ANavigationPoint* GetEnd();
int32_t CostFor(class APawn* P);
};

// Class Engine.AdvancedReachSpec
// 0x0000 (0x00C8 - 0x00C8)
class UAdvancedReachSpec : public UReachSpec
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AdvancedReachSpec");
}

return uClassPointer;
};
```

```
};

// Class Engine.CeilingReachSpec
// 0x0000 (0x00C8 - 0x00C8)
class UCeilingReachSpec : public UReachSpec
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CeilingReachSpec");
}

return uClassPointer;
};

};

// Class Engine.ForcedReachSpec
// 0x0000 (0x00C8 - 0x00C8)
class UForcedReachSpec : public UReachSpec
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForcedReachSpec");
}

return uClassPointer;
};

};

// Class Engine.CoverSlipReachSpec
// 0x0001 (0x00C8 - 0x00C9)
class UCoverSlipReachSpec : public UForcedReachSpec
{
public:
uint8_t                                   SpecDirection;                            // 0x00C8 (0x0001)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CoverSlipReachSpec");
}

return uClassPointer;
};

};

// Class Engine.FloorToCeilingReachSpec
// 0x0000 (0x00C8 - 0x00C8)
class UFloorToCeilingReachSpec : public UForcedReachSpec
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FloorToCeilingReachSpec");
}

return uClassPointer;
};

};

// Class Engine.MantleReachSpec
// 0x0004 (0x00C8 - 0x00CC)
class UMantleReachSpec : public UForcedReachSpec
{
public:
unsigned long                          bClimbUp : 1;                          // 0x00C8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MantleReachSpec");
}

return uClassPointer;
};
```

```
};

// Class Engine.SlotToSlotReachSpec
// 0x0001 (0x00C8 - 0x00C9)
class USlotToSlotReachSpec : public UForcedReachSpec
{
public:
uint8_t                                   SpecDirection;                              // 0x00C8 (0x0001)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SlotToSlotReachSpec");
}

return uClassPointer;
};

};

// Class Engine.SwatTurnReachSpec
// 0x0001 (0x00C8 - 0x00C9)
class USwatTurnReachSpec : public UForcedReachSpec
{
public:
uint8_t                                   SpecDirection;                              // 0x00C8 (0x0001)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SwatTurnReachSpec");
}

return uClassPointer;
};

};

// Class Engine.WallTransReachSpec
// 0x0000 (0x00C8 - 0x00C8)
class UWallTransReachSpec : public UForcedReachSpec
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WallTransReachSpec");
}

return uClassPointer;
};

};

// Class Engine.ProscribedReachSpec
// 0x0000 (0x00C8 - 0x00C8)
class UProscribedReachSpec : public UReachSpec
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ProscribedReachSpec");
}

return uClassPointer;
};

};

// Class Engine.TeleportReachSpec
// 0x0000 (0x00C8 - 0x00C8)
class UTeleportReachSpec : public UReachSpec
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TeleportReachSpec");
}

return uClassPointer;
};

};

// Class Engine.SavedMove
// 0x00B8 (0x0060 - 0x0118)
class USavedMove : public UObject
{
public:
class USavedMove*                    NextMove;                              // 0x0060 (0x0008)
[0x0000000000000000]
float                         TimeStamp;                              // 0x0068 (0x0004)
[0x0000000000000000]
float                         delta;                              // 0x006C (0x0004)
[0x0000000000000000]
unsigned long                      bRun : 1;                              // 0x0070 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                      bDuck : 1;                              // 0x0070 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                      bPressedJump : 1;                         // 0x0070 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                      bDoubleJump : 1;                          // 0x0070 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                      bPreciseDestination : 1;                   // 0x0070 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                      bForceRMVelocity : 1;                      // 0x0070 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                      bForceMaxAccel : 1;                        // 0x0070 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                      bRootMotionFromInterpCurve : 1;             // 0x0070 (0x0004)
[0x0000000000000000] [0x00000080]
uint8_t                         DoubleClickMove;                          // 0x0074 (0x0001)
[0x0000000000000000]
uint8_t                         SavedPhysics;                           // 0x0075 (0x0001)
[0x0000000000000000]
uint8_t                         RootMotionMode;                           // 0x0076 (0x0001)
[0x0000000000000000]
struct FVector                       StartLocation;                          // 0x0078 (0x000C)
[0x0000000000000000]
struct FVector                       StartRelativeLocation;                     // 0x0084 (0x000C)
[0x0000000000000000]
struct FVector                       StartVelocity;                          // 0x0090 (0x000C)
[0x0000000000000000]
struct FVector                       StartFloor;                           // 0x009C (0x000C)
[0x0000000000000000]
struct FVector                       SavedLocation;                           // 0x00A8 (0x000C)
[0x0000000000000000]
struct FVector                       SavedVelocity;                           // 0x00B4 (0x000C)
[0x0000000000000000]
struct FVector                       SavedRelativeLocation;                      // 0x00C0 (0x000C)
[0x0000000000000000]
struct FVector                       RMVelocity;                            // 0x00CC (0x000C)
[0x0000000000000000]
struct FVector                       Acceleration;                           // 0x00D8 (0x000C)
```

```
[0x0000000000000000]
struct FRotator                    Rotation;                          // 0x00E4 (0x000C)
[0x0000000000000000]
class AActor*                      StartBase;                         // 0x00F0 (0x0008)
[0x0000000000000000]
class AActor*                      EndBase;                           // 0x00F8 (0x0008)
[0x0000000000000000]
float                          CustomTimeDilation;                    // 0x0100 (0x0004)
[0x0000000000000000]
float                          AccelDotThreshold;                     // 0x0104 (0x0004)
[0x0000000000000000]
float                          RootMotionInterpCurrentTime;               // 0x0108 (0x0004)
[0x0000000000000000]
struct FVector                     RootMotionInterpCurveLastValue;            // 0x010C
(0x000C) [0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SavedMove");
}

return uClassPointer;
};

class FString GetDebugString();
static uint8_t SetFlags(uint8_t Flags, class APlayerController* PC);
uint8_t CompressedFlags();
void ResetMoveFor(class APawn* P);
void PrepMoveFor(class APawn* P);
void SetMoveFor(class APlayerController* P, float DeltaTime, struct FVector newAccel, uint8_t
InDoubleClick);
bool CanCombineWith(class USavedMove* NewMove, class APawn* inPawn, float MaxDelta);
void SetInitialPosition(class APawn* P);
struct FVector GetStartLocation();
bool IsImportantMove(struct FVector CompareAccel);
void PostUpdate(class APlayerController* P);
void Clear();
};

// Class Engine.SaveGameSummary
// 0x0018 (0x0060 - 0x0078)
class USaveGameSummary : public UObject
{
public:
struct FName                       BaseLevel;                         // 0x0060 (0x0008)
[0x0000000000000000]
class FString                    Description;                         // 0x0068 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SaveGameSummary");
}

return uClassPointer;
};

};

// Class Engine.ScriptViewportClient
// 0x0008 (0x0060 - 0x0068)
class UScriptViewportClient : public UObject
{
public:
struct FPointer                         VfTable_FViewportClient;                  // 0x0060 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ScriptViewportClient");
}

return uClassPointer;
};

};

// Class Engine.GameViewportClient
// 0x01E0 (0x0068 - 0x0248)
class UGameViewportClient : public UScriptViewportClient
{
public:
struct FPointer                         VfTable_FExec;                         // 0x0068 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                         Viewport;                           // 0x0070 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FPointer                         ViewportFrame;                         // 0x0078 (0x0008)
[0x0000000000000002] (CPF_Const)
TArray<class UInteraction*>                   GlobalInteractions;                        // 0x0080 (0x0010)
[0x0000008000500000] (CPF_NeedCtorLink)
class UClass*                           UIControllerClass;                     // 0x0090 (0x0008)
[0x0000000000000000]
class UUIInteraction*                      UIController;                          // 0x0098 (0x0008)
```

```
[0x0000000000000000]
class UConsole*                    ViewportConsole;                    // 0x00A0 (0x0008)
[0x0000000000000000]
struct FExportShowFlags_Mirror               ShowFlags;                        // 0x00A8 (0x0010)
[0x0000000000000002] (CPF_Const)
class FString                     LoadingMessage;                     // 0x00B8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                     SavingMessage;                      // 0x00C8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                     ConnectingMessage;                  // 0x00D8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                     PausedMessage;                      // 0x00E8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                     PrecachingMessage;                  // 0x00F8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
unsigned long                      bShowTitleSafeZone : 1;             // 0x0108 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                      bDisplayHardwareMouseCursor : 1;          // 0x0108
(0x0004) [0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                      bOverrideDiffuseAndSpecular : 1;          // 0x0108 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                      bIsPlayInEditorViewport : 1;             // 0x0108 (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
unsigned long                      bShowSystemMouseCursor : 1;              // 0x0108
(0x0004) [0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                      bDisableWorldRendering : 1;              // 0x0108 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                      bCapturedWorldRendering : 1;             // 0x0108 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                      bDebugNoGFxUI : 1;                  // 0x0108 (0x0004)
[0x0000000000004000] [0x00000080] (CPF_Config)
struct FTitleSafeZoneArea                  TitleSafeZone;                   // 0x010C (0x0010)
[0x0000000000000000]
uint8_t                          GamepadInputAPI;                    // 0x011C (0x0001)
[0x0000000000000000]
uint8_t                          DesiredSplitscreenType;              // 0x011D (0x0001)
[0x0000000000000000]
uint8_t                          ActiveSplitscreenType;               // 0x011E (0x0001)
[0x0000008000000000]
uint8_t                          Default2PSplitType;                 // 0x011F (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                          Default3PSplitType;                 // 0x0120 (0x0001)
[0x0000000000000002] (CPF_Const)
TArray<struct FGamepadInfo>                Gamepads;                        // 0x0128 (0x0010)
[0x0000004000400000] (CPF_NeedCtorLink)
TArray<struct FSplitscreenData>             SplitscreenInfo;                 // 0x0138 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                     ProgressMessage[0x2];               // 0x0148 (0x0020)
[0x0000000000400000] (CPF_NeedCtorLink)
float                            ProgressTimeOut;                    // 0x0168 (0x0004)
[0x0000000000000000]
float                            ProgressFadeTime;                   // 0x016C (0x0004)
[0x0000000000000000]
TArray<struct FDebugDisplayProperty>           DebugProperties;                 // 0x0170
```

```
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
struct FPointer                    ScaleformInteraction;                  // 0x0180 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FScriptDelegate             __HandleInputKey__Delegate;                // 0x0188
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate             __HandleInputAxis__Delegate;               // 0x01A0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate             __HandleInputChar__Delegate;               // 0x01B8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate             __EventScaleformEnabledChanged__Delegate;     //
0x01D0 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate             __EventGamepadInputAPIChanged__Delegate;      //
0x01E8 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate             __EventGamepadConnectionStatusChanged__Delegate;//
0x0200 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate             __EventGampadConnected__Delegate;           // 0x0218
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate             __EventGampadDisconnected__Delegate;         // 0x0230
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameViewportClient");
}

return uClassPointer;
};

void eventSetHardwareMouseCursorVisibility(unsigned long bIsVisible);
void DebugSetUISystemEnabled(unsigned long bOldUISystemActive, unsigned long
bGFxUISystemActive);
bool IsScaleformEnabled();
void DisableScaleform();
void EnableScaleform();
void BecomePrimaryPlayer(int32_t PlayerIndex);
void OnPrimaryPlayerSwitch(class ULocalPlayer* OldPrimaryPlayer, class ULocalPlayer*
NewPrimaryPlayer);
void FixupOwnerReferences(TArray<int32_t> IDMappings);
class ULocalPlayer* GetPlayerOwner(int32_t PlayerIndex);
void ClearProgressMessages();
void SetProgressTime(float T);
void NotifyConnectionError(uint8_t MessageType, class FString Message, class FString Title);
void eventSetProgressMessage(uint8_t MessageType, class FString Message, class FString Title,
unsigned long bIgnoreFutureNetworkMessages);
int32_t RemoveLocalPlayer(class ULocalPlayer* ExistingPlayer);
int32_t AddLocalPlayer(class ULocalPlayer* NewPlayer);
void eventNotifyPlayerRemoved(int32_t PlayerIndex, class ULocalPlayer* RemovedPlayer);
void NotifyPlayerAdded(int32_t PlayerIndex, class ULocalPlayer* AddedPlayer);
void DrawTransitionMessage(class UCanvas* Canvas, class FString Message);
void DrawTransition(class UCanvas* Canvas);
void DisplayProgressMessage(class UCanvas* Canvas);
void eventPostRender(class UCanvas* Canvas);
void DrawTitleSafeArea(class UCanvas* Canvas);
void eventTick(float DeltaTime);
bool CalculateDeadZoneForAllSides(class ULocalPlayer* LPlayer, class UCanvas* Canvas,
unsigned long bUseMaxPercent, float& fTopSafeZone, float& fBottomSafeZone, float&
fLeftSafeZone, float& fRightSafeZone);
void CalculateSafeZoneValues(class UCanvas* Canvas, int32_t LocalPlayerIndex, unsigned long
bUseMaxPercent, float& out_Horizontal, float& out_Vertical);
void GetPixelSizeOfScreen(class UCanvas* Canvas, int32_t LocalPlayerIndex, float& out_Width,
float& out_Height);
bool HasRightSafeZone(int32_t LocalPlayerIndex);
bool HasLeftSafeZone(int32_t LocalPlayerIndex);
bool HasBottomSafeZone(int32_t LocalPlayerIndex);
bool HasTopSafeZone(int32_t LocalPlayerIndex);
int32_t ConvertLocalPlayerToGamePlayerIndex(class ULocalPlayer* LPlayer);
void eventGetSubtitleRegion(struct FVector2D& MinPos, struct FVector2D& MaxPos);
void eventLayoutPlayers();
void UpdateActiveSplitscreenType();
uint8_t GetSplitscreenConfiguration();
void SetSplitscreenConfiguration(uint8_t SplitType);
void eventGameSessionEnded();
class UInteraction* GetInteraction(class UClass* InteractionClass);
void RemoveInteraction(class UInteraction* RemInteraction);
int32_t eventInsertInteraction(class UInteraction* NewInteraction, int32_t InIndex);
bool eventCreateInitialPlayer(class FString& OutError);
bool eventInit(class FString& OutError);
class ULocalPlayer* eventFindPlayerByControllerId(int32_t ControllerId);
bool eventRemovePlayer(class ULocalPlayer* ExPlayer);
class ULocalPlayer* eventCreatePlayer(int32_t ControllerId, unsigned long bSpawnActor, class
FString& OutError);
void SetMouse(int32_t X, int32_t Y);
void ForceUpdateMouseCursor(unsigned long bSetCursor);
void NotifySplitscreenLayoutChanged();
void SetCustomInteractionObject(class UInteraction* InInteraction);
class UClass* GetCustomInteractionClass(int32_t InIndex);
int32_t GetNumCustomInteractions();
bool ShouldForceFullscreenViewport();
struct FVector2D GetMousePosition();
bool IsFullScreenViewport();
void GetViewportSize(struct FVector2D& out_ViewportSize);
class FString ConsoleCommand(class FString Command);
void EventGampadDisconnected(class UGameViewportClient* GVC);
void EventGampadConnected(class UGameViewportClient* GVC);
void EventGamepadConnectionStatusChanged(class UGameViewportClient* GVC, int32_t
ControllerId, unsigned long bConnected);
void EventGamepadInputAPIChanged(class UGameViewportClient* GVC, int32_t ControllerId,
uint8_t InputAPI);
void EventScaleformEnabledChanged(class UGameViewportClient* GVC);
bool HandleInputChar(int32_t ControllerId, class FString Unicode);
bool HandleInputAxis(int32_t ControllerId, struct FName Key, float delta, float DeltaTime,
unsigned long bGamepad);
bool HandleInputKey(int32_t ControllerId, struct FName Key, uint8_t EventType, float
AmountDepressed, unsigned long bGamepad);
};

// Class Engine.Selection
// 0x0030 (0x0060 - 0x0090)
class USelection : public UObject
{
public:
uint8_t                                  UnknownData00[0x30];                        // 0x0060 (0x0030)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Selection");
}

return uClassPointer;
};

};

// Class Engine.ServerCommandlet
// 0x0004 (0x00B4 - 0x00B8)
class UServerCommandlet : public UCommandlet
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ServerCommandlet");
}

return uClassPointer;
};

};

// Class Engine.Settings
// 0x0040 (0x0060 - 0x00A0)
class USettings : public UObject
{
public:
TArray<struct FLocalizedStringSetting>          LocalizedSettings;                        // 0x0060
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
```

```
TArray<struct FSettingsProperty>                Properties;                              // 0x0070 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FLocalizedStringSettingMetaData>     LocalizedSettingsMappings;                  //
0x0080 (0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FSettingsPropertyPropertyMetaData>   PropertyMappings;                        //
0x0090 (0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Settings");
}

return uClassPointer;
};

void UpdateFromURL(class AGameInfo* Game, class FString& URL);
void BuildURL(class FString& URL);
void AppendContextsToURL(class FString& URL);
void AppendPropertiesToURL(class FString& URL);
void AppendDataBindingsToURL(class FString& URL);
void GetQoSAdvertisedStringSettings(TArray<struct FLocalizedStringSetting>& QoSSettings);
void GetQoSAdvertisedProperties(TArray<struct FSettingsProperty>& QoSProps);
bool GetRangedPropertyValue(int32_t PropertyId, float& OutValue);
bool SetRangedPropertyValue(int32_t PropertyId, float NewValue);
bool GetPropertyRange(int32_t PropertyId, float& OutMinValue, float& OutMaxValue, float&
RangeIncrement, uint8_t& bFormatAsInt);
bool GetPropertyMappingType(int32_t PropertyId, uint8_t& OutType);
bool HasStringSetting(int32_t SettingId);
bool HasProperty(int32_t PropertyId);
void UpdateProperties(unsigned long bShouldAddIfMissing, TArray<struct FSettingsProperty>&
Props);
void UpdateStringSettings(unsigned long bShouldAddIfMissing, TArray<struct
FLocalizedStringSetting>& Settings);
uint8_t GetPropertyType(int32_t PropertyId);
bool GetPropertyValueId(int32_t PropertyId, int32_t& ValueId);
bool SetPropertyValueId(int32_t PropertyId, int32_t ValueId);
bool GetStringProperty(int32_t PropertyId, class FString& Value);
void SetStringProperty(int32_t PropertyId, class FString Value);
bool GetIntProperty(int32_t PropertyId, int32_t& Value);
void SetIntProperty(int32_t PropertyId, int32_t Value);
bool GetFloatProperty(int32_t PropertyId, float& Value);
void SetFloatProperty(int32_t PropertyId, float Value);
bool SetPropertyFromStringByName(struct FName PropertyName, class FString& NewValue);
class FString GetPropertyAsStringByName(struct FName PropertyName);
class FString GetPropertyAsString(int32_t PropertyId);
class FString GetPropertyColumnHeader(int32_t PropertyId);
struct FName GetPropertyName(int32_t PropertyId);
bool GetPropertyId(struct FName PropertyName, int32_t& PropertyId);
bool SetStringSettingValueFromStringByName(struct FName StringSettingName, class FString&
NewValue);
struct FName GetStringSettingValueNameByName(struct FName StringSettingName);
struct FName GetStringSettingValueName(int32_t StringSettingId, int32_t ValueIndex);
bool IsWildcardStringSetting(int32_t StringSettingId);
class FString GetStringSettingColumnHeader(int32_t StringSettingId);
struct FName GetStringSettingName(int32_t StringSettingId);
bool GetStringSettingId(struct FName StringSettingName, int32_t& StringSettingId);
bool GetStringSettingValueByName(struct FName StringSettingName, int32_t& ValueIndex);
void SetStringSettingValueByName(struct FName StringSettingName, int32_t ValueIndex,
unsigned long bShouldAutoAdd);
bool GetStringSettingValueNames(int32_t StringSettingId, TArray<struct FIdToStringMapping>&
Values);
bool IncrementStringSettingValue(int32_t StringSettingId, int32_t Direction, unsigned long
bShouldWrap);
bool GetStringSettingValue(int32_t StringSettingId, int32_t& ValueIndex);
void SetStringSettingValue(int32_t StringSettingId, int32_t ValueIndex, unsigned long
bShouldAutoAdd);
static void GetSettingsDataDateTime(struct FSettingsData& Data, int32_t& OutInt1, int32_t&
OutInt2);
static void GetSettingsDataBlob(struct FSettingsData& Data, TArray<uint8_t>& OutBlob);
static int32_t GetSettingsDataInt(struct FSettingsData& Data);
static float GetSettingsDataFloat(struct FSettingsData& Data);
static void EmptySettingsData(struct FSettingsData& Data);
static void SetSettingsData(struct FSettingsData& Data, struct FSettingsData& Data2Copy);
static void SetSettingsDataBlob(struct FSettingsData& Data, TArray<uint8_t>& InBlob);
static void SetSettingsDataDateTime(int32_t InInt1, int32_t InInt2, struct FSettingsData& Data);
static void SetSettingsDataInt(int32_t InInt, struct FSettingsData& Data);
static void SetSettingsDataFloat(float InFloat, struct FSettingsData& Data);
};

// Class Engine.OnlineGameSearch
// 0x00B4 (0x00A0 - 0x0154)
class UOnlineGameSearch : public USettings
{
public:
int32_t                                MaxSearchResults;                        // 0x00A0 (0x0004)
[0x0000000000000000]
struct FLocalizedStringSetting                 Query;                                  // 0x00A4 (0x000C)
[0x0000000000000000]
unsigned long                          bIsLanQuery : 1;                        // 0x00B0 (0x0004)
[0x0000000040000000] [0x00000001] (CPF_EditInlineNotify)
unsigned long                          bUsesArbitration : 1;                   // 0x00B0 (0x0004)
[0x0000000040000000] [0x00000002] (CPF_EditInlineNotify)
unsigned long                          bIsSearchInProgress : 1;                // 0x00B0 (0x0004)
[0x0000000000000002] [0x00000004] (CPF_Const)
class UClass*                          GameSettingsClass;                      // 0x00B8 (0x0008)
[0x0000000000000000]
TArray<struct FOnlineGameSearchResult>          Results;                               // 0x00C0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FOverrideSkill                    ManualSkillOverride;                    // 0x00D0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FNamedObjectProperty>            NamedProperties;                             // 0x0108
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
struct FOnlineGameSearchQuery                 FilterQuery;                            // 0x0118 (0x0020)
```

```
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                          AdditionalSearchCriteria;                  // 0x0138 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                           PingBucketSize;                          // 0x0148 (0x0004)
[0x0000000000000000]
int32_t                           NumPingProbes;                            // 0x014C (0x0004)
[0x0000000000000000]
int32_t                           MaxPingBytes;                            // 0x0150 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineGameSearch");
}

return uClassPointer;
};

class UOnlineGameSettings* eventCreateOnlineGameSettings();
void eventSortSearchResults();
void SetSkillOverride(int32_t LeaderboardId, TArray<struct FUniqueNetId>& Players);
};

// Class Engine.OnlineGameSettings
// 0x00A8 (0x00A0 - 0x0148)
class UOnlineGameSettings : public USettings
{
public:
int32_t                           NumPublicConnections;                     // 0x00A0 (0x0004)
[0x0000000040000000] (CPF_EditInlineNotify)
int32_t                           NumPrivateConnections;                    // 0x00A4 (0x0004)
[0x0000000040000000] (CPF_EditInlineNotify)
int32_t                           NumOpenPublicConnections;                   // 0x00A8 (0x0004)
[0x0000000040000000] (CPF_EditInlineNotify)
int32_t                           NumOpenPrivateConnections;                  // 0x00AC (0x0004)
[0x0000000040000000] (CPF_EditInlineNotify)
uint64_t                           ServerNonce;                            // 0x00B0 (0x0008)
[0x0000000000000002] (CPF_Const)
unsigned long                          bShouldAdvertise : 1;                   // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000001] (CPF_EditInlineNotify)
unsigned long                          bIsLanMatch : 1;                      // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000002] (CPF_EditInlineNotify)
unsigned long                          bUsesStats : 1;                      // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000004] (CPF_EditInlineNotify)
unsigned long                          bAllowJoinInProgress : 1;                 // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000008] (CPF_EditInlineNotify)
unsigned long                          bAllowInvites : 1;                     // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000010] (CPF_EditInlineNotify)
unsigned long                          bUsesPresence : 1;                     // 0x00B8 (0x0004)
```

```
[0x0000000040000000] [0x00000020] (CPF_EditInlineNotify)
unsigned long                          bAllowJoinViaPresence : 1;              // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000040] (CPF_EditInlineNotify)
unsigned long                          bAllowJoinViaPresenceFriendsOnly : 1;       // 0x00B8
(0x0004) [0x0000000040000000] [0x00000080] (CPF_EditInlineNotify)
unsigned long                          bUsesArbitration : 1;                   // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000100] (CPF_EditInlineNotify)
unsigned long                          bAntiCheatProtected : 1;                // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000200] (CPF_EditInlineNotify)
unsigned long                          bWasFromInvite : 1;                     // 0x00B8 (0x0004)
[0x0000000000000002] [0x00000400] (CPF_Const)
unsigned long                          bIsDedicated : 1;                       // 0x00B8 (0x0004)
[0x0000000040000000] [0x00000800] (CPF_EditInlineNotify)
unsigned long                          bHasSkillUpdateInProgress : 1;          // 0x00B8 (0x0004)
[0x0000000000000002] [0x00001000] (CPF_Const)
unsigned long                          bShouldShrinkArbitratedSessions : 1;        // 0x00B8
(0x0004) [0x0000000000000002] [0x00002000] (CPF_Const)
class FString                          OwningPlayerName;                       // 0x00C0 (0x0010)
[0x0000000040400000] (CPF_NeedCtorLink | CPF_EditInlineNotify)
struct FUniqueNetId                       OwningPlayerId;                      // 0x00D0 (0x0048)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                PingInMs;                               // 0x0118 (0x0004)
[0x0000000040000000] (CPF_EditInlineNotify)
float                                  MatchQuality;                           // 0x011C (0x0004)
[0x0000000040000000] (CPF_EditInlineNotify)
uint8_t                                GameState;                              // 0x0120 (0x0001)
[0x0000000040000002] (CPF_Const | CPF_EditInlineNotify)
int32_t                                BuildUniqueId;                          // 0x0124 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<struct FName>                      DataboundPropertiesToAdvertise;          // 0x0128
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class FString                          Payload;                                // 0x0138 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineGameSettings");
}

return uClassPointer;
};

};

// Class Engine.ShaderCache
// 0x0078 (0x0060 - 0x00D8)
class UShaderCache : public UObject
{
public:
```

```
uint8_t                                UnknownData00[0x78];                    // 0x0060 (0x0078)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ShaderCache");
}

return uClassPointer;
};

};

// Class Engine.ShadowMap1D
// 0x0068 (0x0060 - 0x00C8)
class UShadowMap1D : public UObject
{
public:
uint8_t                                UnknownData00[0x68];                    // 0x0060 (0x0068)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ShadowMap1D");
}

return uClassPointer;
};

};

// Class Engine.ShadowMap2D
// 0x003C (0x0060 - 0x009C)
class UShadowMap2D : public UObject
{
public:
class UShadowMapTexture2D*                   Texture;                              // 0x0060 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FVector2D                        CoordinateScale;                      // 0x0068 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FVector2D                        CoordinateBias;                       // 0x0070 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FGuid                         LightGuid;                           // 0x0078 (0x0010)
[0x0000000000000002] (CPF_Const)
```

```
unsigned long                          bIsShadowFactorTexture : 1;                // 0x0088 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
class UInstancedStaticMeshComponent*            Component;                              // 0x0090
(0x0008) [0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)
int32_t                                InstanceIndex;                          // 0x0098 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ShadowMap2D");
}

return uClassPointer;
};

};

// Class Engine.SmokeTestCommandlet
// 0x0004 (0x00B4 - 0x00B8)
class USmokeTestCommandlet : public UCommandlet
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SmokeTestCommandlet");
}

return uClassPointer;
};

};

// Class Engine.SpeechRecognition
// 0x00C8 (0x0060 - 0x0128)
class USpeechRecognition : public UObject
{
public:
class FString                          Language;                               // 0x0060 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                  ConfidenceThreshhold;                   // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FRecogVocabulary>                Vocabularies;                           // 0x0078
```

```
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<uint8_t>                          VoiceData;                              // 0x0088 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                          WorkingVoiceData;                       // 0x0098 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                          UserData;                               // 0x00A8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FRecogUserData                         InstanceData[0x4];                      // 0x00B8 (0x0060)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                            bDirty : 1;                             // 0x0118 (0x0004)
[0x0000000000202000] [0x00000001] (CPF_Transient)
unsigned long                            bInitialised : 1;                       // 0x0118 (0x0004)
[0x0000000000202000] [0x00000002] (CPF_Transient)
struct FPointer                          FnxVoiceData;                           // 0x0120 (0x0008)
[0x0000000000201002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpeechRecognition");
}

return uClassPointer;
};

};

// Class Engine.StaticMesh
// 0x01A0 (0x0060 - 0x0200)
class UStaticMesh : public UObject
{
public:
uint8_t                                  UnknownData00[0x10];                    // 0x0060 (0x0010)
MISSED OFFSET
TArray<struct FStaticMeshLODInfo>             LODInfo;                                // 0x0070 (0x0010)
[0x0000000000001041] (CPF_Edit | CPF_EditConstArray | CPF_Native)
float                                    LODDistanceRatio;                       // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    LODMaxRange;                            // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  UnknownData01[0x10];                    // 0x0088 (0x0010)
MISSED OFFSET
int32_t                                  LightMapResolution;                     // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                  LightMapCoordinateIndex;                // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  UnknownData02[0x58];                    // 0x00A0 (0x0058)
MISSED OFFSET
class URB_BodySetup*                          BodySetup;                              // 0x00F8 (0x0008)
[0x0000000004000001] (CPF_Edit | CPF_EditInline)
```

```
uint8_t                              UnknownDataO3[0x40];                    // 0x0100 (0x0040)
MISSED OFFSET
unsigned long                        UseSimpleLineCollision : 1;                 // 0x0140 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        UseSimpleBoxCollision : 1;                  // 0x0144 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        UseSimpleRigidBodyCollision : 1;            // 0x0148 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        UseFullPrecisionUVs : 1;                    // 0x014C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bUsedForInstancing : 1;                     // 0x0150 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                              ConsolePreallocateInstanceCount;            // 0x0154 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                        bUseMaximumStreamingTexelRatio : 1;         // 0x0158
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bPartitionForEdgeGeometry : 1;              // 0x015C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bCanBecomeDynamic : 1;                      // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bStripComplexCollisionForConsole : 1;       // 0x0168
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bPerLODStaticLightingForInstancing : 1;     // 0x016C
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
float                                StreamingDistanceMultiplier;                // 0x0170 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              UnknownData04[0x8C];                    // 0x0174 (0x008C)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticMesh");
}

return uClassPointer;
};

};

// Class Engine.Surface
// 0x0000 (0x0060 - 0x0060)
class USurface : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Surface");
}

return uClassPointer;
};

float GetSurfaceHeight();
float GetSurfaceWidth();
};

// Class Engine.MaterialInterface
// 0x0214 (0x0060 - 0x0274)
class UMaterialInterface : public USurface
{
public:
struct FRenderCommandFence_Mirror               ParentRefFence;                          // 0x0060
(0x0004) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FLightmassMaterialInterfaceSettings        LightmassSettings;                         // 0x0064
(0x001C) [0x0000000000000001] (CPF_Edit)
class FString                                PreviewMesh;                         // 0x0080 (0x0010)
[0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
struct FGuid                                LightingGuid;                         // 0x0090 (0x0010)
[0x0000000800000002] (CPF_Const)
unsigned long                               bHasQualitySwitch : 1;                     // 0x00A0 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                               bHasHandheldQualityConnection : 1;           // 0x00A0
(0x0004) [0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                               bStoredHandheldQuality : 1;                // 0x00A0 (0x0004)
[0x0000000000000002] [0x00000004] (CPF_Const)
unsigned long                               bAutoFlattenMobile : 1;                    // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                               bAutoFlattenMobileNormalTexture : 1;          // 0x00A0
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                               bMobileAllowFog : 1;                       // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                               bGenerateSubUV : 1;                        // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                               bUseMobileSpecular : 1;                    // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                               bUseMobileVertexSpecular : 1;               // 0x00A0 (0x0004)
[0x0000000020000000] [0x00000100]  CPF_Deprecated)
unsigned long                               bUseMobilePixelSpecular : 1;                // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                               bUseMobileBumpOffset : 1;                  // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                               bLockColorBlending : 1;                    // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                               bUseMobileUniformColorMultiply : 1;          // 0x00A0
(0x0004) [0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                               bUseMobileVertexColorMultiply : 1;           // 0x00A0
(0x0004) [0x0000000000000001] [0x00002000] (CPF_Edit)
```

```
unsigned long                    bUseMobileDetailNormal : 1;                // 0x00A0 (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                    bBaseTextureTransformed : 1;               // 0x00A0 (0x0004)
[0x0000000000000001] [0x00008000] (CPF_Edit)
unsigned long                    bEmissiveTextureTransformed : 1;           // 0x00A0
(0x0004) [0x0000000000000001] [0x00010000] (CPF_Edit)
unsigned long                    bNormalTextureTransformed : 1;             // 0x00A0 (0x0004)
[0x0000000000000001] [0x00020000] (CPF_Edit)
unsigned long                    bMaskTextureTransformed : 1;               // 0x00A0 (0x0004)
[0x0000000000000001] [0x00040000] (CPF_Edit)
unsigned long                    bDetailTextureTransformed : 1;             // 0x00A0 (0x0004)
[0x0000000000000001] [0x00080000] (CPF_Edit)
unsigned long                    bUseMobileWaveVertexMovement : 1;          // 0x00A0
(0x0004) [0x0000000000000001] [0x00100000] (CPF_Edit)
unsigned long                    bMobileEnableBounceLight : 1;              // 0x00A0 (0x0004)
[0x0000000000000001] [0x00200000] (CPF_Edit)
unsigned long                    bUseMobileLandscapeMonochromeLayerBlending : 1;//
0x00A0 (0x0004) [0x0000000000000001] [0x00400000] (CPF_Edit)
unsigned long                    bHandheldDisableDiffuse : 1;               // 0x00A0 (0x0004)
[0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                    bHandheldDisableDiffusePower : 1;          // 0x00A0
(0x0004) [0x0000000000000001] [0x01000000] (CPF_Edit)
unsigned long                    bHandheldDisableEmissive : 1;              // 0x00A0 (0x0004)
[0x0000000000000001] [0x02000000] (CPF_Edit)
unsigned long                    bHandheldDisableSpecular : 1;              // 0x00A0 (0x0004)
[0x0000000000000001] [0x04000000] (CPF_Edit)
unsigned long                    bHandheldDisableSpecualrPower : 1;         // 0x00A0
(0x0004) [0x0000000000000001] [0x08000000] (CPF_Edit)
unsigned long                    bHandheldDisableOpacity : 1;               // 0x00A0 (0x0004)
[0x0000000000000001] [0x10000000] (CPF_Edit)
unsigned long                    bHandheldDisableOpacityMasks : 1;          // 0x00A0
(0x0004) [0x0000000000000001] [0x20000000] (CPF_Edit)
unsigned long                    bHandheldDisableDistortion : 1;            // 0x00A0 (0x0004)
[0x0000000000000001] [0x40000000] (CPF_Edit)
unsigned long                    bHandheldDisableTransmissionMask : 1;      // 0x00A0
(0x0004) [0x0000000000000001] [0x80000000] (CPF_Edit)
unsigned long                    bHandheldDisableTransmissionColor : 1;     // 0x00A4
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bHandheldDisableNormal : 1;                // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bHandheldDisableCustomLighting : 1;        // 0x00A4
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    bHandheldDisableAnisotropicDirection : 1;  // 0x00A4
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bHandheldDisableWorldPositionOffset : 1;   // 0x00A4
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                    bHandheldDisableWorldDisplacement : 1;     // 0x00A4
(0x0004) [0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                    bHandheldDisableTessellationMultiplier : 1; // 0x00A4
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                    bHandheldDisableSubsurfaceInscatteringColor : 1;// 0x00A4
(0x0004) [0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                    bHandheldDisableSubsurfaceAbsorptionColor : 1; // 0x00A4
(0x0004) [0x0000000000000001] [0x00000100] (CPF_Edit)
```

```
unsigned long                    bHandheldDisableSubsurfaceScatteringRadius : 1;// 0x00A4
(0x0004) [0x0000000000000001] [0x00000200] (CPF_Edit)
struct FColor                    FlattenBackgroundColor;              // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture*                  MobileBaseTexture;                   // 0x00B0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture*                  FlattenedTexture;                    // 0x00B8 (0x0008)
[0x0000000020200000]  CPF_Deprecated)
uint8_t                          MobileBaseTextureTexCoordsSource;     // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileAmbientOcclusionSource;         // 0x00C1 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileSpecularMask;                   // 0x00C2 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileEmissiveColorSource;            // 0x00C3 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileEmissiveMaskSource;             // 0x00C4 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileEnvironmentMaskSource;          // 0x00C5 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileEnvironmentBlendMode;           // 0x00C6 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileRimLightingMaskSource;          // 0x00C7 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileMaskTextureTexCoordsSource;     // 0x00C8 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileAlphaValueSource;               // 0x00C9 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileDetailTextureTexCoordsSource;   // 0x00CA (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileTextureBlendFactorSource;       // 0x00CB (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          MobileColorMultiplySource;            // 0x00CC (0x0001)
[0x0000000000000001] (CPF_Edit)
class UTexture*                  MobileNormalTexture;                  // 0x00D0 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            SubUVFrameRate;                       // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          SubUVFrameCountAlongAxes;             // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            SubUVFrameSize;                       // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor              MobileSpecularColor;                  // 0x00E4 (0x0010)
[0x0000000000000001] (CPF_Edit)
float                            MobileSpecularPower;                  // 0x00F4 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture*                  MobileEmissiveTexture;                // 0x00F8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor              MobileEmissiveColor;                  // 0x0100 (0x0010)
[0x0000000000000001] (CPF_Edit)
class UTexture*                  MobileEnvironmentTexture;             // 0x0110 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            MobileEnvironmentAmount;              // 0x0118 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
struct FLinearColor                MobileEnvironmentColor;                // 0x011C (0x0010)
[0x0000000000000001] (CPF_Edit)
float                         MobileEnvironmentFresnelAmount;             // 0x012C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         MobileEnvironmentFresnelExponent;           // 0x0130 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         MobileRimLightingStrength;                // 0x0134 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         MobileRimLightingExponent;                // 0x0138 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor                MobileRimLightingColor;                // 0x013C (0x0010)
[0x0000000000000001] (CPF_Edit)
float                         MobileBumpOffsetReferencePlane;             // 0x014C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         MobileBumpOffsetHeightRatio;               // 0x0150 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture*                   MobileMaskTexture;                    // 0x0158 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                         MobileOpacityMultiplier;                 // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture*                   MobileDetailTexture;                  // 0x0168 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture*                   MobileDetailTexture2;                 // 0x0170 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture*                   MobileDetailTexture3;                 // 0x0178 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor                DefaultUniformColor;                  // 0x0180 (0x0010)
[0x0000000020000000]  CPF_Deprecated)
struct FLinearColor                MobileDefaultUniformColor;               // 0x0190 (0x0010)
[0x0000000000000001] (CPF_Edit)
float                         TransformCenterX;                     // 0x01A0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                         MobileTransformCenterX;                 // 0x01A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         TransformCenterY;                     // 0x01A8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                         MobileTransformCenterY;                 // 0x01AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         PannerSpeedX;                        // 0x01B0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                         MobilePannerSpeedX;                    // 0x01B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         PannerSpeedY;                        // 0x01B8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                         MobilePannerSpeedY;                    // 0x01BC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         RotateSpeed;                         // 0x01C0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                         MobileRotateSpeed;                    // 0x01C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         FixedScaleX;                         // 0x01C8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                         MobileFixedScaleX;                    // 0x01CC (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                                   FixedScaleY;                             // 0x01D0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                                   MobileFixedScaleY;                       // 0x01D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   SineScaleX;                              // 0x01D8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                                   MobileSineScaleX;                        // 0x01DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   SineScaleY;                              // 0x01E0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                                   MobileSineScaleY;                        // 0x01E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   SineScaleFrequencyMultipler;             // 0x01E8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                                   MobileSineScaleFrequencyMultipler;       // 0x01EC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FixedOffsetX;                            // 0x01F0 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                                   MobileFixedOffsetX;                      // 0x01F4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FixedOffsetY;                            // 0x01F8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
float                                   MobileFixedOffsetY;                      // 0x01FC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MobileTangentVertexFrequencyMultiplier;  // 0x0200 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MobileVerticalFrequencyMultiplier;       // 0x0204 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MobileMaxVertexMovementAmplitude;        // 0x0208 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MobileSwayFrequencyMultiplier;           // 0x020C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MobileSwayMaxAngle;                      // 0x0210 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                          MobileDirectionalLightDirection;         // 0x0214 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                   MobileDirectionalLightBrightness;        // 0x0220 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                           MobileDirectionalLightColor;             // 0x0224 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                          MobileBounceLightDirection;              // 0x0228 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                   MobileBounceLightBrightness;             // 0x0234 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                           MobileBounceLightColor;                  // 0x0238 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MobileSkyLightBrightness;                // 0x023C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                           MobileSkyLightColor;                     // 0x0240 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                            MobileLandscapeLayerNames[0x4];          // 0x0244
(0x0020) [0x0000000000000001] (CPF_Edit)
struct FColor                           MobileLandscapeMonochomeLayerColors[0x4];     // 0x0264
(0x0010) [0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialInterface");
}

return uClassPointer;
};

class UMaterialInstance* GetOrCreateInstance();
void SetForceMipLevelsToBeResident(unsigned long OverrideForceMiplevelsToBeResident,
unsigned long bForceMiplevelsToBeResidentValue, float ForceDuration, int32_t
CinematicTextureGroups);
bool GetMobileVectorParameterValue(struct FName ParameterName, struct FLinearColor&
OutValue);
bool GetMobileTextureParameterValue(struct FName ParameterName, class UTexture*&
OutValue);
bool GetMobileScalarParameterValue(struct FName ParameterName, float& OutValue);
bool GetGroupName(struct FName ParameterName, struct FName& GroupName);
bool GetLinearColorCurveParameterValue(struct FName ParameterName, struct
FInterpCurveLinearColor& OutValue);
bool GetLinearColorParameterValue(struct FName ParameterName, struct FLinearColor&
OutValue);
bool GetVectorCurveParameterValue(struct FName ParameterName, struct FInterpCurveVector&
OutValue);
bool GetVectorParameterValue(struct FName ParameterName, struct FLinearColor& OutValue);
bool GetTextureParameterValue(struct FName ParameterName, class UTexture*& OutValue);
bool GetScalarCurveParameterValue(struct FName ParameterName, struct FInterpCurveFloat&
OutValue);
bool GetScalarParameterValue(struct FName ParameterName, float& OutValue);
bool GetFontParameterValue(struct FName ParameterName, class UFont*& OutFontValue,
int32_t& OutFontPage);
bool GetParameterDesc(struct FName ParameterName, class FString& OutDesc);
class UPhysicalMaterial* GetPhysicalMaterial();
class UMaterial* GetMaterial();
};

// Class Engine.RB_BodySetup
// 0x0064 (0x00C0 - 0x0124)
class URB_BodySetup : public UKMeshProps
{
public:
uint8_t                          SleepFamily;                        // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                     BoneName;                           // 0x00C4 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
unsigned long                    bFixed : 1;                         // 0x00CC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bNoCollision : 1;                   // 0x00CC (0x0004)
```

```
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bBlockZeroExtent : 1;              // 0x00CC (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    bBlockNonZeroExtent : 1;            // 0x00CC (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bEnableContinuousCollisionDetection : 1;    // 0x00CC
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                    bAlwaysFullAnimWeight : 1;           // 0x00CC (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                    bConsiderForBounds : 1;             // 0x00CC (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
class UPhysicalMaterial*              PhysMaterial;                 // 0x00D0 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                        MassScale;                  // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FPointer>               CollisionGeom;                // 0x00E0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FVector>                CollisionGeomScale3D;            // 0x00F0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FVector>                PreCachedPhysScale;             // 0x0100 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<struct FKCachedConvexData>          PreCachedPhysData;                // 0x0110
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                         PreCachedPhysDataVersion;           // 0x0120 (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_BodySetup");
}

return uClassPointer;
};

};

// Class Engine.FracturedStaticMesh
// 0x00F0 (0x0200 - 0x02F0)
class UFracturedStaticMesh : public UStaticMesh
{
public:
class UStaticMesh*                 SourceCoreMesh;                // 0x0200 (0x0008)
[0x0000000800020001] (CPF_Edit | CPF_EditConst)
float                        CoreMeshScale;               // 0x0208 (0x0004)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
uint8_t                        UnknownData00[0x30];             // 0x020C (0x0030)
MISSED OFFSET
unsigned long                    bSliceUsingCoreCollision : 1;          // 0x023C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
class UParticleSystem*                    FragmentDestroyEffect;                    // 0x0240 (0x0008)
[0x0000000000000000]
TArray<class UParticleSystem*>                 FragmentDestroyEffects;                    // 0x0248
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                              FragmentDestroyEffectScale;                // 0x0258 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              FragmentHealthScale;                    // 0x025C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              FragmentMinHealth;                      // 0x0260 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              FragmentMaxHealth;                      // 0x0264 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bUniformFragmentHealth : 1;                // 0x0268 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                              ChunkLinVel;                         // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ChunkAngVel;                         // 0x0270 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ChunkLinHorizontalScale;                  // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ExplosionVelScale;                      // 0x0278 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bCompositeChunksExplodeOnImpact : 1;         // 0x027C
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bFixIsolatedChunks : 1;                  // 0x0280 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bAlwaysBreakOffIsolatedIslands : 1;          // 0x0284
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bSpawnPhysicsChunks : 1;                  // 0x0288 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                              ChanceOfPhysicsChunk;                    // 0x028C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ExplosionChanceOfPhysicsChunk;             // 0x0290 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              NormalPhysicsChunkScaleMin;               // 0x0294 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              NormalPhysicsChunkScaleMax;               // 0x0298 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ExplosionPhysicsChunkScaleMin;             // 0x029C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ExplosionPhysicsChunkScaleMax;             // 0x02A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              MinConnectionSupportArea;                 // 0x02A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*                   DynamicOutsideMaterial;                  // 0x02A8
(0x0008) [0x0000000000000001] (CPF_Edit)
class UMaterialInterface*                   LoseChunkOutsideMaterial;                 // 0x02B0
(0x0008) [0x0000000000000001] (CPF_Edit)
int32_t                              OutsideMaterialIndex;                    // 0x02B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                             UnknownData01[0x34];                    // 0x02BC (0x0034)
MISSED OFFSET

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FracturedStaticMesh");
}

return uClassPointer;
};

};

// Class Engine.ParticleSystem
// 0x0130 (0x0060 - 0x0190)
class UParticleSystem : public UObject
{
public:
uint8_t                                  SystemUpdateMode;                          // 0x0060 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  LODMethod;                                 // 0x0061 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  OcclusionBoundsMethod;                     // 0x0062 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                    UpdateTime_FPS;                            // 0x0064 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    UpdateTime_Delta;                          // 0x0068 (0x0004)
[0x0000000000000000]
float                                    WarmupTime;                                // 0x006C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    WarmupTickRate;                            // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class UParticleEmitter*>                 Emitters;                                  // 0x0078 (0x0010)
[0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
class UParticleSystemComponent*                 PreviewComponent;                          // 0x0088
(0x0008) [0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)
struct FRotator                               ThumbnailAngle;                            // 0x0090 (0x000C)
[0x0000000800000000]
float                                    ThumbnailDistance;                         // 0x009C (0x0004)
[0x0000000800000000]
float                                    ThumbnailWarmup;                           // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                                 bLit : 1;                                  // 0x00A4 (0x0004)
[0x0000000020000002] [0x00000001] (CPF_Const | CPF_Deprecated)
unsigned long                                 bOrientZAxisTowardCamera : 1;              // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                 bRegenerateLODDuplicate : 1;               // 0x00A4 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                                 bUseFixedRelativeBoundingBox : 1;          // 0x00A4
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                                 bShouldResetPeakCounts : 1;                // 0x00A4 (0x0004)
[0x0000000000000000] [0x00000010]
```

```
unsigned long                          bHasPhysics : 1;                    // 0x00A4 (0x0004)
[0x0000000000002000] [0x00000020] (CPF_Transient)
unsigned long                          bUseRealtimeThumbnail : 1;                 // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          ThumbnailImageOutOfDate : 1;               // 0x00A4 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                          bSkipSpawnCountCheck : 1;                  // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                          bUseDelayRange : 1;                  // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
class UInterpCurveEdSetup*                   CurveEdSetup;                        // 0x00A8 (0x0008)
[0x0000000000000008] (CPF_ExportObject)
float                             LODDistanceCheckTime;                   // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<float>                            LODDistances;                       // 0x00B8 (0x0010)
[0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
int32_t                              EditorLODSetting;                        // 0x00C8 (0x0004)
[0x0000000800000000]
TArray<struct FParticleSystemLOD>               LODSettings;                              // 0x00D0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FBox                            FixedRelativeBoundingBox;                // 0x00E0 (0x001C)
[0x0000000000000001] (CPF_Edit)
float                             SecondsBeforeInactive;                  // 0x00FC (0x0004)
[0x0000000000000001] (CPF_Edit)
class FString                           FloorMesh;                          // 0x0100 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
struct FVector                          FloorPosition;                         // 0x0110 (0x000C)
[0x0000000800000000]
struct FRotator                          FloorRotation;                         // 0x011C (0x000C)
[0x0000000800000000]
float                             FloorScale;                          // 0x0128 (0x0004)
[0x0000000800000000]
struct FVector                          FloorScale3D;                          // 0x012C (0x000C)
[0x0000000800000000]
struct FColor                           BackgroundColor;                        // 0x0138 (0x0004)
[0x0000000800000000]
class UTexture2D*                          ThumbnailImage;                        // 0x0140 (0x0008)
[0x0000000800000000]
float                             Delay;                           // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                             DelayLow;                          // 0x014C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                          MacroUVPosition;                        // 0x0150 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                             MacroUVRadius;                         // 0x015C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FBox                            CustomOcclusionBounds;                   // 0x0160 (0x001C)
[0x0000000000000001] (CPF_Edit)
TArray<struct FLODSoloTrack>                 SoloTracking;                          // 0x0180 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleSystem");
}

return uClassPointer;
};

float GetMaxLifespan(float InComponentDelay);
bool SetLODDistance(int32_t LODLevelIndex, float InDistance);
void SetCurrentLODMethod(uint8_t InMethod);
float GetLODDistance(int32_t LODLevelIndex);
int32_t GetLODLevelCount();
uint8_t GetCurrentLODMethod();
};

// Class Engine.Texture
// 0x00F0 (0x0060 - 0x0150)
class UTexture : public USurface
{
public:
unsigned long                    SRGB : 1;                          // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    RGBE : 1;                          // 0x0060 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                    bIsSourceArtUncompressed : 1;              // 0x0060 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                    CompressionNoAlpha : 1;                  // 0x0060 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    CompressionNone : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                    CompressionNoMipmaps : 1;                 // 0x0060 (0x0004)
[0x0000000020000000] [0x00000020]  CPF_Deprecated)
unsigned long                    CompressionFullDynamicRange : 1;            // 0x0060
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                    DeferCompression : 1;                   // 0x0060 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                    NeverStream : 1;                      // 0x0060 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                    bDitherMipMapAlpha : 1;                  // 0x0060 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                    bPreserveBorderR : 1;                   // 0x0060 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                    bPreserveBorderG : 1;                   // 0x0060 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                    bPreserveBorderB : 1;                   // 0x0060 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                    bPreserveBorderA : 1;                   // 0x0060 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                    bNoTiling : 1;                       // 0x0060 (0x0004)
[0x0000000000000002] [0x00004000] (CPF_Const)
unsigned long                    bForcePVRTC4 : 1;                     // 0x0060 (0x0004)
```

```
[0x0000000000000001] [0x00008000] (CPF_Edit)
unsigned long                    bAsyncResourceReleaseHasBeenStarted : 1;      // 0x0060
(0x0004) [0x0000000000002002] [0x00010000] (CPF_Const | CPF_Transient)
unsigned long                    bUseCinematicMipLevels : 1;               // 0x0060 (0x0004)
[0x0000000000002002] [0x00020000] (CPF_Const | CPF_Transient)
float                            UnpackMin[0x4];                           // 0x0064 (0x0010)
[0x0000000000000001] (CPF_Edit)
float                            UnpackMax[0x4];                           // 0x0074 (0x0010)
[0x0000000000000001] (CPF_Edit)
struct FUntypedBulkData_Mirror             SourceArt;                           // 0x0088 (0x0058)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                          CompressionSettings;                      // 0x00E0 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          Filter;                                   // 0x00E1 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          LODGroup;                                 // 0x00E2 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          CachedLODGroup;                           // 0x00E3 (0x0001)
[0x0000000000002000] (CPF_Transient)
uint8_t                          MipGenSettings;                           // 0x00E4 (0x0001)
[0x0000000000000001] (CPF_Edit)
int32_t                          LODBias;                                  // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          CachedCombinedLODBias;                    // 0x00EC (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                          NumCinematicMipLevels;                    // 0x00F0 (0x0004)
[0x0000000000000001] (CPF_Edit)
class FString                     SourceFilePath;                          // 0x00F8 (0x0010)
[0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                     SourceFileTimestamp;                     // 0x0108 (0x0010)
[0x0000000800420001] (CPF_Edit | CPF_EditConst | CPF_NeedCtorLink)
struct FPointer                   Resource;                                // 0x0118 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FGuid                      LightingGuid;                            // 0x0120 (0x0010)
[0x0000000800000002] (CPF_Const)
float                            AdjustBrightness;                         // 0x0130 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AdjustBrightnessCurve;                    // 0x0134 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AdjustVibrance;                           // 0x0138 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AdjustSaturation;                         // 0x013C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AdjustRGBCurve;                           // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AdjustHue;                                // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          InternalFormatLODBias;                    // 0x0148 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                          RequiredCreationFlags;                    // 0x014C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Texture");
}

return uClassPointer;
};

};

// Class Engine.Texture2D
// 0x0130 (0x0150 - 0x0280)
class UTexture2D : public UTexture
{
public:
struct FIndirectArray_Mirror                    Mips;                                   // 0x0150 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FIndirectArray_Mirror                    CachedPVRTCMips;                        // 0x0160
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FIndirectArray_Mirror                    CachedATITCMips;                        // 0x0170 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FIndirectArray_Mirror                    CachedETCMips;                          // 0x0180 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                                 CachedFlashMipsMaxResolution;                   // 0x0190 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror                  CachedFlashMips;                        // 0x0198
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                                 SizeX;                                  // 0x01F0 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                 SizeY;                                  // 0x01F4 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                 OriginalSizeX;                          // 0x01F8 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                 OriginalSizeY;                          // 0x01FC (0x0004)
[0x0000000000000002] (CPF_Const)
uint8_t                                 Format;                                 // 0x0200 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                                 AddressX;                               // 0x0201 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 AddressY;                               // 0x0202 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                                   bIsEditorOnly : 1;                      // 0x0204 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                                   bIsStreamable : 1;                      // 0x0204 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                                   bHasCancelationPending : 1;             // 0x0204 (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                                   bHasBeenLoadedFromPersistentArchive : 1;        // 0x0204
(0x0004) [0x0000000000002002] [0x00000008] (CPF_Const | CPF_Transient)
unsigned long                                   bForceMiplevelsToBeResident : 1;                // 0x0204 (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
```

```
unsigned long                              bGlobalForceMipLevelsToBeResident : 1;        // 0x0204
(0x0004) [0x0000000000000003] [0x00000020] (CPF_Edit | CPF_Const)
unsigned long                              bIsCompositingSource : 1;                 // 0x0204 (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                              bHasBeenPaintedInEditor : 1;              // 0x0204 (0x0004)
[0x0000000800000000] [0x00000080]
unsigned long                              bUseAlphaInThumbnail : 1;                 // 0x0204 (0x0004)
[0x0000000800000003] [0x00000100] (CPF_Edit | CPF_Const)
float                               ForceMipLevelsToBeResidentTimestamp;         // 0x0208
(0x0004) [0x0000000000002000] (CPF_Transient)
struct FName                               TextureFileCacheName;                    // 0x020C (0x0008)
[0x0000000000000000]
struct FGuid                             TextureFileCacheGuid;                    // 0x0214 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                              RequestedMips;                          // 0x0224 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                              ResidentMips;                          // 0x0228 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                              MipsToRemoveOnCompress;                  // 0x022C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FThreadSafeCounter                     PendingMipChangeRequestStatus;             // 0x0230
(0x0004) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<uint8_t>                            SystemMemoryData;                     // 0x0238 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class UTextureRenderTarget2D*                 RenderTarget2DRef;                      // 0x0248
(0x0008) [0x0000000000000000]
struct FTextureLinkedListMirror               StreamableTexturesLink;                  // 0x0250
(0x0018) [0x0000000001201002] (CPF_Const | CPF_Native)
int32_t                              StreamingIndex;                        // 0x0268 (0x0004)
[0x0000000000202002] (CPF_Const | CPF_Transient)
int32_t                              MipTailBaseIdx;                        // 0x026C (0x0004)
[0x0000000000000002] (CPF_Const)
struct FPointer                            ResourceMem;                          // 0x0270 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                              FirstResourceMemMip;                     // 0x0278 (0x0004)
[0x0000000000000002] (CPF_Const)
float                               Timer;                             // 0x027C (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Texture2D");
}

return uClassPointer;
};

static class UTexture2D* Create(int32_t InSizeX, int32_t InSizeY, uint8_t InFormat);
void SetForceMipLevelsToBeResident(float Seconds, int32_t CinematicTextureGroups);
```

```
};

// Class Engine.LightMapTexture2D
// 0x0008 (0x0280 - 0x0288)
class ULightMapTexture2D : public UTexture2D
{
public:
uint8_t                          UnknownData00[0x8];                  // 0x0280 (0x0008) MISSED
OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightMapTexture2D");
}

return uClassPointer;
};

};

// Class Engine.ShadowMapTexture2D
// 0x0004 (0x0280 - 0x0284)
class UShadowMapTexture2D : public UTexture2D
{
public:
int32_t                          ShadowmapFlags;                      // 0x0280 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ShadowMapTexture2D");
}

return uClassPointer;
};

};

// Class Engine.TickEventBase
// 0x000C (0x0060 - 0x006C)
class UTickEventBase : public UObject
{
public:
float                            DeltaSeconds;                        // 0x0060 (0x0004)
```

```
[0x0000000000000000]
float                         TimeDilation;                     // 0x0064 (0x0004)
[0x0000000000000000]
int32_t                        Count;                          // 0x0068 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TickEventBase");
}

return uClassPointer;
};

};

// Class Engine.FrameTick
// 0x0004 (0x006C - 0x0070)
class UFrameTick : public UTickEventBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FrameTick");
}

return uClassPointer;
};

};

// Class Engine.GameTick
// 0x0004 (0x006C - 0x0070)
class UGameTick : public UTickEventBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameTick");
}

return uClassPointer;
};

};

// Class Engine.TranslationContext
// 0x0010 (0x0060 - 0x0070)
class UTranslationContext : public UObject
{
public:
TArray<class UTranslatorTag*>                    TranslatorTags;                              // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TranslationContext");
}

return uClassPointer;
};

bool RegisterTranslatorTag(class UTranslatorTag* InTagHandler);
};

// Class Engine.TranslatorTag
// 0x0008 (0x0060 - 0x0068)
class UTranslatorTag : public UObject
{
public:
struct FName                                  Tag;                                      // 0x0060 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TranslatorTag");
}

return uClassPointer;
};
```

```
class FString Translate(class FString InArgument);
};

// Class Engine.StringsTag
// 0x0000 (0x0068 - 0x0068)
class UStringsTag : public UTranslatorTag
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StringsTag");
}

return uClassPointer;
};

class FString Translate(class FString InArgument);
};

// Class Engine.UIRoot
// 0x0010 (0x0060 - 0x0070)
class UUIRoot : public UObject
{
public:
TArray<class FString>                    BadCapsLocContexts;                    // 0x0060 (0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIRoot");
}

return uClassPointer;
};

static class FString SafeCaps(class FString StringToCap);
static class UOnlinePlayerInterfaceEx* GetOnlinePlayerInterfaceEx();
static class UOnlinePlayerInterface* GetOnlinePlayerInterface();
static class UOnlineGameInterface* GetOnlineGameInterface();
static class UUIDataStore* StaticResolveDataStore(struct FName DataStoreTag, class ULocalPlayer* InPlayerOwner);
static class UGameUISceneClient* GetSceneClient();
static class UUIInteraction* GetCurrentUIController();
static uint8_t GetInputPlatformType(class ULocalPlayer* OwningPlayer);
};

// Class Engine.Interaction
// 0x0060 (0x0070 - 0x00D0)
class UInteraction : public UUIRoot
{
public:
struct FScriptDelegate                __OnReceivedNativeInputKey__Delegate;        // 0x0070 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                __OnReceivedNativeInputAxis__Delegate;       // 0x0088 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                __OnReceivedNativeInputChar__Delegate;       // 0x00A0 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                __OnInitialize__Delegate;                    // 0x00B8 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Interaction");
}

return uClassPointer;
};

void NotifyPlayerRemoved(int32_t PlayerIndex, class ULocalPlayer* RemovedPlayer);
void NotifyPlayerAdded(int32_t PlayerIndex, class ULocalPlayer* AddedPlayer);
void NotifyGameSessionEnded();
void Initialized();
void OnInitialize();
void Init();
void eventPostRender(class UCanvas* Canvas);
void eventTick(float DeltaTime);
bool OnReceivedNativeInputChar(int32_t ControllerId, class FString Unicode);
bool OnReceivedNativeInputAxis(int32_t ControllerId, struct FName Key, float delta, float DeltaTime, unsigned long bGamepad);
bool OnReceivedNativeInputKey(int32_t ControllerId, struct FName Key, uint8_t EventType, float AmountDepressed, unsigned long bGamepad);
};

// Class Engine.UIInteraction
// 0x0140 (0x00D0 - 0x0210)
class UUIInteraction : public UInteraction
{
public:
struct FPointer                 VfTable_FExec;                                 // 0x00D0 (0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                 VfTable_FGlobalDataStoreClientManager;         // 0x00D8 (0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                          VfTable_FCallbackEventDevice;                // 0x00E0 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UUIManager*                            UIManager;                              // 0x00E8 (0x0008)
[0x0000000000000000]
class UClass*                            UIManagerClass;                             // 0x00F0 (0x0008)
[0x0000000000000000]
class UClass*                            SceneClientClass;                           // 0x00F8 (0x0008)
[0x0000000000000000]
class UGameUISceneClient*                         SceneClient;                             // 0x0100 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<struct FName>                         SupportedDoubleClickKeys;                 // 0x0108
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
class UDataStoreClient*                      DataStoreManager;                           // 0x0118 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                            bProcessInput : 1;                         // 0x0120 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
float                               UIJoystickDeadZone;                         // 0x0124 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
float                               UIAxisMultiplier;                         // 0x0128 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
float                               AxisRepeatDelay;                          // 0x012C (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
float                               MouseButtonRepeatDelay;                    // 0x0130 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
float                               DoubleClickTriggerSeconds;                 // 0x0134 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
int32_t                              DoubleClickPixelTolerance;                // 0x0138 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
struct FUIKeyRepeatData                      MouseButtonRepeatInfo;                    // 0x0140
(0x0010) [0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<struct FUIAxisEmulationDefinition>        ConfiguredAxisEmulationDefinitions;         //
0x0150 (0x0010) [0x0000000000404002] (CPF_Const | CPF_Config | CPF_NeedCtorLink)
uint8_t                              UnknownData00[0x50];                       // 0x0160 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.UIInteraction.AxisEmulationDefinitions
struct FUIAxisEmulationData                  AxisInputEmulation[0x4];                  // 0x01B0
(0x0060) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIInteraction");
}

return uClassPointer;
};

void NotifyGameSessionEnded();
static uint8_t eventGetNATType();
static int32_t GetConnectedGamepadCount(TArray<unsigned long>
ControllerConnectionStatusOverrides);
```

```
static bool IsGamepadConnected(int32_t ControllerId);
static int32_t GetNumGuestsLoggedIn();
static int32_t GetLoggedInPlayerCount(unsigned long bRequireOnlineLogin);
static bool eventIsLoggedIn(int32_t ControllerId, unsigned long bRequireOnlineLogin);
static bool eventHasLinkConnection();
uint8_t GetLowestLoginStatusOfControllers();
static uint8_t eventGetLoginStatus(int32_t ControllerId);
void NotifyPlayerRemoved(int32_t PlayerIndex, class ULocalPlayer* RemovedPlayer);
void NotifyPlayerAdded(int32_t PlayerIndex, class ULocalPlayer* AddedPlayer);
static class ULocalPlayer* GetLocalPlayer(int32_t PlayerIndex);
static class UDataStoreClient* GetDataStoreClient();
static int32_t GetPlayerControllerId(int32_t PlayerIndex);
static int32_t GetPlayerIndex(int32_t ControllerId);
static int32_t GetPlayerCount();
};

// Class Engine.UIManager
// 0x0000 (0x0060 - 0x0060)
class UUIManager : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIManager");
}

return uClassPointer;
};

int32_t FindLocalPlayerIndex(class UPlayer* P);
void NotifyPlayerRemoved(int32_t PlayerIndex, class ULocalPlayer* RemovedPlayer);
void NotifyPlayerAdded(int32_t PlayerIndex, class ULocalPlayer* AddedPlayer);
void eventPauseGame(unsigned long bDesiredPauseState, int32_t PlayerIndex);
bool CanUnpauseInternalUI();
static class UUIManager* GetUIManager();
};

// Class Engine.WaveFormBase
// 0x0008 (0x0060 - 0x0068)
class UWaveFormBase : public UObject
{
public:
class UForceFeedbackWaveform*                  TheWaveForm;                             // 0x0060
(0x0008) [0x0000000000000000]

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WaveFormBase");
}

return uClassPointer;
};

};

// Class Engine.World
// 0x0358 (0x0060 - 0x03B8)
class UWorld : public UObject
{
public:
uint8_t                              UnknownData00[0x358];                    // 0x0060 (0x0358)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.World");
}

return uClassPointer;
};

};

// Class Engine.EnvironmentVolume
// 0x0018 (0x02A4 - 0x02BC)
class AEnvironmentVolume : public AVolume
{
public:
struct FPointer                           VfTable_IInterface_NavMeshPathObstacle;      // 0x02A8
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                           VfTable_IInterface_NavMeshPathObject;        // 0x02B0
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
unsigned long                             bSplitNavMesh : 1;                       // 0x02B8 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.EnvironmentVolume");
}

return uClassPointer;
};

void SetSplitNavMesh(unsigned long bNewValue);
};

// Class Engine.TestSplittingVolume
// 0x000C (0x02A4 - 0x02B0)
class ATestSplittingVolume : public AVolume
{
public:
struct FPointer                          VfTable_IInterface_NavMeshPathObject;        // 0x02A8
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TestSplittingVolume");
}

return uClassPointer;
};

};

// Class Engine.AIController
// 0x002C (0x0474 - 0x04A0)
class AAIController : public AController
{
public:
unsigned long                            bAdjustFromWalls : 1;                    // 0x0478 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                            bReverseScriptedRoute : 1;                 // 0x0478 (0x0004)
[0x0000000000000000] [0x00000002]
float                                  Skill;                                 // 0x047C (0x0004)
[0x0000000000000000]
class AActor*                            ScriptedMoveTarget;                      // 0x0480 (0x0008)
[0x0000000000000000]
class ARoute*                            ScriptedRoute;                         // 0x0488 (0x0008)
[0x0000000000000000]
int32_t                                 ScriptedRouteIndex;                     // 0x0490 (0x0004)
[0x0000000000000000]
class AActor*                            ScriptedFocus;                          // 0x0498 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AIController");
}

return uClassPointer;
};

void OnAIMoveToActor(class USeqAct_AIMoveToActor* Action);
void eventSetTeam(int32_t inTeamIdx);
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void Reset();
void HandlePRITeamChanged(class APlayerReplicationInfo* PRI);
void HandlePRIRemoved(class APlayerReplicationInfo* PRI);
void HandlePRIAdded(class APlayerReplicationInfo* PRI);
void eventPostBeginPlay();
void eventPreBeginPlay();
};

// Class Engine.CrowdAgentBase
// 0x0008 (0x0268 - 0x0270)
class ACrowdAgentBase : public AActor
{
public:
struct FPointer                    VfTable_IInterface_NavigationHandle;        // 0x0268
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CrowdAgentBase");
}

return uClassPointer;
};

void eventNotifyPathChanged();
};

// Class Engine.CrowdPopulationManagerBase
// 0x0000 (0x0268 - 0x0268)
class ACrowdPopulationManagerBase : public AActor
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CrowdPopulationManagerBase");
}

return uClassPointer;
};

};

// Class Engine.PathTargetPoint
// 0x0000 (0x0270 - 0x0270)
class APathTargetPoint : public AKeypoint
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PathTargetPoint");
}

return uClassPointer;
};

bool ShouldBeHiddenBySHOW_NavigationNodes();
};

// Class Engine.NavMeshObstacle
// 0x000C (0x0268 - 0x0274)
class ANavMeshObstacle : public AActor
{
public:
struct FPointer                         VfTable_IInterface_NavMeshPathObstacle;       // 0x0268
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
unsigned long                           bEnabled : 1;                            // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bPreserveInternalGeo : 1;                  // 0x0270 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.NavMeshObstacle");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct ANavMeshObstacle_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ANavMeshObstacle_FCheckpointRecord& Record);
void SetEnabled(unsigned long bInEnabled);
void OnToggle(class USeqAct_Toggle* Action);
void PostBeginPlay();
void UnRegisterObstacle();
void RegisterObstacle();
bool eventGetObstacleBoudingShape(TArray<struct FVector>& Shape);
};

// Class Engine.PylonSeed
// 0x0008 (0x0268 - 0x0270)
class APylonSeed : public AActor
{
public:
struct FPointer                         VfTable_IInterface_NavMeshPathObject;        // 0x0268
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PylonSeed");
}

return uClassPointer;
};

};

// Class Engine.CoverGroupRenderingComponent
// 0x0000 (0x0258 - 0x0258)
class UCoverGroupRenderingComponent : public UPrimitiveComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CoverGroupRenderingComponent");
}
```

```
return uClassPointer;
};

};

// Class Engine.MeshComponent
// 0x0028 (0x0258 - 0x0280)
class UMeshComponent : public UPrimitiveComponent
{
public:
struct FPointer                              VfTable_IISetParameter;                    // 0x0258 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
TArray<class UMaterialInterface*>                 Materials;                                   // 0x0260 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<class UMaterialInterface*>                 InvisiTekMaterials;                          // 0x0270
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MeshComponent");
}

return uClassPointer;
};

void PrepMeshForThumbnail(struct FRotator InRotation, struct FVector InDrawScale3D);
void ForceMipLevelsToBeResident();
void SetActorParameter(struct FName Key, class AActor* Value);
void SetLinearColorParameter(struct FName Key, struct FLinearColor Value);
void SetVectorParameter(struct FName Key, struct FVector Value);
void SetFloatParameter(struct FName Key, float Value);
void SetNameParameter(struct FName Key, struct FName Value);
class UMaterialInstanceConstant* ConditionalCreateMIC(int32_t ElementIndex);
class UMaterialInstanceTimeVarying* CreateAndSetMaterialInstanceTimeVarying(int32_t
ElementIndex);
class UMaterialInstanceConstant* CreateAndSetMaterialInstanceConstant(int32_t
ElementIndex);
void PrestreamTextures(float Seconds, unsigned long bPrioritizeCharacterTextures, int32_t
CinematicTextureGroups);
int32_t GetNumElements();
void SetMaterial(int32_t ElementIndex, class UMaterialInterface* Material);
class UMaterialInterface* GetDefaultMaterial(int32_t ElementIndex);
class UMaterialInterface* GetMaterial(int32_t ElementIndex);
};

// Class Engine.StaticMeshComponent
// 0x0080 (0x0280 - 0x0300)
class UStaticMeshComponent : public UMeshComponent
```

```
{
public:
int32_t                         ForcedLodModel;                    // 0x0280 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         PreviousLODLevel;                  // 0x0284 (0x0004)
[0x0000000000000000]
class UStaticMesh*                     StaticMesh;                      // 0x0288 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FColor                      WireframeColor;                  // 0x0290 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bIgnoreInstanceForTextureStreaming : 1;       // 0x0294
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bOverrideLightMapResolution : 1;           // 0x0294 (0x0004)
[0x0000000020000002] [0x00000002] (CPF_Const | CPF_Deprecated)
unsigned long                       bOverrideLightMapRes : 1;              // 0x0294 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
int32_t                         OverriddenLightMapResolution;            // 0x0298 (0x0004)
[0x0000000020000002] (CPF_Const | CPF_Deprecated)
int32_t                         OverriddenLightMapRes;               // 0x029C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          OverriddenLODMaxRange;               // 0x02A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          StreamingDistanceMultiplier;            // 0x02A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         SubDivisionStepSize;                 // 0x02A8 (0x0004)
[0x0000000000000002] (CPF_Const)
unsigned long                       bUseSubDivisions : 1;                // 0x02AC (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                       bForceStaticDecals : 1;              // 0x02AC (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                       bCanHighlightSelectedSections : 1;          // 0x02AC
(0x0004) [0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                       bUseSimpleLightmapModifications : 1;         // 0x02AC
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
class UTexture*                      SimpleLightmapModificationTexture;          // 0x02B0
(0x0008) [0x0000000800000001] (CPF_Edit)
uint8_t                         SimpleLightmapModificationFunction;         // 0x02B8 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bNeverBecomeDynamic : 1;               // 0x02BC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<struct FGuid>                    IrrelevantLights;                 // 0x02C0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FStaticMeshComponentLODInfo>       LODData;                         // 0x02D0
(0x0010) [0x0000000080001002] (CPF_Const | CPF_Native)
int32_t                         VertexPositionVersionNumber;             // 0x02E0 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FLightmassPrimitiveSettings             LightmassSettings;                    // 0x02E4
(0x001C) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticMeshComponent");
}

return uClassPointer;
};

bool LineCheckUVs(struct FVector TraceEnd, struct FVector TraceStart, struct FVector2D&
OutUVs);
bool CanBecomeDynamic();
void SetForceStaticDecals(unsigned long bInForceStaticDecals);
void DisableRBCollisionWithSMC(class UPrimitiveComponent* OtherSMC, unsigned long
bDisabled);
bool SetStaticMesh(class UStaticMesh* NewMesh, unsigned long bForce);
};

// Class Engine.CoverMeshComponent
// 0x003C (0x0300 - 0x033C)
class UCoverMeshComponent : public UStaticMeshComponent
{
public:
TArray<struct FCoverMeshes>             Meshes;                         // 0x0300 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FVector                          LocationOffset;                 // 0x0310 (0x000C)
[0x0000000000000000]
class UStaticMesh*                      AutoAdjustOn;                   // 0x0320 (0x0008)
[0x0000000000000000]
class UStaticMesh*                      AutoAdjustOff;                  // 0x0328 (0x0008)
[0x0000000000000000]
class UStaticMesh*                      Disabled;                       // 0x0330 (0x0008)
[0x0000000000000000]
unsigned long                           bShowWhenNotSelected : 1;       // 0x0338 (0x0004)
[0x0000000800002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CoverMeshComponent");
}

return uClassPointer;
};

};

// Class Engine.NavMeshRenderingComponent
// 0x0000 (0x0258 - 0x0258)
class UNavMeshRenderingComponent : public UPrimitiveComponent
{
```

```
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshRenderingComponent");
}

return uClassPointer;
};

};

// Class Engine.PathRenderingComponent
// 0x0000 (0x0258 - 0x0258)
class UPathRenderingComponent : public UPrimitiveComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PathRenderingComponent");
}

return uClassPointer;
};

};

// Class Engine.RouteRenderingComponent
// 0x0000 (0x0258 - 0x0258)
class URouteRenderingComponent : public UPrimitiveComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RouteRenderingComponent");
}
return uClassPointer;
};

};

// Class Engine.AICommandBase
// 0x0000 (0x0060 - 0x0060)
class UAICommandBase : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AICommandBase");
}

return uClassPointer;
};

static int32_t eventGetUtility(class AAIController* InAI);
};

// Class Engine.AutoNavMeshPathObstacleUnregister
// 0x0010 (0x0060 - 0x0070)
class UAutoNavMeshPathObstacleUnregister : public UObject
{
public:
class UInterface_NavMeshPathObstacle*          PathObstacleRef_Object;                    // 0x0060 (0x0008) [0x0000000000001000] (CPF_Native)
class UInterface_NavMeshPathObstacle*          PathObstacleRef_Interface;                 // 0x0068 (0x0008) [0x0000000000001000] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AutoNavMeshPathObstacleUnregister");
}

return uClassPointer;
};

};

// Class Engine.Interface_NavMeshPathObject
// 0x0000 (0x0060 - 0x0060)
```

```
class UInterface_NavMeshPathObject : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Interface_NavMeshPathObject");
}

return uClassPointer;
};

};

// Class Engine.Interface_NavMeshPathSwitch
// 0x0000 (0x0060 - 0x0060)
class UInterface_NavMeshPathSwitch : public UInterface_NavMeshPathObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Interface_NavMeshPathSwitch");
}

return uClassPointer;
};

bool eventAIActivateSwitch(class AAIController* AI);
};

// Class Engine.Interface_NavMeshPathObstacle
// 0x0000 (0x0060 - 0x0060)
class UInterface_NavMeshPathObstacle : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.Interface_NavMeshPathObstacle");
}

return uClassPointer;
};

};

// Class Engine.Interface_PylonGeometryProvider
// 0x0000 (0x0060 - 0x0060)
class UInterface_PylonGeometryProvider : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Interface_PylonGeometryProvider");
}

return uClassPointer;
};

};

// Class Engine.Interface_RVO
// 0x0000 (0x0060 - 0x0060)
class UInterface_RVO : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Interface_RVO");
}

return uClassPointer;
};

};

// Class Engine.NavigationHandle
// 0x0148 (0x0060 - 0x01A8)
class UNavigationHandle : public UObject
{
```

```
public:
class APylon*                        AnchorPylon;                       // 0x0060 (0x0008)
[0x0000000000000000]
struct FPointer                      AnchorPoly;                        // 0x0068 (0x0008)
[0x0000000000001000] (CPF_Native)
struct FPathStore                      PathCache;                       // 0x0070 (0x0010)
[0x0000000000000000]
struct FPointer                      BestUnfinishedPathPoint;                 // 0x0080 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)
struct FPointer                      CurrentEdge;                       // 0x0088 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                      SubGoal_DestPoly;                    // 0x0090 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FBasedPosition                      FinalDestination;                  // 0x0098 (0x0038)
[0x0000000000000000]
unsigned long                        bSkipRouteCacheUpdates : 1;               // 0x00D0 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                        bUseORforEvaluateGoal : 1;               // 0x00D0 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                        bDebugConstraintsAndGoalEvals : 1;          // 0x00D0
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                        bUltraVerbosePathDebugging : 1;           // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                        bVisualPathDebugging : 1;                // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                        bDebug_Breadcrumbs : 1;                 // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
class UNavMeshPathConstraint*                PathConstraintList;                   // 0x00D8
(0x0008) [0x0000000000000000]
class UNavMeshPathGoalEvaluator*              PathGoalList;                        // 0x00E0
(0x0008) [0x0000000000000000]
struct FNavMeshPathParams                CachedPathParams;                     // 0x00E8
(0x0038) [0x0000000000000000]
uint8_t                            LastPathError;                     // 0x0120 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                            LastPathFailTime;                    // 0x0124 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                       Breadcrumbs[0xA];                    // 0x0128 (0x0078)
[0x0000000000000000]
int32_t                            BreadCrumbMostRecentIdx;                 // 0x01A0 (0x0004)
[0x0000000000000000]
float                            BreadCrumbDistanceInterval;              // 0x01A4 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavigationHandle");
}
return uClassPointer;
};

static bool GetClosestPointOnMesh(float SearchRadius, struct FVector& SearchPoint);
void DrawBreadCrumbs(unsigned long bPersistent);
static bool StaticGetValidatedAnchorPosition(struct FVector StartCheckBaseLocation, struct
FVector Extent, struct FVector& out_NewAnchorLoc);
bool GetValidatedAnchorPosition(struct FVector OverrideStartLoc, struct FVector&
out_NewAnchorLoc);
static bool GetAllCoverSlotsInRadius(struct FVector FromLoc, float Radius, TArray<struct
FCoverInfo>& out_CoverList);
bool PopulatePathfindingParamCache();
static struct FVector MoveToDesiredHeightAboveMesh(struct FVector Point, float Height);
void CopyMovePointsFromPathCache(struct FVector FinalDest, TArray<struct FVector>&
out_MovePoints);
float CalculatePathDistance(struct FVector FinalDest);
struct FVector GetFirstMoveLocation();
bool IsAnchorInescapable();
void LimitPathCacheDistance(float MaxDist);
static void GetValidPositionsForBox(struct FVector pos, float Radius, struct FVector Extent,
unsigned long bMustBeReachableFromStartPos, int32_t MaxPositions, float MinRadius, struct
FVector ValidBoxAroundStartPos, TArray<struct FVector>& out_ValidPositions);
static void GetAllPolyCentersWithinBounds(struct FVector pos, struct FVector Extent,
TArray<struct FVector>& out_PolyCtrs);
uint8_t GetCurrentEdgeType();
void ClearCurrentEdge();
class FString GetCurrentEdgeDebugText();
void PrintPathCacheDebugText();
void DrawPathCache(struct FVector DrawOffset, unsigned long bPersistent, struct FColor
DrawColor);
bool ActorReachable(class AActor* A);
bool PointReachable(struct FVector Point, struct FVector OverrideStartPoint, unsigned long
bAllowHitsInEndCollisionBox);
bool PointCheck(struct FVector Pt, struct FVector Extent);
bool LineCheck(struct FVector Start, struct FVector End, struct FVector Extent, struct FVector&
out_HitLocation, struct FVector& out_HitNormal);
static bool ObstaclePointCheck(struct FVector Pt, struct FVector Extent);
static bool ObstacleLineCheck(struct FVector Start, struct FVector End, struct FVector Extent,
unsigned long bIgnoreNormalMesh, struct FVector& out_HitLoc, struct FVector& out_HitNorm);
bool SuggestMovePreparation(class AController* C, struct FVector& MovePt);
bool FindPath(class AActor*& out_DestActor, int32_t& out_DestItem);
bool ComputeValidFinalDestination(struct FVector& out_ComputedPosition);
bool SetFinalDestination(struct FVector FinalDest);
bool GetNextMoveLocation(float ArrivalDistance, struct FVector& out_MoveDest);
static class APylon* GetPylonFromPos(struct FVector Position);
bool FindPylon();
struct FVector GetBestUnfinishedPathPoint();
bool PathCache_RemoveIndex(int32_t InIdx, int32_t Count);
struct FVector PathCache_GetGoalPoint();
bool PathCache_Empty();
int32_t GetPathCacheLength();
class UNavMeshPathGoalEvaluator* CreatePathGoalEvaluator(class UClass* GoalEvalClass);
class UNavMeshPathConstraint* CreatePathConstraint(class UClass* ConstraintClass);
bool DoesPylonAHaveAPathToPylonB(class APylon* A, class APylon* B);
```

```
class APylon* BuildFromPylonAToPylonB(class APylon* A, class APylon* B);
void AddGoalEvaluator(class UNavMeshPathGoalEvaluator* Evaluator);
void AddPathConstraint(class UNavMeshPathConstraint* Constraint);
void ClearConstraints();
bool GetNextBreadCrumb(struct FVector& out_BreadCrumbLoc);
void UpdateBreadCrumbs(struct FVector InLocation);
void CopyPathStoreToPathCache(struct FPathStore& InStore);
};

// Class Engine.NavMeshGoal_Filter
// 0x000C (0x0060 - 0x006C)
class UNavMeshGoal_Filter : public UObject
{
public:
unsigned long                      bShowDebug : 1;                     // 0x0060 (0x0004)
[0x0000000000000000] [0x00000001]
int32_t                        NumNodesThrownOut;                      // 0x0064 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                        NumNodesProcessed;                      // 0x0068 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_Filter");
}

return uClassPointer;
};

class FString eventGetDumpString();
};

// Class Engine.NavMeshGoalFilter_MinPathDistance
// 0x0008 (0x006C - 0x0074)
class UNavMeshGoalFilter_MinPathDistance : public UNavMeshGoal_Filter
{
public:
int32_t                        MinDistancePathShouldBe;                // 0x0070 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoalFilter_MinPathDistance");
}
```

```
return uClassPointer;
};

static bool MustBeLongerPathThan(class UNavMeshGoal_GenericFilterContainer*
FilterContainer, int32_t InMinDistancePathShouldBe);
};

// Class Engine.NavMeshGoalFilter_NotNearOtherAI
// 0x0008 (0x006C - 0x0074)
class UNavMeshGoalFilter_NotNearOtherAI : public UNavMeshGoal_Filter
{
public:
float                                   DistanceToCheck;                        // 0x0070 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoalFilter_NotNearOtherAI");
}

return uClassPointer;
};

static bool NotNearOtherAI(class UNavMeshGoal_GenericFilterContainer* FilterContainer, float
InDistanceToCheck);
};

// Class Engine.NavMeshGoalFilter_OutOfViewFrom
// 0x0018 (0x006C - 0x0084)
class UNavMeshGoalFilter_OutOfViewFrom : public UNavMeshGoal_Filter
{
public:
struct FPointer                         GoalPoly;                               // 0x0070 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)
struct FVector                          OutOfViewLocation;                      // 0x0078 (0x000C)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoalFilter_OutOfViewFrom");
}

return uClassPointer;
```

```
};

static bool MustBeHiddenFromThisPoint(class UNavMeshGoal_GenericFilterContainer*
FilterContainer, struct FVector InOutOfViewLocation);
};

// Class Engine.NavMeshGoalFilter_OutSideOfDotProductWedge
// 0x0020 (0x006C - 0x008C)
class UNavMeshGoalFilter_OutSideOfDotProductWedge : public UNavMeshGoal_Filter
{
public:
struct FVector                    Location;                    // 0x0070 (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FVector                    Rotation;                    // 0x007C (0x000C)
[0x0000000000002000] (CPF_Transient)
float                         Epsilon;                     // 0x0088 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.NavMeshGoalFilter_OutSideOfDotProductWedge");
}

return uClassPointer;
};

static bool OutsideOfDotProductWedge(class UNavMeshGoal_GenericFilterContainer*
FilterContainer, struct FVector InLocation, struct FRotator InRotation, float InEpsilon);
};

// Class Engine.NavMeshGoalFilter_PolyEncompassesAI
// 0x0010 (0x006C - 0x007C)
class UNavMeshGoalFilter_PolyEncompassesAI : public UNavMeshGoal_Filter
{
public:
struct FVector                    OverrideExtentToCheck;              // 0x0070 (0x000C)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoalFilter_PolyEncompassesAI");
}
```

```
return uClassPointer;
};

static bool MakeSureAIFits(class UNavMeshGoal_GenericFilterContainer* FilterContainer, struct
FVector InOverrideExtentToCheck);
};

// Class Engine.NavMeshPathConstraint
// 0x0018 (0x0060 - 0x0078)
class UNavMeshPathConstraint : public UObject
{
public:
class UNavMeshPathConstraint*                  NextConstraint;                          // 0x0060
(0x0008) [0x0000000000000000]
int32_t                                        NumNodesProcessed;                       // 0x0068 (0x0004)
[0x0000000000000000]
int32_t                                        NumThrownOutNodes;                       // 0x006C (0x0004)
[0x0000000000000000]
float                                          AddedDirectCost;                         // 0x0070 (0x0004)
[0x0000000000000000]
float                                          AddedHeuristicCost;                      // 0x0074 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPathConstraint");
}

return uClassPointer;
};

class FString eventGetDumpString();
void eventRecycle();
};

// Class Engine.NavMeshPath_AlongLine
// 0x000C (0x0078 - 0x0084)
class UNavMeshPath_AlongLine : public UNavMeshPathConstraint
{
public:
struct FVector                                 Direction;                               // 0x0078 (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPath_AlongLine");
}

return uClassPointer;
};

void Recycle();
static bool AlongLine(class UNavigationHandle* NavHandle, struct FVector Dir);
};

// Class Engine.NavMeshPath_EnforceTwoWayEdges
// 0x0000 (0x0078 - 0x0078)
class UNavMeshPath_EnforceTwoWayEdges : public UNavMeshPathConstraint
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPath_EnforceTwoWayEdges");
}

return uClassPointer;
};

static bool EnforceTwoWayEdges(class UNavigationHandle* NavHandle);
};

// Class Engine.NavMeshPath_MinDistBetweenSpecsOfType
// 0x0018 (0x0078 - 0x0090)
class UNavMeshPath_MinDistBetweenSpecsOfType : public UNavMeshPathConstraint
{
public:
float                          MinDistBetweenEdgeTypes;                 // 0x0078 (0x0004)
[0x0000000000000000]
struct FVector                      InitLocation;                          // 0x007C (0x000C)
[0x0000000000000000]
uint8_t                          EdgeType;                            // 0x0088 (0x0001)
[0x0000000000000000]
float                          Penalty;                             // 0x008C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class
Engine.NavMeshPath_MinDistBetweenSpecsOfType");
}

return uClassPointer;
};

void Recycle();
static bool EnforceMinDist(class UNavigationHandle* NavHandle, float InMinDist, uint8_t
InEdgeType, struct FVector LastLocation, float InPenalty);
};

// Class Engine.NavMeshPath_SameCoverLink
// 0x0008 (0x0078 - 0x0080)
class UNavMeshPath_SameCoverLink : public UNavMeshPathConstraint
{
public:
class ACoverLink*                    TestLink;                         // 0x0078 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPath_SameCoverLink");
}

return uClassPointer;
};

void Recycle();
static void SameCoverLink(class UNavigationHandle* NavHandle, class ACoverLink* InLink);
};

// Class Engine.NavMeshPath_Toward
// 0x001C (0x0078 - 0x0094)
class UNavMeshPath_Toward : public UNavMeshPathConstraint
{
public:
unsigned long                        bBiasAgainstHighLevelPath : 1;         // 0x0078 (0x0004)
[0x0000000000000000] [0x00000001]
float                             OutOfHighLevelPathBias;              // 0x007C (0x0004)
[0x0000000000000000]
class AActor*                        GoalActor;                         // 0x0080 (0x0008)
[0x0000000000000000]
struct FVector                       GoalPoint;                         // 0x0088 (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPath_Toward");
}

return uClassPointer;
};

void Recycle();
static bool eventTowardPoint(class UNavigationHandle* NavHandle, struct FVector Point);
static bool TowardGoal(class UNavigationHandle* NavHandle, class AActor* Goal);
};

// Class Engine.NavMeshPath_WithinDistanceEnvelope
// 0x001C (0x0078 - 0x0094)
class UNavMeshPath_WithinDistanceEnvelope : public UNavMeshPathConstraint
{
public:
float                          MaxDistance;                      // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MinDistance;                      // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                  bSoft : 1;                        // 0x0080 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                  bOnlyThrowOutNodesThatLeaveEnvelope : 1;    // 0x0080
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
float                          SoftStartPenalty;                 // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                 EnvelopeTestPoint;                // 0x0088 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPath_WithinDistanceEnvelope");
}

return uClassPointer;
};

void Recycle();
static bool StayWithinEnvelopeToLoc(class UNavigationHandle* NavHandle, struct FVector
InEnvelopeTestPoint, float InMaxDistance, float InMinDistance, unsigned long bInSoft, float
InSoftStartPenalty, unsigned long bOnlyTossOutSpecsThatLeave);
};

// Class Engine.NavMeshPath_WithinTraversalDist
// 0x000C (0x0078 - 0x0084)
```

```
class UNavMeshPath_WithinTraversalDist : public UNavMeshPathConstraint
{
public:
float                                   MaxTraversalDist;                          // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bSoft : 1;                                 // 0x007C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                   SoftStartPenalty;                         // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPath_WithinTraversalDist");
}

return uClassPointer;
};

void Recycle();
static bool DontExceedMaxDist(class UNavigationHandle* NavHandle, float InMaxTraversalDist,
unsigned long bInSoft);
};

// Class Engine.NavMeshPathGoalEvaluator
// 0x001C (0x0060 - 0x007C)
class UNavMeshPathGoalEvaluator : public UObject
{
public:
class UNavMeshPathGoalEvaluator*                NextEvaluator;                            // 0x0060
(0x0008) [0x0000000000002000] (CPF_Transient)
int32_t                                 MaxPathVisits;                             // 0x0068 (0x0004)
[0x0000000000000000]
unsigned long                           bAlwaysCallEvaluateGoal : 1;                // 0x006C (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                           bDoPartialAStar : 1;                      // 0x006C (0x0004)
[0x0000000000000000] [0x00000002]
int32_t                                 NumNodesThrownOut;                        // 0x0070 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                 NumNodesProcessed;                        // 0x0074 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                 MaxOpenListSize;                            // 0x0078 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshPathGoalEvaluator");
}

return uClassPointer;
};

class FString eventGetDumpString();
void eventRecycle();
};

// Class Engine.NavMeshGoal_At
// 0x002C (0x007C - 0x00A8)
class UNavMeshGoal_At : public UNavMeshPathGoalEvaluator
{
public:
struct FVector                          Goal;                                  // 0x0080 (0x000C)
[0x0000000000000000]
float                                   GoalDist;                              // 0x008C (0x0004)
[0x0000000000000000]
unsigned long                           bKeepPartial : 1;                      // 0x0090 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                           bWeightPartialByDist : 1;              // 0x0090 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                           bGoalInSamePolyAsAnchor : 1;           // 0x0090 (0x0004)
[0x0000000000000000] [0x00000004]
float                                   PartialDistSq;                         // 0x0094 (0x0004)
[0x0000000000000000]
struct FPointer                         GoalPoly;                              // 0x0098 (0x0008)
[0x0000000000001000] (CPF_Native)
struct FPointer                         PartialGoal;                           // 0x00A0 (0x0008)
[0x0000000000001000] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_At");
}

return uClassPointer;
};

void Recycle();
static bool eventAtLocation(class UNavigationHandle* NavHandle, struct FVector GoalLocation,
float Dist, unsigned long bReturnPartial, unsigned long bInWeightPartialByDist);
static bool AtActor(class UNavigationHandle* NavHandle, class AActor* GoalActor, float Dist,
unsigned long bReturnPartial, unsigned long bInWeightPartialByDist);
void RecycleNative();
};
```

```
// Class Engine.NavMeshGoal_ClosestActorInList
// 0x006C (0x007C - 0x00E8)
class UNavMeshGoal_ClosestActorInList : public UNavMeshPathGoalEvaluator
{
public:
TArray<struct FBiasedGoalActor>                GoalList;                              // 0x0080 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FMultiMap_Mirror                        PolyToGoalActorMap;                    // 0x0090 (0x0050)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                                CachedAnchorPoly;                      // 0x00E0 (0x0008)
[0x0000000000001000] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_ClosestActorInList");
}

return uClassPointer;
};

void RecycleInternal();
void eventRecycle();
static class UNavMeshGoal_ClosestActorInList* ClosestActorInList(class UNavigationHandle*
NavHandle, TArray<struct FBiasedGoalActor>& InGoalList);
};

// Class Engine.NavMeshGoal_GenericFilterContainer
// 0x0034 (0x007C - 0x00B0)
class UNavMeshGoal_GenericFilterContainer : public UNavMeshPathGoalEvaluator
{
public:
TArray<class UNavMeshGoal_Filter*>             GoalFilters;                           // 0x0080 (0x0010)
[0x0000000004402008] (CPF_ExportObject | CPF_Transient | CPF_NeedCtorLink | CPF_EditInline)
struct FPointer                                SuccessfulGoal;                        // 0x0090 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)
class UNavigationHandle*                       MyNavigationHandle;                    // 0x0098
(0x0008) [0x0000000000002000] (CPF_Transient)
TArray<struct FVector>                         SeedLocations;                         // 0x00A0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_GenericFilterContainer");
}
```

```
return uClassPointer;
};

void Recycle();
struct FVector GetGoalPoint();
class UNavMeshGoal_Filter* GetFilterOfType(class UClass* Filter_Class);
static class UNavMeshGoal_GenericFilterContainer*
CreateAndAddFilterToNavHandleFromSeedList(class UNavigationHandle* NavHandle, int32_t
InMaxPathVisits, TArray<struct FVector>& InSearchSeeds);
static class UNavMeshGoal_GenericFilterContainer* CreateAndAddFilterToNavHandle(class
UNavigationHandle* NavHandle, int32_t InMaxPathVisits);
};

// Class Engine.NavMeshGoal_Null
// 0x000C (0x007C - 0x0088)
class UNavMeshGoal_Null : public UNavMeshPathGoalEvaluator
{
public:
struct FPointer                    PartialGoal;                       // 0x0080 (0x0008)
[0x0000000000001000] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_Null");
}

return uClassPointer;
};

void Recycle();
void RecycleNative();
static bool GoUntilBust(class UNavigationHandle* NavHandle, int32_t InMaxPathVisits);
};

// Class Engine.NavMeshGoal_PolyEncompassesAI
// 0x0010 (0x007C - 0x008C)
class UNavMeshGoal_PolyEncompassesAI : public UNavMeshPathGoalEvaluator
{
public:
struct FVector                     OverrideExtentToCheck;             // 0x0080 (0x000C)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_PolyEncompassesAI");
}

return uClassPointer;
};

void Recycle();
static bool MakeSureAIFits(class UNavigationHandle* NavHandle, struct FVector
InOverrideExtentToCheck);
};

// Class Engine.NavMeshGoal_Random
// 0x0014 (0x007C - 0x0090)
class UNavMeshGoal_Random : public UNavMeshPathGoalEvaluator
{
public:
int32_t                              MinDist;                              // 0x0080 (0x0004)
[0x0000000000000000]
float                                BestRating;                           // 0x0084 (0x0004)
[0x0000000000000000]
struct FPointer                      PartialGoal;                          // 0x0088 (0x0008)
[0x0000000000001000] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_Random");
}

return uClassPointer;
};

void Recycle();
void RecycleNative();
static bool FindRandom(class UNavigationHandle* NavHandle, int32_t InMinDist, int32_t
InMaxPathVisits);
};

// Class Engine.NavMeshGoal_WithinDistanceEnvelope
// 0x001C (0x007C - 0x0098)
class UNavMeshGoal_WithinDistanceEnvelope : public UNavMeshPathGoalEvaluator
{
public:
float                                MaxDistance;                          // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                MinDistance;                          // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                MinTraversalDist;                     // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
struct FVector                    EnvelopeTestPoint;                // 0x008C (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshGoal_WithinDistanceEnvelope");
}

return uClassPointer;
};

void Recycle();
static bool GoalWithinEnvelopeToLoc(class UNavigationHandle* NavHandle, struct FVector
InEnvelopeTestPoint, float InMaxDistance, float InMinDistance, float InMinTraversalDist);
};

// Class Engine.PathConstraint
// 0x0010 (0x0060 - 0x0070)
class UPathConstraint : public UObject
{
public:
int32_t                           CacheIdx;                         // 0x0060 (0x0004)
[0x0000000000000002] (CPF_Const)
class UPathConstraint*                    NextConstraint;                   // 0x0068 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PathConstraint");
}

return uClassPointer;
};

class FString eventGetDumpString();
void eventRecycle();
};

// Class Engine.Path_AlongLine
// 0x000C (0x0070 - 0x007C)
class UPath_AlongLine : public UPathConstraint
{
public:
struct FVector                    Direction;                        // 0x0070 (0x000C)
```

```
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Path_AlongLine");
}

return uClassPointer;
};

void Recycle();
static bool AlongLine(class APawn* P, struct FVector Dir);
};

// Class Engine.Path_AvoidInEscapableNodes
// 0x0010 (0x0070 - 0x0080)
class UPath_AvoidInEscapableNodes : public UPathConstraint
{
public:
int32_t                        Radius;                        // 0x0070 (0x0004)
[0x0000000000000000]
int32_t                        Height;                        // 0x0074 (0x0004)
[0x0000000000000000]
int32_t                        MaxFallSpeed;                      // 0x0078 (0x0004)
[0x0000000000000000]
int32_t                        MoveFlags;                       // 0x007C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Path_AvoidInEscapableNodes");
}

return uClassPointer;
};

void Recycle();
static bool DontGetStuck(class APawn* P);
void CachePawnReacFlags(class APawn* P);
};

// Class Engine.Path_MinDistBetweenSpecsOfType
// 0x0018 (0x0070 - 0x0088)
class UPath_MinDistBetweenSpecsOfType : public UPathConstraint
```

```
{
public:
float                              MinDistBetweenSpecTypes;                  // 0x0070 (0x0004)
[0x0000000000000000]
struct FVector                        InitLocation;                          // 0x0074 (0x000C)
[0x0000000000000000]
class UClass*                         ReachSpecClass;                           // 0x0080 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Path_MinDistBetweenSpecsOfType");
}

return uClassPointer;
};

void Recycle();
static bool EnforceMinDist(class APawn* P, float InMinDist, class UClass* InSpecClass, struct
FVector LastLocation);
};

// Class Engine.Path_TowardGoal
// 0x0008 (0x0070 - 0x0078)
class UPath_TowardGoal : public UPathConstraint
{
public:
class AActor*                         GoalActor;                             // 0x0070 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Path_TowardGoal");
}

return uClassPointer;
};

void Recycle();
static bool TowardGoal(class APawn* P, class AActor* Goal);
};

// Class Engine.Path_TowardPoint
// 0x000C (0x0070 - 0x007C)
```

```
class UPath_TowardPoint : public UPathConstraint
{
public:
struct FVector                      GoalPoint;                         // 0x0070 (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Path_TowardPoint");
}

return uClassPointer;
};

void Recycle();
static bool TowardPoint(class APawn* P, struct FVector Point);
};

// Class Engine.Path_WithinDistanceEnvelope
// 0x001C (0x0070 - 0x008C)
class UPath_WithinDistanceEnvelope : public UPathConstraint
{
public:
float                               MaxDistance;                       // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               MinDistance;                       // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bSoft : 1;                         // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bOnlyThrowOutNodesThatLeaveEnvelope : 1;      // 0x0078
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
float                               SoftStartPenalty;                  // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                      EnvelopeTestPoint;                 // 0x0080 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Path_WithinDistanceEnvelope");
}

return uClassPointer;
};
```

```
void Recycle();
static bool StayWithinEnvelopeToLoc(class APawn* P, struct FVector InEnvelopeTestPoint, float
InMaxDistance, float InMinDistance, unsigned long bInSoft, float InSoftStartPenalty, unsigned
long bOnlyTossOutSpecsThatLeave);
};

// Class Engine.Path_WithinTraversalDist
// 0x000C (0x0070 - 0x007C)
class UPath_WithinTraversalDist : public UPathConstraint
{
public:
float                                   MaxTraversalDist;                          // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bSoft : 1;                                 // 0x0074 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                   SoftStartPenalty;                          // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Path_WithinTraversalDist");
}

return uClassPointer;
};

void Recycle();
static bool DontExceedMaxDist(class APawn* P, float InMaxTraversalDist, unsigned long
bInSoft);
};

// Class Engine.PathGoalEvaluator
// 0x0018 (0x0060 - 0x0078)
class UPathGoalEvaluator : public UObject
{
public:
class UPathGoalEvaluator*               NextEvaluator;                             // 0x0060 (0x0008)
[0x0000000000000000]
class ANavigationPoint*                 GeneratedGoal;                             // 0x0068 (0x0008)
[0x0000000000000000]
int32_t                                 MaxPathVisits;                             // 0x0070 (0x0004)
[0x0000000000000000]
int32_t                                 CacheIdx;                                  // 0x0074 (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PathGoalEvaluator");
}

return uClassPointer;
};

class FString eventGetDumpString();
void eventRecycle();
};

// Class Engine.Goal_AtActor
// 0x0010 (0x0078 - 0x0088)
class UGoal_AtActor : public UPathGoalEvaluator
{
public:
class AActor*                    GoalActor;                    // 0x0078 (0x0008)
[0x0000000000000000]
float                         GoalDist;                    // 0x0080 (0x0004)
[0x0000000000000000]
unsigned long                    bKeepPartial : 1;               // 0x0084 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Goal_AtActor");
}

return uClassPointer;
};

void Recycle();
static bool AtActor(class APawn* P, class AActor* Goal, float Dist, unsigned long bReturnPartial);
};

// Class Engine.Goal_Null
// 0x0000 (0x0078 - 0x0078)
class UGoal_Null : public UPathGoalEvaluator
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.Goal_Null");
}

return uClassPointer;
};

void Recycle();
static bool GoUntilBust(class APawn* P, int32_t InMaxPathVisits);
};

// Class Engine.SkeletalMeshActor
// 0x0064 (0x0268 - 0x02CC)
class ASkeletalMeshActor : public AActor
{
public:
unsigned long                          bDamageAppliesImpulse : 1;                   // 0x0268 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bShouldDoAnimNotifies : 1;                   // 0x0268 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bForceSaveInCheckpoint : 1;                  // 0x0268 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bCollideActors_OldValue : 1;                 // 0x0268 (0x0004)
[0x0000000020000000] [0x00000008]  CPF_Deprecated)
unsigned long                          bShouldShadowParentAllAttachedActors : 1;      // 0x0268
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
class USkeletalMeshComponent*                   SkeletalMeshComponent;                    // 0x0270
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class ULightEnvironmentComponent*                LightEnvironment;                        // 0x0278
(0x0008) [0x000000000408000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class UAudioComponent*                        FacialAudioComp;                          // 0x0280 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class USkeletalMesh*                        ReplicatedMesh;                           // 0x0288 (0x0008)
[0x0000000100002020] (CPF_Net | CPF_Transient)
class UMaterialInterface*                     ReplicatedMaterial0;                      // 0x0290 (0x0008)
[0x0000000100000020] (CPF_Net)
class UMaterialInterface*                     ReplicatedMaterial1;                      // 0x0298 (0x0008)
[0x0000000100000020] (CPF_Net)
TArray<struct FSkelMeshActorControlTarget>        ControlTargets;                           // 0x02A0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UInterpGroup*>                    InterpGroupList;                          // 0x02B0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FName                           SavedAnimSeqName;                         // 0x02C0 (0x0008)
[0x0000000000002000] (CPF_Transient)
float                                SavedCurrentTime;                         // 0x02C8 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshActor");
}

return uClassPointer;
};

bool eventCreateForceField(class UAnimNotify_ForceField* AnimNotifyData);
void SkelMeshActorOnParticleSystemFinished(class UParticleSystemComponent* PSC);
bool eventPlayParticleEffect(class UAnimNotify_PlayParticleEffect* AnimNotifyData);
void ApplyCheckpointRecord(struct ASkeletalMeshActor_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ASkeletalMeshActor_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
void DoKismetAttachment(class AActor* Attachment, class USeqAct_AttachToActor* Action);
void eventOnSetSkelControlTarget(class USeqAct_SetSkelControlTarget* Action);
void eventOnUpdatePhysBonesFromAnim(class USeqAct_UpdatePhysBonesFromAnim* Action);
void eventOnSetMesh(class USeqAct_SetMesh* Action);
bool IsActorPlayingFaceFXAnim();
class UFaceFXAsset* eventGetActorFaceFXAsset();
void OnPlayFaceFXAnim(class USeqAct_PlayFaceFXAnim* inAction);
class UAudioComponent* eventGetFaceFXAudioComponent();
void eventStopActorFaceFXAnim();
bool eventPlayActorFaceFXAnim(class UFaceFXAnimSet* AnimSet, class FString GroupName,
class FString SeqName, class USoundCue* SoundCueToPlay, class UAkEvent* AkEventToPlay);
void MAT_FinishAnimControl(class UInterpGroup* InInterpGroup);
void eventFinishAnimControl(class UInterpGroup* InInterpGroup);
void eventSetAnimPosition(struct FName SlotName, int32_t ChannelIndex, struct FName
InAnimSeqName, float InPosition, unsigned long bFireNotifies, unsigned long bLooping, unsigned
long bEnableRootMotion);
void MAT_BeginAnimControl(class UInterpGroup* InInterpGroup);
void eventBeginAnimControl(class UInterpGroup* InInterpGroup);
void OnSetMaterial(class USeqAct_SetMaterial* Action);
void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
void UpdateAnimSetList();
void eventDestroyed();
void eventPostBeginPlay();
};

// Class Engine.SkeletalMeshActorBasedOnExtremeContent
// 0x0024 (0x02CC - 0x02F0)
class ASkeletalMeshActorBasedOnExtremeContent : public ASkeletalMeshActor
{
public:
TArray<struct FSkelMaterialSetterDatum>          ExtremeContent;                          // 0x02D0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FSkelMaterialSetterDatum>          NonExtremeContent;                          // 0x02E0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshActorBasedOnExtremeContent");
}

return uClassPointer;
};

void SetMaterialBasedOnExtremeContent();
void eventPostBeginPlay();
};

// Class Engine.SkeletalMeshActorSpawnable
// 0x0004 (0x02CC - 0x02D0)
class ASkeletalMeshActorSpawnable : public ASkeletalMeshActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshActorSpawnable");
}

return uClassPointer;
};

};

// Class Engine.SkeletalMeshCinematicActor
// 0x0004 (0x02CC - 0x02D0)
class ASkeletalMeshCinematicActor : public ASkeletalMeshActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshCinematicActor");
}

return uClassPointer;
};

};
```

```
// Class Engine.SkeletalMeshActorMAT
// 0x0010 (0x02D0 - 0x02E0)
class ASkeletalMeshActorMAT : public ASkeletalMeshCinematicActor
{
public:
TArray<class UAnimNodeSlot*>                  SlotNodes;                           // 0x02D0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshActorMAT");
}

return uClassPointer;
};

void eventSetSkelControlScale(struct FName SkelControlName, float Scale);
void eventSetMorphWeight(struct FName MorphNodeName, float MorphWeight);
void eventFinishAnimControl(class UInterpGroup* InInterpGroup);
void MAT_SetAnimPosition(struct FName SlotName, int32_t ChannelIndex, struct FName
InAnimSeqName, float InPosition, unsigned long bFireNotifies, unsigned long bLooping, unsigned
long bEnableRootMotion);
void eventSetAnimPosition(struct FName SlotName, int32_t ChannelIndex, struct FName
InAnimSeqName, float InPosition, unsigned long bFireNotifies, unsigned long bLooping, unsigned
long bEnableRootMotion);
void ClearAnimNodes();
void CacheAnimNodes();
void eventPostInitAnimTree(class USkeletalMeshComponent* SkelComp);
void eventDestroyed();
void MAT_SetSkelControlStrength(struct FName SkelControlName, float ControlStrength);
void MAT_SetSkelControlScale(struct FName SkelControlName, float Scale);
void MAT_SetMorphWeight(struct FName MorphNodeName, float MorphWeight);
void MAT_SetAnimWeights(TArray<struct FAnimSlotInfo> SlotInfos);
};

// Class Engine.HeadTrackingComponent
// 0x00CB (0x009D - 0x0168)
class UHeadTrackingComponent : public UActorComponent
{
public:
TArray<struct FName>                  TrackControllerName;                  // 0x00A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                 LookAtActorRadius;                    // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                         bDisableBeyondLimit : 1;              // 0x00B4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                 MaxLookAtTime;                        // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                               MinLookAtTime;                          // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               MaxInterestTime;                        // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class UClass*>                       ActorClassesToLookAt;                   // 0x00C8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FName>                        TargetBoneNames;                        // 0x00D8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
uint8_t                             UnknownData00[0x50];                    // 0x00E8 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.HeadTrackingComponent.CurrentActorMap
class USkeletalMeshComponent*               SkeletalMeshComp;                       // 0x0138
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
TArray<class USkelControlLookAt*>           TrackControls;                          // 0x0140
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FVector                          RootMeshLocation;                       // 0x0150 (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FRotator                         RootMeshRotation;                       // 0x015C (0x000C)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HeadTrackingComponent");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm
// 0x0016 (0x0060 - 0x0076)
class UAnimationCompressionAlgorithm : public UObject
{
public:
class FString                           Description;                            // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                           bNeedsSkeleton : 1;                     // 0x0070 (0x0004)
[0x0000000000000000] [0x00000001]
uint8_t                             TranslationCompressionFormat;               // 0x0074 (0x0001)
[0x0000000000000000]
uint8_t                             RotationCompressionFormat;                  // 0x0075 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimationCompressionAlgorithm");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_Automatic
// 0x000A (0x0076 - 0x0080)
class UAnimationCompressionAlgorithm_Automatic : public UAnimationCompressionAlgorithm
{
public:
float                                   MaxEndEffectorError;                        // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bTryFixedBitwiseCompression : 1;            // 0x007C
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bTryPerTrackBitwiseCompression : 1;         // 0x007C
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                           bTryLinearKeyRemovalCompression : 1;        // 0x007C
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           bTryIntervalKeyRemoval : 1;                 // 0x007C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           bRunCurrentDefaultCompressor : 1;           // 0x007C
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                           bAutoReplaceIfExistingErrorTooGreat : 1;    // 0x007C
(0x0004) [0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                           bRaiseMaxErrorToExisting : 1;               // 0x007C (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_Automatic");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_BitwiseCompressOnly
// 0x0002 (0x0076 - 0x0078)
class UAnimationCompressionAlgorithm_BitwiseCompressOnly : public
UAnimationCompressionAlgorithm
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_BitwiseCompressOnly");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_LeastDestructive
// 0x0002 (0x0076 - 0x0078)
class UAnimationCompressionAlgorithm_LeastDestructive : public
UAnimationCompressionAlgorithm
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_LeastDestructive");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_RemoveEverySecondKey
// 0x000A (0x0076 - 0x0080)
class UAnimationCompressionAlgorithm_RemoveEverySecondKey : public
UAnimationCompressionAlgorithm
{
public:
int32_t                              MinKeys;                             // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                        bStartAtSecondKey : 1;               // 0x007C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_RemoveEverySecondKey");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_RemoveLinearKeys
// 0x001E (0x0076 - 0x0094)
class UAnimationCompressionAlgorithm_RemoveLinearKeys : public
UAnimationCompressionAlgorithm
{
public:
float                            MaxPosDiff;                     // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            MaxAngleDiff;                   // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            MaxEffectorDiff;                // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            MinEffectorDiff;                // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            EffectorDiffSocket;             // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            ParentKeyScale;                 // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bRetarget : 1;                  // 0x0090 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bActuallyFilterLinearKeys : 1;          // 0x0090 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_RemoveLinearKeys");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_PerTrackCompression
// 0x0064 (0x0094 - 0x00F8)
class UAnimationCompressionAlgorithm_PerTrackCompression : public
```

```
UAnimationCompressionAlgorithm_RemoveLinearKeys
{
public:
float                          MaxZeroingThreshold;                    // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MaxPosDiffBitwise;                      // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MaxAngleDiffBitwise;                    // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<uint8_t>                     AllowedRotationFormats;                  // 0x00A8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<uint8_t>                     AllowedTranslationFormats;               // 0x00B8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                       bResampleAnimation : 1;                  // 0x00C8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bUseAdaptiveError : 1;                   // 0x00C8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       bUseOverrideForEndEffectors : 1;         // 0x00C8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                       bUseAdaptiveError2 : 1;                  // 0x00C8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
float                          ResampledFramerate;                     // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         MinKeysForResampling;                   // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         TrackHeightBias;                        // 0x00D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          ParentingDivisor;                       // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          ParentingDivisorExponent;               // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          RotationErrorSourceRatio;               // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          TranslationErrorSourceRatio;            // 0x00E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MaxErrorPerTrackRatio;                  // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          PerturbationProbeSize;                  // 0x00EC (0x0004)
[0x0000000000000000]
struct FPointer                     PerReductionCachedData;                  // 0x00F0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_PerTrackCompression");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_RemoveTrivialKeys
// 0x000A (0x0076 - 0x0080)
class UAnimationCompressionAlgorithm_RemoveTrivialKeys : public
UAnimationCompressionAlgorithm
{
public:
float                                   MaxPosDiff;                          // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MaxAngleDiff;                        // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_RemoveTrivialKeys");
}

return uClassPointer;
};

};

// Class Engine.AnimationCompressionAlgorithm_RevertToRaw
// 0x0002 (0x0076 - 0x0078)
class UAnimationCompressionAlgorithm_RevertToRaw : public
UAnimationCompressionAlgorithm
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.AnimationCompressionAlgorithm_RevertToRaw");
}

return uClassPointer;
};

};
```

```
// Class Engine.AnimMetaData
// 0x0000 (0x0060 - 0x0060)
class UAnimMetaData : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimMetaData");
}

return uClassPointer;
};

};

// Class Engine.AnimMetaData_SkelControl
// 0x001C (0x0060 - 0x007C)
class UAnimMetaData_SkelControl : public UAnimMetaData
{
public:
TArray<struct FName>                    SkelControlNameList;                // 0x0060 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                       bFullControlOverController : 1;           // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FName                        SkelControlName;                  // 0x0074 (0x0008)
[0x0000000020000000]  CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimMetaData_SkelControl");
}

return uClassPointer;
};

};

// Class Engine.AnimMetaData_SkelControlKeyFrame
// 0x0014 (0x007C - 0x0090)
class UAnimMetaData_SkelControlKeyFrame : public UAnimMetaData_SkelControl
{
public:
TArray<struct FTimeModifier>                KeyFrames;                        // 0x0080 (0x0010)
```

```
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimMetaData_SkelControlKeyFrame");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify
// 0x0004 (0x0060 - 0x0064)
class UAnimNotify : public UObject
{
public:
struct FColor                          NotifyColor;                          // 0x0060 (0x0004)
[0x0000000800000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify");
}

return uClassPointer;
};

bool FindNextNotifyOfClass(class UAnimNodeSequence* AnimSeqInstigator, class UClass*
NotifyClass, struct FAnimNotifyEvent& OutEvent);
};

// Class Engine.AnimNotify_AkEvent
// 0x0018 (0x0064 - 0x007C)
class UAnimNotify_AkEvent : public UAnimNotify
{
public:
class UAkEvent*                          AkEvent;                          // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bFollowActor : 1;                          // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FName                          BoneName;                          // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_AkEvent");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_CameraEffect
// 0x000C (0x0064 - 0x0070)
class UAnimNotify_CameraEffect : public UAnimNotify
{
public:
class UClass*                        CameraLensEffect;                    // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_CameraEffect");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_ClothingMaxDistanceScale
// 0x0014 (0x0064 - 0x0078)
class UAnimNotify_ClothingMaxDistanceScale : public UAnimNotify
{
public:
float                                StartScale;                          // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                EndScale;                            // 0x006C (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              ScaleMode;                           // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                Duration;                            // 0x0074 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_ClothingMaxDistanceScale");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_Footstep
// 0x0008 (0x0064 - 0x006C)
class UAnimNotify_Footstep : public UAnimNotify
{
public:
int32_t                                    FootDown;                              // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_Footstep");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_ForceField
// 0x0020 (0x0064 - 0x0084)
class UAnimNotify_ForceField : public UAnimNotify
{
public:
class UNxForceFieldComponent*                  ForceFieldComponent;                    // 0x0068
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
unsigned long                              bAttach : 1;                            // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FName                               SocketName;                             // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                               BoneName;                               // 0x007C (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_ForceField");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_Kismet
// 0x000C (0x0064 - 0x0070)
class UAnimNotify_Kismet : public UAnimNotify
{
public:
struct FName                    NotifyName;                 // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_Kismet");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_PlayParticleEffect
// 0x0034 (0x0064 - 0x0098)
class UAnimNotify_PlayParticleEffect : public UAnimNotify
{
public:
class UParticleSystem*              PSTemplate;                 // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                   bIsExtremeContent : 1;          // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                   bAttach : 1;                // 0x0070 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                   bPreview : 1;               // 0x0070 (0x0004)
[0x0000000800000001] [0x00000004] (CPF_Edit)
unsigned long                   bSkipIfOwnerIsHidden : 1;           // 0x0070 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
class UParticleSystem*              PSNonExtremeContentTemplate;            // 0x0078
(0x0008) [0x0000000000000001] (CPF_Edit)
struct FName                    SocketName;                 // 0x0080 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
struct FName                              BoneName;                              // 0x0088 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                              BoneSocketModuleActorName;                  // 0x0090
(0x0008) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_PlayParticleEffect");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_Rumble
// 0x001C (0x0064 - 0x0080)
class UAnimNotify_Rumble : public UAnimNotify
{
public:
class UClass*                             PredefinedWaveForm;                        // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UForceFeedbackWaveform*                  WaveForm;                              // 0x0070
(0x0008) [0x0000000004000001] (CPF_Edit | CPF_EditInline)
unsigned long                             bCheckForBasedPlayer : 1;                  // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                               EffectRadius;                          // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_Rumble");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_Script
// 0x001C (0x0064 - 0x0080)
class UAnimNotify_Script : public UAnimNotify
{
public:
```

```
struct FName                              NotifyName;                              // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                              NotifyTickName;                          // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                              NotifyEndName;                           // 0x0078 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_Script");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_Scripted
// 0x0004 (0x0064 - 0x0068)
class UAnimNotify_Scripted : public UAnimNotify
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_Scripted");
}

return uClassPointer;
};

void eventNotifyEnd(class AActor* Owner, class UAnimNodeSequence* AnimSeqInstigator);
void eventNotify(class AActor* Owner, class UAnimNodeSequence* AnimSeqInstigator);
};

// Class Engine.AnimNotify_PawnMaterialParam
// 0x0010 (0x0068 - 0x0078)
class UAnimNotify_PawnMaterialParam : public UAnimNotify_Scripted
{
public:
TArray<struct FScalarParameterInterpStruct>       ScalarParameterInterpArray;                //
0x0068 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_PawnMaterialParam");
}

return uClassPointer;
};

void eventNotify(class AActor* Owner, class UAnimNodeSequence* AnimSeqInstigator);
};

// Class Engine.AnimNotify_ViewShake
// 0x0058 (0x0068 - 0x00C0)
class UAnimNotify_ViewShake : public UAnimNotify_Scripted
{
public:
float                                   Duration;                          // 0x0068 (0x0004)
[0x0000000000020000] (CPF_EditConst)
struct FVector                          RotAmplitude;                      // 0x006C (0x000C)
[0x0000000000020000] (CPF_EditConst)
struct FVector                          RotFrequency;                      // 0x0078 (0x000C)
[0x0000000000020000] (CPF_EditConst)
struct FVector                          LocAmplitude;                      // 0x0084 (0x000C)
[0x0000000000020000] (CPF_EditConst)
struct FVector                          LocFrequency;                      // 0x0090 (0x000C)
[0x0000000000020000] (CPF_EditConst)
float                                   FOVAmplitude;                      // 0x009C (0x0004)
[0x0000000000020000] (CPF_EditConst)
float                                   FOVFrequency;                      // 0x00A0 (0x0004)
[0x0000000000020000] (CPF_EditConst)
unsigned long                           bDoControllerVibration : 1;        // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bUseBoneLocation : 1;              // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                                   ShakeRadius;                       // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                            BoneName;                          // 0x00AC (0x0008)
[0x0000000000000001] (CPF_Edit)
class UCameraShake*                     ShakeParams;                       // 0x00B8 (0x0008)
[0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_ViewShake");
}
return uClassPointer;
};

void eventNotify(class AActor* Owner, class UAnimNodeSequence* AnimSeqInstigator);
};

// Class Engine.AnimNotify_Sound
// 0x0024 (0x0064 - 0x0088)
class UAnimNotify_Sound : public UAnimNotify
{
public:
class USoundCue*                    SoundCue;                       // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bFollowActor : 1;               // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bIgnoreIfActorHidden : 1;       // 0x0070 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FName                        BoneName;                       // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                               PercentToPlay;                  // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               VolumeMultiplier;               // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               PitchMultiplier;                // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_Sound");
}

return uClassPointer;
};

};

// Class Engine.AnimNotify_Trails
// 0x0074 (0x0064 - 0x00D8)
class UAnimNotify_Trails : public UAnimNotify
{
public:
class UParticleSystem*              PSTemplate;                     // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USkeletalMesh*                SampledSkeletalMesh;            // 0x0070 (0x0008)
[0x0000000800020001] (CPF_Edit | CPF_EditConst)
unsigned long                       bIsExtremeContent : 1;          // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bPreview : 1;                   // 0x0078 (0x0004)
```

```
[0x0000000800000001] [0x00000002] (CPF_Edit)
unsigned long                      bPreviewForceExplicit : 1;            // 0x0078 (0x0004)
[0x0000000800000001] [0x00000004] (CPF_Edit)
unsigned long                      bSkipIfOwnerIsHidden : 1;             // 0x0078 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                      bResampleRequired : 1;                // 0x0078 (0x0004)
[0x0000000000000000] [0x00000010]
struct FName                       FirstEdgeSocketName;                  // 0x007C (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                       ControlPointSocketName;               // 0x0084 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                       SecondEdgeSocketName;                 // 0x008C (0x0008)
[0x0000000000000001] (CPF_Edit)
float                         LastStartTime;                   // 0x0094 (0x0004)
[0x0000000000000000]
float                         EndTime;                         // 0x0098 (0x0004)
[0x0000000000000000]
float                         SampleTimeStep;                  // 0x009C (0x0004)
[0x0000000020000000]  CPF_Deprecated)
TArray<struct FTrailSamplePoint>          TrailSampleData;                 // 0x00A0
(0x0010) [0x0000000020400000] (CPF_NeedCtorLink | CPF_Deprecated)
float                         SamplesPerSecond;                // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FTrailSample>             TrailSampledData;                // 0x00B8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                         CurrentTime;                     // 0x00C8 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                         TimeStep;                        // 0x00CC (0x0004)
[0x0000000000002000] (CPF_Transient)
class UAnimNodeSequence*                  AnimNodeSeq;                     // 0x00D0
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_Trails");
}

return uClassPointer;
};

int32_t GetNumSteps(int32_t InLastTrailIndex);
};

// Class Engine.AnimObject
// 0x0030 (0x0060 - 0x0090)
class UAnimObject : public UObject
{
public:
int32_t                        DrawWidth;                       // 0x0060 (0x0004)
```

```
[0x0000000800000000]
int32_t                          DrawHeight;                          // 0x0064 (0x0004)
[0x0000000800000000]
int32_t                          NodePosX;                            // 0x0068 (0x0004)
[0x0000000800000000]
int32_t                          NodePosY;                            // 0x006C (0x0004)
[0x0000000800000000]
int32_t                          OutDrawY;                           // 0x0070 (0x0004)
[0x0000000800000000]
class FString                          CategoryDesc;                         // 0x0078 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
class USkeletalMeshComponent*                    SkelComponent;                         // 0x0088
(0x0008) [0x000000000428200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimObject");
}

return uClassPointer;
};

};

// Class Engine.AnimNode
// 0x00A0 (0x0090 - 0x0130)
class UAnimNode : public UAnimObject
{
public:
unsigned long                          bRelevant : 1;                         // 0x0090 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                          bJustBecameRelevant : 1;                  // 0x0090 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                          bTickDuringPausedAnims : 1;                // 0x0090 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bEditorOnly : 1;                       // 0x0090 (0x0004)
[0x0000000000000002] [0x00000008] (CPF_Const)
unsigned long                          bDisableCaching : 1;                    // 0x0090 (0x0004)
[0x0000000000002002] [0x00000010] (CPF_Const | CPF_Transient)
unsigned long                          bCallScriptEventOnInit : 1;                // 0x0090 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bCallScriptEventOnBecomeRelevant : 1;         // 0x0090
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          bCallScriptEventOnCeaseRelevant : 1;          // 0x0090
(0x0004) [0x0000000000000001] [0x00000080] (CPF_Edit)
int32_t                          NodeTickTag;                          // 0x0094 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                          NodeInitTag;                          // 0x0098 (0x0004)
```

```
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                              NodeEndEventTick;                        // 0x009C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                              TickArrayIndex;                      // 0x00A0 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                              NodeCachedAtomsTag;                        // 0x00A4 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                NodeTotalWeight;                           // 0x00A8 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<class UAnimNodeBlendBase*>               ParentNodes;                              // 0x00B0
(0x0010) [0x0000000000600000] (CPF_NeedCtorLink)
struct FName                              NodeName;                             // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<struct FBoneAtom>                        CachedBoneAtoms;                          // 0x00C8
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
uint8_t                              CachedNumDesiredBones;                     // 0x00D8 (0x0001)
[0x0000000000002000] (CPF_Transient)
struct FBoneAtom                          CachedRootMotionDelta;                    // 0x00E0 (0x0020)
[0x0000000000002000] (CPF_Transient)
int32_t                              bCachedHasRootMotion;                   // 0x0100 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FCurveKey>                      CachedCurveKeys;                          // 0x0108 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                              SearchTag;                               // 0x0118 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FCurveKey>                      LastUpdatedAnimMorphKeys;                  // 0x0120
(0x0010) [0x0000000800422001] (CPF_Edit | CPF_Transient | CPF_EditConst |
CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNode");
}

return uClassPointer;
};

void ReplayAnim();
void StopAnim();
void PlayAnim(unsigned long bLoop, float Rate, float StartTime);
class UAnimNode* FindAnimNode(struct FName InNodeName);
void eventOnCeaseRelevant();
void eventOnBecomeRelevant();
void eventOnInit();
};

// Class Engine.AnimNodeBlendBase
// 0x0015 (0x0130 - 0x0145)
class UAnimNodeBlendBase : public UAnimNode
```

```
{
public:
TArray<struct FAnimBlendChild>                Children;                              // 0x0130 (0x0010)
[0x0000000004400048] (CPF_ExportObject | CPF_EditConstArray | CPF_NeedCtorLink |
CPF_EditInline)
unsigned long                          bFixNumChildren : 1;                    // 0x0140 (0x0004)
[0x0000000000000000] [0x00000001]
uint8_t                                BlendType;                              // 0x0144 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendBase");
}

return uClassPointer;
};

void ReplayAnim();
void StopAnim();
void PlayAnim(unsigned long bLoop, float Rate, float StartTime);
};

// Class Engine.AnimNode_MultiBlendPerBone
// 0x001C (0x0145 - 0x0161)
class UAnimNode_MultiBlendPerBone : public UAnimNodeBlendBase
{
public:
class APawn*                           PawnOwner;                              // 0x0148 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<struct FPerBoneMaskInfo>               MaskList;                              // 0x0150 (0x0010)
[0x0000000004400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink | CPF_EditInline)
uint8_t                                RotationBlendType;                      // 0x0160 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNode_MultiBlendPerBone");
}

return uClassPointer;
};

void SetMaskWeight(int32_t MaskIndex, float DesiredWeight, float BlendTime);
};

// Class Engine.AnimNodeAimOffset
// 0x005F (0x0145 - 0x01A4)
class UAnimNodeAimOffset : public UAnimNodeBlendBase
{
public:
struct FVector2D                          Aim;                                  // 0x0148 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                          AngleOffset;                          // 0x0150 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bForceAimDir : 1;                      // 0x0158 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bBakeFromAnimations : 1;                // 0x0158 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                            bPassThroughWhenNotRendered : 1;             // 0x0158
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                            bSynchronizeNodesInEditor : 1;              // 0x0158 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
int32_t                              PassThroughAtOrAboveLOD;                   // 0x015C (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              ForcedAimDir;                          // 0x0160 (0x0001)
[0x0000000000000001] (CPF_Edit)
TArray<uint8_t>                           RequiredBones;                         // 0x0168 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<uint8_t>                           AimCpntIndexLUT;                        // 0x0178 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
class UAnimNodeAimOffset*                      TemplateNode;                          // 0x0188
(0x0008) [0x0000000000002000] (CPF_Transient)
TArray<struct FAimOffsetProfile>                 Profiles;                            // 0x0190 (0x0010)
[0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
int32_t                              CurrentProfileIndex;                      // 0x01A0 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeAimOffset");
}

return uClassPointer;
};

void SetActiveProfileByIndex(int32_t ProfileIndex);
void SetActiveProfileByName(struct FName ProfileName);
};

// Class Engine.AnimNodeBlend
// 0x0013 (0x0145 - 0x0158)
class UAnimNodeBlend : public UAnimNodeBlendBase
```

```
{
public:
float                               Child2Weight;                          // 0x0148 (0x0004)
[0x0000000000000000]
float                               Child2WeightTarget;                    // 0x014C (0x0004)
[0x0000000000000000]
float                               BlendTimeToGo;                         // 0x0150 (0x0004)
[0x0000000000000000]
unsigned long                             bSkipBlendWhenNotRendered : 1;              // 0x0154
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlend");
}

return uClassPointer;
};

void SetBlendTarget(float BlendTarget, float BlendTime);
};

// Class Engine.AnimNodeAdditiveBlending
// 0x0004 (0x0158 - 0x015C)
class UAnimNodeAdditiveBlending : public UAnimNodeBlend
{
public:
unsigned long                             bPassThroughWhenNotRendered : 1;            // 0x0158
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeAdditiveBlending");
}

return uClassPointer;
};

void SetBlendTarget(float BlendTarget, float BlendTime);
};

// Class Engine.AnimNodeBlendPerBone
// 0x0038 (0x0158 - 0x0190)
class UAnimNodeBlendPerBone : public UAnimNodeBlend
```

```
{
public:
unsigned long                    bForceLocalSpaceBlend : 1;          // 0x0158 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
TArray<struct FName>                  BranchStartBoneName;               // 0x0160
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<float>                    Child2PerBoneWeight;              // 0x0170 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                   LocalToCompReqBones;               // 0x0180 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendPerBone");
}

return uClassPointer;
};

void SetBlendTarget(float BlendTarget, float BlendTime);
};

// Class Engine.AnimNodeCrossfader
// 0x0010 (0x0158 - 0x0168)
class UAnimNodeCrossfader : public UAnimNodeBlend
{
public:
struct FName                     DefaultAnimSeqName;               // 0x0158 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                    bDontBlendOutOneShot : 1;          // 0x0160 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
float                        PendingBlendOutTimeOneShot;          // 0x0164 (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeCrossfader");
}

return uClassPointer;
};

class UAnimNodeSequence* GetActiveChild();
struct FName GetAnimName();
```

```
void BlendToLoopingAnim(struct FName AnimSeqName, float BlendInTime, float Rate);
void PlayOneShotAnim(struct FName AnimSeqName, float BlendInTime, float BlendOutTime,
unsigned long bDontBlendOut, float Rate);
};

// Class Engine.AnimNodePlayCustomAnim
// 0x0008 (0x0158 - 0x0160)
class UAnimNodePlayCustomAnim : public UAnimNodeBlend
{
public:
unsigned long                    bIsPlayingCustomAnim : 1;          // 0x0158 (0x0004)
[0x0000000000000000] [0x00000001]
float                            CustomPendingBlendOutTime;         // 0x015C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodePlayCustomAnim");
}

return uClassPointer;
};

void SetRootBoneAxisOption(uint8_t AxisX, uint8_t AxisY, uint8_t AxisZ);
class UAnimNodeSequence* GetCustomAnimNodeSeq();
void SetActorAnimEndNotification(unsigned long bNewStatus);
void SetCustomAnim(struct FName AnimName);
void StopCustomAnim(float BlendOutTime);
void PlayCustomAnimByDuration(struct FName AnimName, float Duration, float BlendInTime,
float BlendOutTime, unsigned long bLooping, unsigned long bOverride);
float PlayCustomAnim(struct FName AnimName, float Rate, float BlendInTime, float
BlendOutTime, unsigned long bLooping, unsigned long bOverride);
};

// Class Engine.AnimNodeBlendDirectional
// 0x001F (0x0145 - 0x0164)
class UAnimNodeBlendDirectional : public UAnimNodeBlendBase
{
public:
float                            DirDegreesPerSecond;          // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            DirAngle;                  // 0x014C (0x0004)
[0x0000000000000000]
int32_t                           SingleAnimAtOrAboveLOD;          // 0x0150 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRotator                      RotationOffset;              // 0x0154 (0x000C)
[0x0000000000000000]
unsigned long                    bUseAcceleration : 1;          // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendDirectional");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeBlendList
// 0x0027 (0x0145 - 0x016C)
class UAnimNodeBlendList : public UAnimNodeBlendBase
{
public:
TArray<float>                       TargetWeight;                       // 0x0148 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                               BlendTimeToGo;                      // 0x0158 (0x0004)
[0x0000000000000000]
int32_t                             ActiveChildIndex;                   // 0x015C (0x0004)
[0x0000000000000000]
unsigned long                       bPlayActiveChild : 1;               // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bForceChildFullWeightWhenBecomingRelevant : 1; // 0x0160
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       bSkipBlendWhenNotRendered : 1;         // 0x0160
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
float                               SliderPosition;                     // 0x0164 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                             EditorActiveChildIndex;             // 0x0168 (0x0004)
[0x0000000800000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendList");
}

return uClassPointer;
};

void SetActiveChild(int32_t ChildIndex, float BlendTime);
};
```

```
// Class Engine.AnimNodeBlendByBase
// 0x002C (0x016C - 0x0198)
class UAnimNodeBlendByBase : public UAnimNodeBlendList
{
public:
uint8_t                                    Type;                                    // 0x0170 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                                    ActorTag;                                    // 0x0174 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UClass*                                    ActorClass;                                    // 0x0180 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                    BlendTime;                                    // 0x0188 (0x0004)
[0x0000000000000001] (CPF_Edit)
class AActor*                                    CachedBase;                                    // 0x0190 (0x0008)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendByBase");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeBlendByPhysics
// 0x0004 (0x016C - 0x0170)
class UAnimNodeBlendByPhysics : public UAnimNodeBlendList
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendByPhysics");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeBlendByPosture
// 0x0004 (0x016C - 0x0170)
```

```
class UAnimNodeBlendByPosture : public UAnimNodeBlendList
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendByPosture");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeBlendByProperty
// 0x0050 (0x016C - 0x01BC)
class UAnimNodeBlendByProperty : public UAnimNodeBlendList
{
public:
struct FName                    PropertyName;                      // 0x0170 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                   bUseOwnersBase : 1;                // 0x0178 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                   bForceUpdate : 1;                  // 0x0178 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                   bUseSpecificBlendTimes : 1;        // 0x0178 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                   bSynchronizeNodesInEditor : 1;     // 0x0178 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FName                    CachedPropertyName;                // 0x017C (0x0008)
[0x0000000000002000] (CPF_Transient)
struct FPointer                 CachedFloatProperty;               // 0x0188 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                 CachedBoolProperty;                // 0x0190 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                 CachedByteProperty;                // 0x0198 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
class AActor*                   CachedOwner;                       // 0x01A0 (0x0008)
[0x0000000000002000] (CPF_Transient)
float                      BlendTime;                              // 0x01A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                      FloatPropMin;                           // 0x01AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                      FloatPropMax;                           // 0x01B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                      BlendToChild1Time;                      // 0x01B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                      BlendToChild2Time;                      // 0x01B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendByProperty");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeBlendBySpeed
// 0x003C (0x016C - 0x01A8)
class UAnimNodeBlendBySpeed : public UAnimNodeBlendList
{
public:
float                          Speed;                         // 0x0170 (0x0004)
[0x0000000000000000]
int32_t                        LastChannel;                   // 0x0174 (0x0004)
[0x0000000000000000]
float                          BlendUpTime;                   // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BlendDownTime;                 // 0x017C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BlendDownPerc;                 // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<float>                  Constraints;                   // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                  bUseAcceleration : 1;          // 0x0198 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                          BlendUpDelay;                  // 0x019C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BlendDownDelay;                // 0x01A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BlendDelayRemaining;           // 0x01A4 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendBySpeed");
}

return uClassPointer;
};
```

```
};

// Class Engine.AnimNodeRandom
// 0x0024 (0x016C - 0x0190)
class UAnimNodeRandom : public UAnimNodeBlendList
{
public:
TArray<struct FRandomAnimInfo>                RandomInfo;                              // 0x0170
(0x0010) [0x0000000004400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink |
CPF_EditInline)
class UAnimNodeSequence*                      PlayingSeqNode;                          // 0x0180
(0x0008) [0x0000000000002000] (CPF_Transient)
int32_t                                       PendingChildIndex;                       // 0x0188 (0x0004)
[0x0000000000002000] (CPF_Transient)
unsigned long                                 bPickedPendingChildIndex : 1;            // 0x018C (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeRandom");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeBlendMultiBone
// 0x0023 (0x0145 - 0x0168)
class UAnimNodeBlendMultiBone : public UAnimNodeBlendBase
{
public:
TArray<struct FChildBoneBlendInfo>            BlendTargetList;                         // 0x0148
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<uint8_t>                               SourceRequiredBones;                     // 0x0158 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeBlendMultiBone");
}

return uClassPointer;
};

void SetTargetStartBone(int32_t TargetIdx, struct FName StartBoneName, float
PerBoneIncrease);
};

// Class Engine.AnimNodeMirror
// 0x0007 (0x0145 - 0x014C)
class UAnimNodeMirror : public UAnimNodeBlendBase
{
public:
unsigned long                                   bEnableMirroring : 1;                       // 0x0148 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeMirror");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeScalePlayRate
// 0x0007 (0x0145 - 0x014C)
class UAnimNodeScalePlayRate : public UAnimNodeBlendBase
{
public:
float                                           ScaleByValue;                               // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeScalePlayRate");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeScaleRateBySpeed
// 0x0008 (0x014C - 0x0154)
```

```
class UAnimNodeScaleRateBySpeed : public UAnimNodeScalePlayRate
{
public:
float                                  BaseSpeed;                          // 0x0150 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeScaleRateBySpeed");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeSlot
// 0x0027 (0x0145 - 0x016C)
class UAnimNodeSlot : public UAnimNodeBlendBase
{
public:
unsigned long                          bIsPlayingCustomAnim : 1;                  // 0x0148 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                          bEarlyAnimEndNotify : 1;                   // 0x0148 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bSkipBlendWhenNotRendered : 1;             // 0x0148
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bAdditiveAnimationsOverrideSource : 1;     // 0x0148
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bIsBeingUsedByInterpGroup : 1;             // 0x0148 (0x0004)
[0x0000000000002002] [0x00000010] (CPF_Const | CPF_Transient)
unsigned long                          bDontAddToAlwaysTickArray : 1;             // 0x0148 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
float                                  PendingBlendOutTime;                // 0x014C (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                CustomChildIndex;                   // 0x0150 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                TargetChildIndex;                   // 0x0154 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<float>                          TargetWeight;                       // 0x0158 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                  BlendTimeToGo;                      // 0x0168 (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeSlot");
}

return uClassPointer;
};

void TickChildWeights(float DeltaSeconds);
void SetRootBoneRotationOption(uint8_t AxisX, uint8_t AxisY, uint8_t AxisZ);
void SetRootBoneAxisOption(uint8_t AxisX, uint8_t AxisY, uint8_t AxisZ);
class UAnimNodeSequence* GetCustomAnimNodeSeq();
void SetActorAnimEndNotification(unsigned long bNewStatus);
void SetCustomAnim(struct FName AnimName);
void SetAllowPauseAnims(unsigned long bSet);
void StopCustomAnim(float BlendOutTime);
struct FName GetPlayedAnimation();
bool PlayCustomAnimByDuration(struct FName AnimName, float Duration, float BlendInTime,
float BlendOutTime, unsigned long bLooping, unsigned long bOverride);
float PlayCustomAnim(struct FName AnimName, float Rate, float BlendInTime, float
BlendOutTime, unsigned long bLooping, unsigned long bOverride, float StartTime, float EndTime);
};

// Class Engine.AnimNodeSynch
// 0x0013 (0x0145 - 0x0158)
class UAnimNodeSynch : public UAnimNodeBlendBase
{
public:
TArray<struct FSynchGroup>                    Groups;                              // 0x0148 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeSynch");
}

return uClassPointer;
};

void SetGroupRateScale(struct FName GroupName, float NewRateScale);
float GetRelativePosition(struct FName GroupName);
void ForceRelativePosition(struct FName GroupName, float RelativePosition);
class UAnimNodeSequence* GetMasterNodeOfGroup(struct FName GroupName);
void RemoveNodeFromGroup(class UAnimNodeSequence* SeqNode, struct FName
GroupName);
void AddNodeToGroup(class UAnimNodeSequence* SeqNode, struct FName GroupName);
};

// Class Engine.AnimTree
// 0x015B (0x0145 - 0x02A0)
class UAnimTree : public UAnimNodeBlendBase
{
public:
class UAnimTree*                    AnimTreeTemplate;                    // 0x0148 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bEnablePooling : 1;                  // 0x0150 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bUseSavedPose : 1;                   // 0x0150 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                       bBeingEdited : 1;                    // 0x0150 (0x0004)
[0x0000000800002000] [0x00000004] (CPF_Transient)
unsigned long                       bParentNodeArrayBuilt : 1;           // 0x0150 (0x0004)
[0x0000000000200000] [0x00000008]
unsigned long                       bRebuildAnimTickArray : 1;           // 0x0150 (0x0004)
[0x0000000000000000] [0x00000010]
TArray<struct FAnimGroup>                AnimGroups;                     // 0x0158 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FName>                PrioritizedSkelBranches;             // 0x0168 (0x0010)
[0x0000000020400000] (CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FName>                ComposePrePassBoneNames;             // 0x0178
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FName>                ComposePostPassBoneNames;            // 0x0188
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UMorphNodeBase*>            RootMorphNodes;                 // 0x0198
(0x0010) [0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
TArray<struct FSkelControlListHead>      SkelControlLists;               // 0x01A8
(0x0010) [0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
TArray<struct FBoneAtom>                 SavedPose;                      // 0x01B8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                         MorphConnDrawY;                          // 0x01C8 (0x0004)
[0x0000000800000000]
float                           PreviewPlayRate;                         // 0x01CC (0x0004)
[0x0000000800000001] (CPF_Edit)
class USkeletalMesh*                PreviewSkelMesh;                     // 0x01D0 (0x0008)
[0x0000000820000000]  CPF_Deprecated)
class USkeletalMesh*                SocketSkelMesh;                      // 0x01D8 (0x0008)
[0x0000000820000000]  CPF_Deprecated)
class UStaticMesh*                  SocketStaticMesh;                    // 0x01E0 (0x0008)
[0x0000000820000000]  CPF_Deprecated)
struct FName                        SocketName;                          // 0x01E8 (0x0008)
[0x0000000820000000]  CPF_Deprecated)
TArray<class UAnimSet*>                  PreviewAnimSets;                // 0x01F0 (0x0010)
[0x0000000820400000] (CPF_NeedCtorLink | CPF_Deprecated)
TArray<class UMorphTargetSet*>           PreviewMorphSets;               // 0x0200
(0x0010) [0x0000000820400000] (CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FPreviewSkelMeshStruct>    PreviewMeshList;                // 0x0210
(0x0010) [0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                         PreviewMeshIndex;                        // 0x0220 (0x0004)
[0x0000000800000000]
TArray<struct FPreviewSocketStruct>      PreviewSocketList;              // 0x0228
(0x0010) [0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                         PreviewSocketIndex;                      // 0x0238 (0x0004)
[0x0000000800000000]
```

```
TArray<struct FPreviewAnimSetsStruct>            PreviewAnimSetList;                        // 0x0240
(0x0010) [0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                            PreviewAnimSetListIndex;                  // 0x0250 (0x0004)
[0x0000000800000000]
int32_t                            PreviewAnimSetIndex;                     // 0x0254 (0x0004)
[0x0000000800000000]
struct FVector                         PreviewCamPos;                        // 0x0258 (0x000C)
[0x0000000800000000]
struct FRotator                        PreviewCamRot;                        // 0x0264 (0x000C)
[0x0000000800000000]
struct FVector                         PreviewFloorPos;                      // 0x0270 (0x000C)
[0x0000000800000000]
int32_t                            PreviewFloorYaw;                         // 0x027C (0x0004)
[0x0000000800000000]
TArray<class UAnimNodeFrame*>                AnimNodeFrames;                          // 0x0280
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<class UAnimNode*>                 AnimTickArray;                       // 0x0290 (0x0010)
[0x0000000000600000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimTree");
}

return uClassPointer;
};

int32_t GetGroupIndex(struct FName GroupName);
float GetGroupRateScale(struct FName GroupName);
void SetGroupRateScale(struct FName GroupName, float NewRateScale);
float GetGroupRelativePosition(struct FName GroupName);
void ForceGroupRelativePosition(struct FName GroupName, float RelativePosition);
class UAnimNodeSequence* GetGroupNotifyMaster(struct FName GroupName);
class UAnimNodeSequence* GetGroupSynchMaster(struct FName GroupName);
bool SetAnimGroupForNode(class UAnimNodeSequence* SeqNode, struct FName GroupName,
unsigned long bCreateIfNotFound);
void SetUseSavedPose(unsigned long bUseSaved);
class UMorphNodeBase* FindMorphNode(struct FName InNodeName);
void AllSkelControlsNamed(class UClass* SkelClass, struct FName ControlName, class
USkelControlBase*& OutControl);
class USkelControlBase* FindSkelControl(struct FName InControlName);
};

// Class Engine.AnimNodeSequence
// 0x0078 (0x0130 - 0x01A8)
class UAnimNodeSequence : public UAnimNode
{
public:
struct FName                           AnimSeqName;                          // 0x0130 (0x0008)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                        Rate;                        // 0x0138 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                bPlaying : 1;                // 0x013C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                bLooping : 1;                // 0x013C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                bCauseActorAnimEnd : 1;      // 0x013C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                bCauseActorAnimPlay : 1;     // 0x013C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                bZeroRootRotation : 1;       // 0x013C (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                bZeroRootTranslation : 1;    // 0x013C (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                bDisableWarningWhenAnimNotFound : 1;      // 0x013C
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                bNoNotifies : 1;             // 0x013C (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                bForceRefposeWhenNotPlaying : 1;          // 0x013C
(0x0004) [0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                bIsIssuingNotifies : 1;      // 0x013C (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                bForceAlwaysSlave : 1;       // 0x013C (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                bSynchronize : 1;            // 0x013C (0x0004)
[0x0000000000000003] [0x00000800] (CPF_Edit | CPF_Const)
unsigned long                bReverseSync : 1;            // 0x013C (0x0004)
[0x0000000000000003] [0x00001000] (CPF_Edit | CPF_Const)
unsigned long                bShowTimeLineSlider : 1;     // 0x013C (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                bLoopCameraAnim : 1;         // 0x013C (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                bRandomizeCameraAnimLoopStartTime : 1;    // 0x013C
(0x0004) [0x0000000000000001] [0x00008000] (CPF_Edit)
unsigned long                bEditorOnlyAddRefPoseToAdditiveAnimation : 1;  // 0x013C
(0x0004) [0x0000000000000002] [0x00010000] (CPF_Const)
unsigned long                bCheckForFinishAnimEarly : 1;        // 0x013C (0x0004)
[0x0000000000002000] [0x00020000] (CPF_Transient)
unsigned long                bBlendingOut : 1;            // 0x013C (0x0004)
[0x0000000000002000] [0x00040000] (CPF_Transient)
float                        CurrentTime;                 // 0x0140 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                        PreviousTime;                // 0x0144 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                        EndTime;                     // 0x0148 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class UAnimSequence*                  AnimSeq;                    // 0x0150 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                       AnimLinkupIndex;            // 0x0158 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                        NotifyWeightThreshold;       // 0x015C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                 SynchGroupName;              // 0x0160 (0x0008)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                              SynchPosOffset;                        // 0x0168 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UCameraAnim*                           CameraAnim;                           // 0x0170 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UCameraAnimInst*                        ActiveCameraAnimInstance;                 // 0x0178
(0x0008) [0x0000000000002000] (CPF_Transient)
float                              CameraAnimScale;                        // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CameraAnimPlayRate;                      // 0x0184 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CameraAnimBlendInTime;                     // 0x0188 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CameraAnimBlendOutTime;                    // 0x018C (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              RootBoneOption[0x3];                     // 0x0190 (0x0003)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                              RootRotationOption[0x3];                  // 0x0193 (0x0003)
[0x0000000000000003] (CPF_Edit | CPF_Const)
TArray<class USkelControlBase*>                MetaDataSkelControlList;                 // 0x0198
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeSequence");
}

return uClassPointer;
};

void SetRootBoneRotationOption(uint8_t AxisX, uint8_t AxisY, uint8_t AxisZ);
void SetRootBoneAxisOption(uint8_t AxisX, uint8_t AxisY, uint8_t AxisZ);
float GetTimeLeft();
float GetAnimPlaybackLength();
float GetGlobalPlayRate();
float GetGroupRelativePosition();
float FindGroupPosition(float GroupRelativePosition);
float FindGroupRelativePosition(float GroupRelativePosition);
float GetNormalizedPosition();
void SetPosition(float NewTime, unsigned long bFireNotifies);
void ReplayAnim();
void StopAnim();
void PlayAnim(unsigned long bLoop, float InRate, float StartTime);
void SetAnim(struct FName Sequence);
};

// Class Engine.AnimNodeSequenceBlendBase
// 0x0010 (0x01A8 - 0x01B8)
class UAnimNodeSequenceBlendBase : public UAnimNodeSequence
```

```
{
public:
TArray<struct FAnimBlendInfo>                    Anims;                                      // 0x01A8 (0x0010)
[0x0000000004400049] (CPF_Edit | CPF_ExportObject | CPF_EditConstArray | CPF_NeedCtorLink
| CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeSequenceBlendBase");
}

return uClassPointer;
};

};

// Class Engine.AnimNodeSequenceBlendByAim
// 0x0070 (0x01B8 - 0x0228)
class UAnimNodeSequenceBlendByAim : public UAnimNodeSequenceBlendBase
{
public:
struct FVector2D                                 Aim;                                        // 0x01B8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                                 PreviousAim;                                // 0x01C0 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FVector2D                                 HorizontalRange;                            // 0x01C8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                                 VerticalRange;                              // 0x01D0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                                 AngleOffset;                                // 0x01D8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_LU;                                // 0x01E0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_LC;                                // 0x01E8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_LD;                                // 0x01F0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_CU;                                // 0x01F8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_CC;                                // 0x0200 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_CD;                                // 0x0208 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_RU;                                // 0x0210 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_RC;                                // 0x0218 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                     AnimName_RD;                                // 0x0220 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeSequenceBlendByAim");
}

return uClassPointer;
};

void CheckAnimsUpToDate();
};

// Class Engine.AnimNodeFrame
// 0x0038 (0x0090 - 0x00C8)
class UAnimNodeFrame : public UAnimObject
{
public:
int32_t                                SizeX;                              // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                SizeY;                              // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                BorderWidth;                            // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bDrawBox : 1;                           // 0x009C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bFilled : 1;                        // 0x009C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bTileFill : 1;                      // 0x009C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FColor                          BorderColor;                          // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                          FillColor;                          // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                          FillTexture;                        // 0x00A8 (0x0008)
[0x0000000800000001] (CPF_Edit)
class UMaterial*                         FillMaterial;                       // 0x00B0 (0x0008)
[0x0000000800000001] (CPF_Edit)
class FString                          ObjComment;                           // 0x00B8 (0x0010)
[0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNodeFrame");
}
```

```
return uClassPointer;
};

};

// Class Engine.MorphNodeBase
// 0x000C (0x0090 - 0x009C)
class UMorphNodeBase : public UAnimObject
{
public:
struct FName                          NodeName;                          // 0x0090 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                         bDrawSlider : 1;                   // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphNodeBase");
}

return uClassPointer;
};

};

// Class Engine.MorphNodeMultiPose
// 0x0034 (0x009C - 0x00D0)
class UMorphNodeMultiPose : public UMorphNodeBase
{
public:
TArray<class UMorphTarget*>              Targets;                          // 0x00A0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FName>                  MorphNames;                          // 0x00B0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<float>                         Weights;                          // 0x00C0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphNodeMultiPose");
}

return uClassPointer;
```

```
};

bool UpdateMorphTarget(class UMorphTarget* Target, float InWeight);
void RemoveMorphTarget(struct FName MorphTargetName);
bool AddMorphTarget(struct FName MorphTargetName, float InWeight);
};

// Class Engine.MorphNodePose
// 0x0018 (0x009C - 0x00B4)
class UMorphNodePose : public UMorphNodeBase
{
public:
class UMorphTarget*                    Target;                          // 0x00A0 (0x0008)
[0x0000000000002000] (CPF_Transient)
struct FName                       MorphName;                        // 0x00A8 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                          Weight;                          // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphNodePose");
}

return uClassPointer;
};

void SetMorphTarget(struct FName MorphTargetName);
};

// Class Engine.MorphNodeWeightBase
// 0x0014 (0x009C - 0x00B0)
class UMorphNodeWeightBase : public UMorphNodeBase
{
public:
TArray<struct FMorphNodeConn>              NodeConns;                          // 0x00A0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphNodeWeightBase");
}

return uClassPointer;
```

```
};

};

// Class Engine.MorphNodeWeight
// 0x0004 (0x00B0 - 0x00B4)
class UMorphNodeWeight : public UMorphNodeWeightBase
{
public:
float                              NodeWeight;                         // 0x00B0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphNodeWeight");
}

return uClassPointer;
};

void SetNodeWeight(float NewWeight);
};

// Class Engine.MorphNodeWeightByBoneAngle
// 0x0048 (0x00B0 - 0x00F8)
class UMorphNodeWeightByBoneAngle : public UMorphNodeWeightBase
{
public:
float                              Angle;                           // 0x00B0 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                              NodeWeight;                         // 0x00B4 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FName                             BaseBoneName;                          // 0x00B8 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                            BaseBoneAxis;                        // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                            AngleBoneAxis;                       // 0x00C1 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bInvertBaseBoneAxis : 1;                 // 0x00C4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bInvertAngleBoneAxis : 1;                // 0x00C4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                            bControlMaterialParameter : 1;             // 0x00C4 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FName                             AngleBoneName;                         // 0x00C8 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                            MaterialSlotId;                      // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                             ScalarParameterName;                     // 0x00D4 (0x0008)
```

```
[0x0000000000000001] (CPF_Edit)
class UMaterialInstanceConstant*              MaterialInstanceConstant;                  // 0x00E0
(0x0008) [0x0000000000002000] (CPF_Transient)
TArray<struct FBoneAngleMorph>                WeightArray;                               // 0x00E8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphNodeWeightByBoneAngle");
}

return uClassPointer;
};

};

// Class Engine.MorphNodeWeightByBoneRotation
// 0x0040 (0x00B0 - 0x00F0)
class UMorphNodeWeightByBoneRotation : public UMorphNodeWeightBase
{
public:
float                                         Angle;                          // 0x00B0 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                         NodeWeight;                        // 0x00B4 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FName                                  BoneName;                           // 0x00B8 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                                       BoneAxis;                          // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                                 bInvertBoneAxis : 1;                  // 0x00C4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                 bControlMaterialParameter : 1;           // 0x00C4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
TArray<struct FBoneAngleMorph>                WeightArray;                               // 0x00C8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                                       MaterialSlotId;                      // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                                  ScalarParameterName;                  // 0x00DC (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInstanceConstant*              MaterialInstanceConstant;                  // 0x00E8
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.MorphNodeWeightByBoneRotation");
}

return uClassPointer;
};

};

// Class Engine.SkelControlBase
// 0x0070 (0x0090 - 0x0100)
class USkelControlBase : public UAnimObject
{
public:
struct FName                              ControlName;                          // 0x0090 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                             ControlStrength;                          // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                             BlendInTime;                          // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                             BlendOutTime;                          // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                             BlendType;                          // 0x00A4 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                             bPostPhysicsController : 1;                // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                             bSetStrengthFromAnimNode : 1;               // 0x00A8
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                             bInitializedCachedNodeList : 1;            // 0x00A8 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                             bControlledByAnimMetada : 1;               // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                             bInvertMetadataWeight : 1;               // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                             bPropagateSetActive : 1;                // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                             bIgnoreWhenNotRendered : 1;               // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                             bShouldTickInScript : 1;                 // 0x00A8 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                             bShouldTickOwner : 1;                   // 0x00A8 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                             bEnableEaseInOut : 1;                  // 0x00A8 (0x0004)
[0x0000000020020001] [0x00000200] (CPF_Edit | CPF_EditConst | CPF_Deprecated)
float                             StrengthTarget;                          // 0x00AC (0x0004)
[0x0000000000000000]
float                             BlendTimeToGo;                          // 0x00B0 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FName>                             StrengthAnimNodeNameList;                // 0x00B8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UAnimNode*>                             CachedNodeList;                    // 0x00C8
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                             AnimMetadataWeight;                     // 0x00D8 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                             AnimMetaDataUpdateTag;                  // 0x00DC (0x0004)
```

```
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                         BoneScale;                    // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                        ControlTickTag;                  // 0x00E4 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                        IgnoreAtOrAboveLOD;                 // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
class USkelControlBase*                NextControl;                     // 0x00F0 (0x0008)
[0x0000000000000000]
int32_t                        ControlPosX;                    // 0x00F8 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
int32_t                        ControlPosY;                    // 0x00FC (0x0004)
[0x0000000020000000]  CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlBase");
}

return uClassPointer;
};

float GetControlMetadataWeight();
void eventTickSkelControl(float DeltaTime, class USkeletalMeshComponent* SkelComp);
void SetSkelControlStrength(float NewStrength, float InBlendTime);
void SetSkelControlActive(unsigned long bInActive);
};

// Class Engine.SkelControl_CCD_IK
// 0x004C (0x0100 - 0x014C)
class USkelControl_CCD_IK : public USkelControlBase
{
public:
struct FVector                    EffectorLocation;                  // 0x0100 (0x000C)
[0x0000000000000001] (CPF_Edit)
uint8_t                        EffectorLocationSpace;               // 0x010C (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                     EffectorSpaceBoneName;                // 0x0110 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                    EffectorTranslationFromBone;            // 0x0118 (0x000C)
[0x0000000000000001] (CPF_Edit)
int32_t                        NumBones;                      // 0x0124 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                        MaxPerBoneIterations;                // 0x0128 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                        IterationsCount;                  // 0x012C (0x0004)
[0x0000000000000002] (CPF_Const)
float                         Precision;                    // 0x0130 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
unsigned long                          bStartFromTail : 1;                    // 0x0134 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bNoTurnOptimization : 1;                    // 0x0134 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
TArray<float>                          AngleConstraint;                    // 0x0138 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
float                                  MaxAngleSteps;                    // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControl_CCD_IK");
}

return uClassPointer;
};

};

// Class Engine.SkelControl_Multiply
// 0x0004 (0x0100 - 0x0104)
class USkelControl_Multiply : public USkelControlBase
{
public:
float                                  Multiplier;                    // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControl_Multiply");
}

return uClassPointer;
};

};

// Class Engine.SkelControl_TwistBone
// 0x000C (0x0100 - 0x010C)
class USkelControl_TwistBone : public USkelControlBase
{
public:
struct FName                           SourceBoneName;                    // 0x0100 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
float                                   TwistAngleScale;                          // 0x0108 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControl_TwistBone");
}

return uClassPointer;
};

};

// Class Engine.SkelControlLimb
// 0x0058 (0x0100 - 0x0158)
class USkelControlLimb : public USkelControlBase
{
public:
struct FVector                          EffectorLocation;                         // 0x0100 (0x000C)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 EffectorLocationSpace;                    // 0x010C (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 JointTargetLocationSpace;                 // 0x010D (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 JointOffsetSpace;                         // 0x010E (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 BoneAxis;                                 // 0x010F (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 JointAxis;                                // 0x0110 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                            EffectorSpaceBoneName;                    // 0x0114 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                          JointTargetLocation;                      // 0x011C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FName                            JointTargetSpaceBoneName;                 // 0x0128 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                          JointOffset;                              // 0x0130 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FName                            JointOffsetBoneName;                      // 0x013C (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bInvertBoneAxis : 1;                      // 0x0144 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bInvertJointAxis : 1;                     // 0x0144 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                           bRotateJoint : 1;                         // 0x0144 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           bMaintainEffectorRelRot : 1;              // 0x0144 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           bTakeRotationFromEffectorSpace : 1;       // 0x0144
```

```
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                    bAllowStretching : 1;              // 0x0144 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
struct FVector2D                  StretchLimits;                    // 0x0148 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                     StretchRollBoneName;               // 0x0150 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlLimb");
}

return uClassPointer;
};

};

// Class Engine.SkelControlFootPlacement
// 0x0024 (0x0158 - 0x017C)
class USkelControlFootPlacement : public USkelControlLimb
{
public:
float                           FootOffset;                         // 0x0158 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                          FootUpAxis;                        // 0x015C (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRotator                  FootRotOffset;                     // 0x0160 (0x000C)
[0x0000000000000001] (CPF_Edit)
unsigned long                    bInvertFootUpAxis : 1;             // 0x016C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bOrientFootToGround : 1;           // 0x016C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bOnlyEnableForUpAdjustment : 1;         // 0x016C
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
float                           MaxUpAdjustment;                    // 0x0170 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           MaxDownAdjustment;                  // 0x0174 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           MaxFootOrientAdjust;                // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SkelControlFootPlacement");
}

return uClassPointer;
};

};

// Class Engine.SkelControlLookAt
// 0x009C (0x0100 - 0x019C)
class USkelControlLookAt : public USkelControlBase
{
public:
struct FVector                          TargetLocation;                          // 0x0100 (0x000C)
[0x0000000000000001] (CPF_Edit)
uint8_t                              TargetLocationSpace;                      // 0x010C (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                              LookAtAxis;                           // 0x010D (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                              UpAxis;                            // 0x010E (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                              AllowRotationSpace;                      // 0x010F (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                            TargetSpaceBoneName;                       // 0x0110 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bInvertLookAtAxis : 1;                   // 0x0118 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bDefineUpAxis : 1;                     // 0x0118 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                           bInvertUpAxis : 1;                     // 0x0118 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           bEnableLimit : 1;                     // 0x0118 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           bLimitBasedOnRefPose : 1;                  // 0x0118 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                           bDisableBeyondLimit : 1;                   // 0x0118 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                           bNotifyBeyondLimit : 1;                   // 0x0118 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                           bShowLimit : 1;                      // 0x0118 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                           bAllowRotationX : 1;                    // 0x0118 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                           bAllowRotationY : 1;                    // 0x0118 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                           bAllowRotationZ : 1;                    // 0x0118 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
float                                TargetLocationInterpSpeed;                  // 0x011C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                          DesiredTargetLocation;                    // 0x0120 (0x000C)
[0x0000000000000000]
struct FVector                          ActorSpaceLookAtTarget;                    // 0x012C (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                MaxAngle;                           // 0x0138 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                          OuterMaxAngle;                    // 0x013C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DeadZoneAngle;                    // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                    RotationAngleRangeX;                  // 0x0144 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                    RotationAngleRangeY;                  // 0x014C (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                    RotationAngleRangeZ;                  // 0x0154 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                       AllowRotationOtherBoneName;               // 0x015C
(0x0008) [0x0000000000000001] (CPF_Edit)
float                          LookAtAlpha;                    // 0x0164 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          LookAtAlphaTarget;                  // 0x0168 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          LookAtAlphaBlendTimeToGo;                // 0x016C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FVector                    LimitLookDir;                    // 0x0170 (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FVector                    BaseLookDir;                     // 0x017C (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FVector                    BaseBonePos;                      // 0x0188 (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          LastCalcTime;                    // 0x0194 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                         ControlBoneIndex;                  // 0x0198 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlLookAt");
}

return uClassPointer;
};

bool CanLookAtPoint(struct FVector PointLoc, unsigned long bDrawDebugInfo, unsigned long
bDebugUsePersistentLines, unsigned long bDebugFlushLinesFirst);
void SetLookAtAlpha(float DesiredAlpha, float DesiredBlendTime);
void InterpolateTargetLocation(float DeltaTime);
void SetTargetLocation(struct FVector NewTargetLocation);
};

// Class Engine.SkelControlSingleBone
// 0x0030 (0x0100 - 0x0130)
class USkelControlSingleBone : public USkelControlBase
{
```

```
public:
unsigned long                          bApplyTranslation : 1;                    // 0x0100 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bApplyRotation : 1;                       // 0x0100 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bAddTranslation : 1;                      // 0x0100 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bAddRotation : 1;                         // 0x0100 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bRemoveMeshRotation : 1;                  // 0x0100 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
struct FVector                         BoneTranslation;                          // 0x0104 (0x000C)
[0x0000000000000001] (CPF_Edit)
uint8_t                                BoneTranslationSpace;                     // 0x0110 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                BoneRotationSpace;                        // 0x0111 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                           TranslationSpaceBoneName;                 // 0x0114 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FRotator                        BoneRotation;                             // 0x011C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FName                           RotationSpaceBoneName;                    // 0x0128 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlSingleBone");
}

return uClassPointer;
};

};

// Class Engine.SkelControlHandlebars
// 0x0014 (0x0130 - 0x0144)
class USkelControlHandlebars : public USkelControlSingleBone
{
public:
uint8_t                                WheelRollAxis;                            // 0x0130 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                HandlebarRotateAxis;                      // 0x0131 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                           WheelBoneName;                            // 0x0134 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bInvertRotation : 1;                      // 0x013C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                                SteerWheelBoneIndex;                      // 0x0140 (0x0004)
[0x0000000000000000]
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlHandlebars");
}

return uClassPointer;
};

};

// Class Engine.SkelControlWheel
// 0x0018 (0x0130 - 0x0148)
class USkelControlWheel : public USkelControlSingleBone
{
public:
float                                    WheelDisplacement;                         // 0x0130 (0x0004)
[0x0000000000002001] (CPF_Edit | CPF_Transient)
float                                    WheelMaxRenderDisplacement;                // 0x0134 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    WheelRoll;                                 // 0x0138 (0x0004)
[0x0000000000002001] (CPF_Edit | CPF_Transient)
uint8_t                                  WheelRollAxis;                             // 0x013C (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  WheelSteeringAxis;                         // 0x013D (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                    WheelSteering;                             // 0x0140 (0x0004)
[0x0000000000002001] (CPF_Edit | CPF_Transient)
unsigned long                            bInvertWheelRoll : 1;                      // 0x0144 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bInvertWheelSteering : 1;                  // 0x0144 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlWheel");
}

return uClassPointer;
};

};

// Class Engine.SkelControlSpline
// 0x0014 (0x0100 - 0x0114)
class USkelControlSpline : public USkelControlBase
{
public:
int32_t                              SplineLength;                        // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              SplineBoneAxis;                      // 0x0104 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                              BoneRotMode;                         // 0x0105 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bInvertSplineBoneAxis : 1;              // 0x0108 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                               EndSplineTension;                    // 0x010C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               StartSplineTension;                  // 0x0110 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlSpline");
}

return uClassPointer;
};

};

// Class Engine.SkelControlTrail
// 0x0080 (0x0100 - 0x0180)
class USkelControlTrail : public USkelControlBase
{
public:
int32_t                              ChainLength;                         // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              ChainBoneAxis;                       // 0x0104 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bInvertChainBoneAxis : 1;               // 0x0108 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bLimitStretch : 1;                  // 0x0108 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                            bActorSpaceFakeVel : 1;                 // 0x0108 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                            bHadValidStrength : 1;                 // 0x0108 (0x0004)
[0x0000000000000000] [0x00000008]
float                               TrailRelaxation;                     // 0x010C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               StretchLimit;                       // 0x0110 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                           FakeVelocity;                        // 0x0114 (0x000C)
```

```
[0x0000000000000001] (CPF_Edit)
float                          ThisTimstep;                        // 0x0120 (0x0004)
[0x0000000000000000]
TArray<struct FVector>                  TrailBoneLocations;                    // 0x0128 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
uint8_t                         UnknownData00[0x8];                   // 0x0138 (0x0008) MISSED
OFFSET
struct FMatrix                     OldLocalToWorld;                     // 0x0140 (0x0040)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkelControlTrail");
}

return uClassPointer;
};

};

// Class Engine.AnimSequence
// 0x013C (0x0060 - 0x019C)
class UAnimSequence : public UObject
{
public:
struct FName                          SequenceName;                       // 0x0060 (0x0008)
[0x0000000000000000]
TArray<struct FAnimNotifyEvent>              Notifies;                          // 0x0068 (0x0010)
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UAnimMetaData*>                MetaData;                          // 0x0078 (0x0010)
[0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
TArray<struct FSkelControlModifier>            BoneControlModifiers;                  // 0x0088
(0x0010) [0x0000000024400000] (CPF_NeedCtorLink | CPF_EditInline | CPF_Deprecated)
float                          SequenceLength;                      // 0x0098 (0x0004)
[0x0000000000000000]
int32_t                         NumFrames;                          // 0x009C (0x0004)
[0x0000000000000000]
float                          RateScale;                          // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                      bNoLoopingInterpolation : 1;              // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                      bIsAdditive : 1;                    // 0x00A4 (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                      bAdditiveBuiltLooping : 1;                // 0x00A4 (0x0004)
[0x0000000800000000] [0x00000004]
unsigned long                      bDoNotOverrideCompression : 1;             // 0x00A4
(0x0004) [0x0000000800000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                      bHasBeenUsed : 1;                   // 0x00A4 (0x0004)
[0x0000000000002002] [0x00000010] (CPF_Const | CPF_Transient)
```

```
unsigned long                                bWasCompressedWithoutTranslations : 1;        // 0x00A4
(0x0004) [0x0000000800000000] [0x00000020]
TArray<struct FRawAnimSequenceTrack>            RawAnimData;                              // 0x00A8
(0x0010) [0x0000000020400002] (CPF_Const | CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FRawAnimSequenceTrack>            RawAnimationData;                          // 0x00B8
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FTranslationTrack>                TranslationData;                      // 0x00C8
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<struct FRotationTrack>                   RotationData;                     // 0x00D8 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<struct FCurveTrack>                      CurveData;                        // 0x00E8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class UAnimationCompressionAlgorithm*           CompressionScheme;                       //
0x00F8 (0x0008) [0x0000000804020001] (CPF_Edit | CPF_EditConst | CPF_EditInline)
uint8_t                                 TranslationCompressionFormat;             // 0x0100 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                                 RotationCompressionFormat;                // 0x0101 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                                 KeyEncodingFormat;                    // 0x0102 (0x0001)
[0x0000000000000002] (CPF_Const)
TArray<int32_t>                              CompressedTrackOffsets;                 // 0x0108 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                              CompressedByteStream;                   // 0x0118 (0x0010)
[0x0000000000001000] (CPF_Native)
struct FPointer                              TranslationCodec;                      // 0x0128 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)
struct FPointer                              RotationCodec;                        // 0x0130 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)
TArray<struct FBoneAtom>                        AdditiveRefPose;                     // 0x0138 (0x0010)
[0x0000000020400002] (CPF_Const | CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FRawAnimSequenceTrack>            AdditiveBasePose;                        // 0x0148
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FName                                 AdditiveRefName;                        // 0x0158 (0x0008)
[0x0000000800000002] (CPF_Const)
TArray<class UAnimSequence*>                    AdditiveBasePoseAnimSeq;                  // 0x0160
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<class UAnimSequence*>                    AdditiveTargetPoseAnimSeq;                // 0x0170
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<class UAnimSequence*>                    RelatedAdditiveAnimSeqs;                 // 0x0180
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
int32_t                                 EncodingPkgVersion;                   // 0x0190 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                 CompressCommandletVersion;                // 0x0194 (0x0004)
[0x0000000800000002] (CPF_Const)
float                                  UseScore;                          // 0x0198 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.AnimSequence");
}

return uClassPointer;
};

float GetNotifyTimeByClass(class UClass* NotifyClass, float PlayRate, float StartPosition, class
UAnimNotify*& out_Notify, float& out_Duration);
};

// Class Engine.AnimSet
// 0x0128 (0x0060 - 0x0188)
class UAnimSet : public UObject
{
public:
unsigned long                          bAnimRotationOnly : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<struct FName>                          TrackBoneNames;                          // 0x0068 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class UAnimSequence*>                    Sequences;                               // 0x0078 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                                UnknownData00[0x50];                      // 0x0088 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AnimSet.SequenceCache
TArray<struct FAnimSetMeshLinkup>               LinkupCache;                             // 0x00D8
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
uint8_t                                UnknownData01[0x50];                      // 0x00E8 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AnimSet.SkelMesh2LinkupCache
TArray<uint8_t>                        BoneUseAnimTranslation;                   // 0x0138 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<uint8_t>                        ForceUseMeshTranslation;                  // 0x0148 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FName>                          UseTranslationBoneNames;                 // 0x0158
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FName>                          ForceMeshTranslationBoneNames;            // 0x0168
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FName                           PreviewSkelMeshName;                      // 0x0178 (0x0008)
[0x0000000000000000]
struct FName                           BestRatioSkelMeshName;                    // 0x0180 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimSet");
}

return uClassPointer;
};

};
```

```
// Class Engine.MorphTarget
// 0x001C (0x0060 - 0x007C)
class UMorphTarget : public UObject
{
public:
TArray<int32_t>                         MorphLODModels;                          // 0x0060 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                                 MaterialSlotId;                          // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                            ScalarParameterName;                     // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphTarget");
}

return uClassPointer;
};

};

// Class Engine.MorphTargetSet
// 0x0028 (0x0060 - 0x0088)
class UMorphTargetSet : public UObject
{
public:
TArray<class UMorphTarget*>             Targets;                                 // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class USkeletalMesh*                    BaseSkelMesh;                            // 0x0070 (0x0008)
[0x0000000000000000]
struct FArray_Mirror                    RawWedgePointIndices;                    // 0x0078 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphTargetSet");
}

return uClassPointer;
};

class UMorphTarget* FindMorphTarget(struct FName MorphTargetName);
```

```
};

// Class Engine.MorphWeightSequence
// 0x0000 (0x0060 - 0x0060)
class UMorphWeightSequence : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MorphWeightSequence");
}

return uClassPointer;
};

};

// Class Engine.DecalActorBase
// 0x0010 (0x0268 - 0x0278)
class ADecalActorBase : public AActor
{
public:
struct FPointer                    VfTable_IEditorLinkSelectionInterface;      // 0x0268
(0x0008) [0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UDecalComponent*                    Decal;                          // 0x0270 (0x0008)
[0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DecalActorBase");
}

return uClassPointer;
};

};

// Class Engine.DecalActor
// 0x0000 (0x0278 - 0x0278)
class ADecalActor : public ADecalActorBase
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DecalActor");
}

return uClassPointer;
};

};

// Class Engine.DecalActorMovable
// 0x0000 (0x0278 - 0x0278)
class ADecalActorMovable : public ADecalActorBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DecalActorMovable");
}

return uClassPointer;
};

};

// Class Engine.DecalManager
// 0x0040 (0x0268 - 0x02A8)
class ADecalManager : public AActor
{
public:
class UDecalComponent*                DecalTemplate;                // 0x0268 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
TArray<class UDecalComponent*>        PoolDecals;                   // 0x0270
(0x0010) [0x0000000004480008] (CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)
int32_t                               MaxActiveDecals;              // 0x0280 (0x0004)
[0x0000000000000000]
float                                 DecalLifeSpan;                // 0x0284 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                                 DecalDepthBias;               // 0x0288 (0x0004)
[0x0000000000000000]
struct FVector2D                      DecalBlendRange;              // 0x028C (0x0008)
```

```
[0x0000000000000000]
TArray<struct FActiveDecalInfo>                ActiveDecals;                          // 0x0298 (0x0010)
[0x0000000000480000] (CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DecalManager");
}

return uClassPointer;
};

class UDecalComponent* SpawnDecal(class UMaterialInterface* DecalMaterial, struct FVector
DecalLocation, struct FRotator DecalOrientation, float Width, float Height, float Thickness,
unsigned long bNoClip, float DecalRotation, class UPrimitiveComponent* HitComponent,
unsigned long bProjectOnTerrain, unsigned long bProjectOnSkeletalMeshes, struct FName
HitBone, int32_t HitNodeIndex, int32_t HitLevelIndex, float InDecalLifeSpan, int32_t
InFracturedStaticMeshComponentIndex, float InDepthBias, struct FVector2D InBlendRange);
class UDecalComponent* GetPooledComponent();
static void SetDecalParameters(class UDecalComponent* TheDecal, class UMaterialInterface*
DecalMaterial, struct FVector DecalLocation, struct FRotator DecalOrientation, float Width, float
Height, float Thickness, unsigned long bNoClip, float DecalRotation, class UPrimitiveComponent*
HitComponent, unsigned long bProjectOnTerrain, unsigned long bProjectOnSkeletalMeshes,
struct FName HitBone, int32_t HitNodeIndex, int32_t HitLevelIndex, int32_t
InFracturedStaticMeshComponentIndex, float DepthBias, struct FVector2D BlendRange);
bool CanSpawnDecals();
void eventSpawnDecalOnParticleCollision(class UMaterialInterface* DecalMaterial, struct
FVector DecalLocation, struct FRotator DecalOrientation, float Width, float Height, float
Thickness, unsigned long bNoClip, float DecalLifetime, float InDepthBias, struct FVector2D
InBlendRange);
void eventDecalFinished(class UDecalComponent* Decal);
static bool AreDynamicDecalsEnabled();
};

// Class Engine.DecalComponent
// 0x0188 (0x0258 - 0x03E0)
class UDecalComponent : public UPrimitiveComponent
{
public:
struct FPointer                        VfTable_IISetParameter;                 // 0x0258 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UMaterialInterface*                   DecalMaterial;                      // 0x0260 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                                  Width;                              // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  Height;                             // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  TileX;                              // 0x0270 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                              TileY;                              // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              OffsetX;                            // 0x0278 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              OffsetY;                            // 0x027C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              DecalRotation;                      // 0x0280 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              FieldOfView;                        // 0x0284 (0x0004)
[0x0000000000000000]
float                              NearPlane;                          // 0x0288 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              FarPlane;                           // 0x028C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                          Location;                          // 0x0290 (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FRotator                         Orientation;                       // 0x029C (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FVector                          HitLocation;                       // 0x02A8 (0x000C)
[0x0000000000000000]
struct FVector                          HitNormal;                         // 0x02B4 (0x000C)
[0x0000000000000000]
struct FVector                          HitTangent;                        // 0x02C0 (0x000C)
[0x0000000000000000]
struct FVector                          HitBinormal;                       // 0x02CC (0x000C)
[0x0000000000000000]
unsigned long                           bNoClip : 1;                       // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bStaticDecal : 1;                    // 0x02D8 (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                           bProjectOnBackfaces : 1;                 // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           bProjectOnHidden : 1;                   // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           bProjectOnBSP : 1;                    // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                           bProjectOnStaticMeshes : 1;               // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                           bProjectOnSkeletalMeshes : 1;              // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                           bProjectOnTerrain : 1;                  // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                           bProjectOnOwner : 1;                   // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                           bFlipBackfaceDirection : 1;               // 0x02D8 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                           bMovableDecal : 1;                    // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                           bHasBeenAttached : 1;                  // 0x02D8 (0x0004)
[0x0000000000002000] [0x00000800] (CPF_Transient)
unsigned long                           bDecalMaterialSetAtRunTime : 1;            // 0x02D8 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
class UPrimitiveComponent*                     HitComponent;                   // 0x02E0 (0x0008)
[0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component | CPF_EditInline)
```

```
struct FName                                HitBone;                                // 0x02E8 (0x0008)
[0x0000000000002000] (CPF_Transient)
int32_t                             HitNodeIndex;                           // 0x02F0 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                             HitLevelIndex;                          // 0x02F4 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                             FracturedStaticMeshComponentIndex;           // 0x02F8 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<int32_t>                             HitNodeIndices;                         // 0x0300 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<struct FDecalReceiver>                 DecalReceivers;                         // 0x0310 (0x0010)
[0x0000000001680002] (CPF_Const | CPF_Component | CPF_NeedCtorLink)
TArray<struct FPointer>                     StaticReceivers;                        // 0x0320 (0x0010)
[0x0000000001203002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                             ReleaseResourcesFence;                  // 0x0330 (0x0008)
[0x0000000000203002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPlane>                       Planes;                                 // 0x0338 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                               DepthBias;                              // 0x0348 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               SlopeScaleDepthBias;                    // 0x034C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             SortOrder;                              // 0x0350 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               BackfaceAngle;                          // 0x0354 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                            BlendRange;                             // 0x0358 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                               StreamingDistanceMultiplier;              // 0x0360 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                             DecalTransform;                         // 0x0364 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                             FilterMode;                             // 0x0365 (0x0001)
[0x0000000000000001] (CPF_Edit)
TArray<class AActor*>                       Filter;                                 // 0x0368 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UPrimitiveComponent*>            ReceiverImages;                         // 0x0378
(0x0010) [0x0000000004480009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
struct FVector                              ParentRelativeLocation;                 // 0x0388 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FRotator                             ParentRelativeOrientation;              // 0x0394 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FMatrix                              ParentRelLocRotMatrix;                  // 0x03A0 (0x0040)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DecalComponent");
```

```
}

return uClassPointer;
};

class UMaterialInstance* GetOrCreateDecalMaterialInstance();
void SetActorParameter(struct FName Key, class AActor* Value);
void SetLinearColorParameter(struct FName Key, struct FLinearColor Value);
void SetVectorParameter(struct FName Key, struct FVector Value);
void SetFloatParameter(struct FName Key, float Value);
void SetNameParameter(struct FName Key, struct FName Value);
bool IsWaitingForResetToDefaultsToComplete();
class UMaterialInterface* GetDecalMaterial();
void SetDecalMaterial(class UMaterialInterface* NewDecalMaterial);
void ResetToDefaults();
};

// Class Engine.ActorFactoryDecal
// 0x000C (0x009C - 0x00A8)
class UActorFactoryDecal : public UActorFactory
{
public:
class UMaterialInterface*                DecalMaterial;                    // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryDecal");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryDecalMovable
// 0x0000 (0x00A8 - 0x00A8)
class UActorFactoryDecalMovable : public UActorFactoryDecal
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryDecalMovable");
```

```
}

return uClassPointer;
};

};

// Class Engine.Material
// 0x0664 (0x0274 - 0x08D8)
class UMaterial : public UMaterialInterface
{
public:
class UPhysicalMaterial*                PhysMaterial;                          // 0x0278 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UClass*                           PhysicalMaterial;                      // 0x0280 (0x0008)
[0x0000000000000000]
class UTexture2D*                       PhysMaterialMask;                      // 0x0288 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                                 PhysMaterialMaskUVChannel;             // 0x0290 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UPhysicalMaterial*                BlackPhysicalMaterial;                 // 0x0298 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UPhysicalMaterial*                WhitePhysicalMaterial;                 // 0x02A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FColorMaterialInput              DiffuseColor;                          // 0x02A8 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScalarMaterialInput             DiffusePower;                          // 0x02E8 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FColorMaterialInput              SpecularColor;                         // 0x0328 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScalarMaterialInput             SpecularPower;                         // 0x0368 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FVectorMaterialInput             Normal;                                // 0x03A8 (0x0048)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FColorMaterialInput              EmissiveColor;                         // 0x03F0 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScalarMaterialInput             Opacity;                               // 0x0430 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScalarMaterialInput             OpacityMask;                           // 0x0470 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                   OpacityMaskClipValue;                  // 0x04B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   ShadowDepthBias;                       // 0x04B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector2MaterialInput            Distortion;                            // 0x04B8 (0x0048)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                                 BlendMode;                             // 0x0500 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 LightingModel;                         // 0x0501 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 D3D11TessellationMode;                 // 0x0502 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FColorMaterialInput              CustomLighting;                        // 0x0508 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
struct FColorMaterialInput                    CustomSkylightDiffuse;                    // 0x0548
(0x0040) [0x0000000000400000] (CPF_NeedCtorLink)
struct FVectorMaterialInput                   AnisotropicDirection;                    // 0x0588 (0x0048)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScalarMaterialInput                   TwoSidedLightingMask;                    // 0x05D0
(0x0040) [0x0000000000400000] (CPF_NeedCtorLink)
struct FColorMaterialInput                    TwoSidedLightingColor;                   // 0x0610
(0x0040) [0x0000000000400000] (CPF_NeedCtorLink)
struct FVectorMaterialInput                   WorldPositionOffset;                     // 0x0650 (0x0048)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FVectorMaterialInput                   WorldDisplacement;                       // 0x0698 (0x0048)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FScalarMaterialInput                   TessellationMultiplier;                  // 0x06E0 (0x0040)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FColorMaterialInput                    SubsurfaceInscatteringColor;             // 0x0720
(0x0040) [0x0000000000400000] (CPF_NeedCtorLink)
struct FColorMaterialInput                    SubsurfaceAbsorptionColor;               // 0x0760
(0x0040) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScalarMaterialInput                   SubsurfaceScatteringRadius;              // 0x07A0
(0x0040) [0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                                 EnableSubsurfaceScattering : 1;          // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                 EnableSeparateTranslucency : 1;          // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                 EnableSeparateTranslucencyDuring4KCheckerboard : 1;//
0x07E0 (0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                 bEnableMaskedAntialiasing : 1;           // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                                 TwoSided : 1;                            // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                                 TwoSidedSeparatePass : 1;                // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                                 bDisableDepthTest : 1;                   // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                                 bSceneTextureRenderBehindTranslucency : 1;     // 0x07E0
(0x0004) [0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                                 bAllowFog : 1;                           // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                                 bTranslucencyReceiveDominantShadowsFromStatic : 1;//
0x07E0 (0x0004) [0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                                 bTranslucencyInheritDominantShadowsFromOpaque : 1;//
0x07E0 (0x0004) [0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                                 bAllowTranslucencyDoF : 1;               // 0x07E0 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                                 bUseOneLayerDistortion : 1;              // 0x07E0 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                                 bUseLitTranslucencyDepthPass : 1;             // 0x07E0
(0x0004) [0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                                 bUseLitTranslucencyPostRenderDepthPass : 1;    // 0x07E0
(0x0004) [0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                                 bCastLitTranslucencyShadowAsMasked : 1;        // 0x07E0
(0x0004) [0x0000000000000001] [0x00008000] (CPF_Edit)
unsigned long                                 bDrawLitTranslucencyPrepassInOpaquePrepassWhen4KCB :
1;// 0x07E0 (0x0004) [0x0000000000000001] [0x00010000] (CPF_Edit)
```

```
unsigned long                          bUsedAsLightFunction : 1;                  // 0x07E0 (0x0004)
[0x0000000000000003] [0x00020000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithFogVolumes : 1;                   // 0x07E0 (0x0004)
[0x0000000000000003] [0x00040000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedAsSpecialEngineMaterial : 1;          // 0x07E0
(0x0004) [0x0000000000200002] [0x00080000] (CPF_Const)
unsigned long                          bUsedWithSkeletalMesh : 1;                 // 0x07E0 (0x0004)
[0x0000000000000003] [0x00100000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithTerrain : 1;                  // 0x07E0 (0x0004)
[0x0000000000000003] [0x00200000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithLandscape : 1;                    // 0x07E0 (0x0004)
[0x0000000000000003] [0x00400000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithMobileLandscape : 1;              // 0x07E0 (0x0004)
[0x0000000000000003] [0x00800000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithFracturedMeshes : 1;              // 0x07E0 (0x0004)
[0x0000000000000003] [0x01000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithParticleSystem : 1;               // 0x07E0 (0x0004)
[0x0000000000000002] [0x02000000] (CPF_Const)
unsigned long                          bUsedWithParticleSprites : 1;              // 0x07E0 (0x0004)
[0x0000000000000003] [0x04000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithBeamTrails : 1;                   // 0x07E0 (0x0004)
[0x0000000000000003] [0x08000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithParticleSubUV : 1;                // 0x07E0 (0x0004)
[0x0000000000000003] [0x10000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithSpeedTree : 1;                    // 0x07E0 (0x0004)
[0x0000000000000003] [0x20000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithStaticLighting : 1;               // 0x07E0 (0x0004)
[0x0000000000000003] [0x40000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithLensFlare : 1;                    // 0x07E0 (0x0004)
[0x0000000000000003] [0x80000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithGammaCorrection : 1;              // 0x07E4
(0x0004) [0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithInstancedMeshParticles : 1;       // 0x07E4
(0x0004) [0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithFluidSurfaces : 1;                // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithDecals : 1;                   // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithMaterialEffect : 1;               // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithMorphTargets : 1;                 // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000020] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithRadialBlur : 1;                   // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithInstancedMeshes : 1;              // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000080] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithSplineMeshes : 1;                 // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000100] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithAPEXMeshes : 1;                   // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000200] (CPF_Edit | CPF_Const)
unsigned long                          bUsedWithScreenDoorFade : 1;               // 0x07E4 (0x0004)
[0x0000000000000003] [0x00000400] (CPF_Edit | CPF_Const)
unsigned long                          bEnableCrackFreeDisplacement : 1;          // 0x07E4
(0x0004) [0x0000000000000003] [0x00000800] (CPF_Edit | CPF_Const)
```

```
unsigned long                          bUseImageBasedReflections : 1;              // 0x07E4 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                          Wireframe : 1;                     // 0x07E4 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                          bPerPixelCameraVector : 1;               // 0x07E4 (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                          bAllowLightmapSpecular : 1;               // 0x07E4 (0x0004)
[0x0000000000000001] [0x00008000] (CPF_Edit)
unsigned long                          bIsFallbackMaterial : 1;              // 0x07E4 (0x0004)
[0x0000000020000000] [0x00010000]  CPF_Deprecated)
unsigned long                          bUsesDistortion : 1;                  // 0x07E4 (0x0004)
[0x0000000000000000] [0x00020000]
unsigned long                          bIsMasked : 1;                      // 0x07E4 (0x0004)
[0x0000000000000000] [0x00040000]
unsigned long                          bIsPreviewMaterial : 1;                 // 0x07E4 (0x0004)
[0x0000000000202000] [0x00080000] (CPF_Transient)
float                            ImageReflectionNormalDampening;             // 0x07E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                        MaterialResources[0x2];                  // 0x07F0 (0x0010)
[0x0000000000201002] (CPF_Const | CPF_Native)
struct FPointer                        DefaultMaterialInstances[0x3];            // 0x0800 (0x0018)
[0x0000000000201002] (CPF_Const | CPF_Native)
int32_t                          EditorX;                            // 0x0818 (0x0004)
[0x0000000000000000]
int32_t                          EditorY;                            // 0x081C (0x0004)
[0x0000000000000000]
int32_t                          EditorPitch;                          // 0x0820 (0x0004)
[0x0000000000000000]
int32_t                          EditorYaw;                            // 0x0824 (0x0004)
[0x0000000000000000]
int32_t                          MaterialEditorX;                        // 0x0828 (0x0004)
[0x0000000800000000]
int32_t                          MaterialEditorY;                        // 0x082C (0x0004)
[0x0000000800000000]
int32_t                          MaterialWidth;                         // 0x0830 (0x0004)
[0x0000000800000000]
int32_t                          MaterialHeight;                         // 0x0834 (0x0004)
[0x0000000800000000]
TArray<class UMaterialExpression*>             Expressions;                          // 0x0838
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<class UMaterialExpressionComment*>         EditorComments;                        // 0x0848
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<struct FMaterialFunctionInfo>            MaterialFunctionInfos;                  // 0x0858
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                          UnknownData00[0x50];                // 0x0868 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.Material.EditorParameters
TArray<class UTexture*>                   ReferencedTextures;                 // 0x08B8 (0x0010)
[0x0000000020400002] (CPF_Const | CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FGuid>                     ReferencedTextureGuids;              // 0x08C8 (0x0010)
[0x0000000800400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Material");
}

return uClassPointer;
};

bool GetFontParameterValue(struct FName ParameterName, class UFont*& OutFontValue,
int32_t& OutFontPage);
bool GetTextureParameterValue(struct FName ParameterName, class UTexture*& OutValue);
bool GetScalarParameterValue(struct FName ParameterName, float& OutValue);
bool GetVectorParameterValue(struct FName ParameterName, struct FLinearColor& OutValue);
bool GetParameterDesc(struct FName ParameterName, class FString& OutDesc);
};

// Class Engine.DecalMaterial
// 0x0000 (0x08D8 - 0x08D8)
class UDecalMaterial : public UMaterial
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DecalMaterial");
}

return uClassPointer;
};

};

// Class Engine.FogVolumeDensityInfo
// 0x0014 (0x0268 - 0x027C)
class AFogVolumeDensityInfo : public AInfo
{
public:
class UFogVolumeDensityComponent*                DensityComponent;                          // 0x0268
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class UStaticMeshComponent*                  AutomaticMeshComponent;                    // 0x0270
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
unsigned long                            bEnabled : 1;                              // 0x0278 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FogVolumeDensityInfo");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct AFogVolumeDensityInfo_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct AFogVolumeDensityInfo_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.FogVolumeConeDensityInfo
// 0x0004 (0x027C - 0x0280)
class AFogVolumeConeDensityInfo : public AFogVolumeDensityInfo
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FogVolumeConeDensityInfo");
}

return uClassPointer;
};

};

// Class Engine.FogVolumeConstantDensityInfo
// 0x0004 (0x027C - 0x0280)
class AFogVolumeConstantDensityInfo : public AFogVolumeDensityInfo
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.FogVolumeConstantDensityInfo");
}

return uClassPointer;
};

};

// Class Engine.FogVolumeLinearHalfspaceDensityInfo
// 0x0004 (0x027C - 0x0280)
class AFogVolumeLinearHalfspaceDensityInfo : public AFogVolumeDensityInfo
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FogVolumeLinearHalfspaceDensityInfo");
}

return uClassPointer;
};

};

// Class Engine.FogVolumeSphericalDensityInfo
// 0x0004 (0x027C - 0x0280)
class AFogVolumeSphericalDensityInfo : public AFogVolumeDensityInfo
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FogVolumeSphericalDensityInfo");
}

return uClassPointer;
};

};

// Class Engine.ExponentialHeightFogComponent
// 0x002F (0x009D - 0x00CC)
class UExponentialHeightFogComponent : public UActorComponent
{
```

```
public:
unsigned long                          bEnabled : 1;                          // 0x00A0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
float                                  FogHeight;                             // 0x00A4 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                  FogDensity;                            // 0x00A8 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                                  FogHeightFalloff;                      // 0x00AC (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                                  FogMaxOpacity;                         // 0x00B0 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                                  StartDistance;                         // 0x00B4 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                                  LightTerminatorAngle;                  // 0x00B8 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                                  OppositeLightBrightness;               // 0x00BC (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
struct FColor                          OppositeLightColor;                    // 0x00C0 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
float                                  LightInscatteringBrightness;           // 0x00C4 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
struct FColor                          LightInscatteringColor;                // 0x00C8 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ExponentialHeightFogComponent");
}

return uClassPointer;
};

void SetEnabled(unsigned long bSetEnabled);
};

// Class Engine.FogVolumeDensityComponent
// 0x0053 (0x009D - 0x00F0)
class UFogVolumeDensityComponent : public UActorComponent
{
public:
class UMaterialInterface*              FogMaterial;                           // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              DefaultFogVolumeMaterial;              // 0x00A8
(0x0008) [0x0000000000000000]
unsigned long                          bEnabled : 1;                          // 0x00B0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                          bAffectsTranslucency : 1;              // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bOnlyAffectsTranslucency : 1;          // 0x00B0 (0x0004)
```

```
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FLinearColor                SimpleLightColor;                 // 0x00B4 (0x0010)
[0x0000000200000001] (CPF_Edit)
struct FLinearColor                ApproxFogLightColor;               // 0x00C4 (0x0010)
[0x0000000200000001] (CPF_Edit)
float                              StartDistance;                 // 0x00D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                              MaxDistance;                   // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class AActor*>                FogVolumeActors;                 // 0x00E0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FogVolumeDensityComponent");
}

return uClassPointer;
};

void SetEnabled(unsigned long bSetEnabled);
};

// Class Engine.FogVolumeConeDensityComponent
// 0x0030 (0x00F0 - 0x0120)
class UFogVolumeConeDensityComponent : public UFogVolumeDensityComponent
{
public:
float                              MaxDensity;                    // 0x00F0 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FVector                     ConeVertex;                      // 0x00F4 (0x000C)
[0x0000000200000001] (CPF_Edit)
float                              ConeRadius;                    // 0x0100 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FVector                     ConeAxis;                       // 0x0104 (0x000C)
[0x0000000200000001] (CPF_Edit)
float                              ConeMaxAngle;                   // 0x0110 (0x0004)
[0x0000000200000001] (CPF_Edit)
class UDrawLightConeComponent*           PreviewCone;                      // 0x0118
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.FogVolumeConeDensityComponent");
}

return uClassPointer;
};

};

// Class Engine.FogVolumeConstantDensityComponent
// 0x0004 (0x00F0 - 0x00F4)
class UFogVolumeConstantDensityComponent : public UFogVolumeDensityComponent
{
public:
float                               Density;                            // 0x00F0 (0x0004)
[0x0000000200000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FogVolumeConstantDensityComponent");
}

return uClassPointer;
};

};

// Class Engine.FogVolumeLinearHalfspaceDensityComponent
// 0x0020 (0x00F0 - 0x0110)
class UFogVolumeLinearHalfspaceDensityComponent : public UFogVolumeDensityComponent
{
public:
float                               PlaneDistanceFactor;                // 0x00F0 (0x0004)
[0x0000000200000001] (CPF_Edit)
uint8_t                             UnknownData00[0xC];                 // 0x00F4 (0x000C) MISSED
OFFSET
struct FPlane                       HalfspacePlane;                     // 0x0100 (0x0010)
[0x0000000200000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.FogVolumeLinearHalfspaceDensityComponent");
}
```

```
return uClassPointer;
};

};

// Class Engine.FogVolumeSphericalDensityComponent
// 0x0020 (0x00F0 - 0x0110)
class UFogVolumeSphericalDensityComponent : public UFogVolumeDensityComponent
{
public:
float                                   MaxDensity;                             // 0x00F0 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FVector                          SphereCenter;                           // 0x00F4 (0x000C)
[0x0000000000000000]
float                                   SphereRadius;                           // 0x0100 (0x0004)
[0x0000000000000000]
class UDrawLightRadiusComponent*                PreviewSphereRadius;                    // 0x0108
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FogVolumeSphericalDensityComponent");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryFogVolumeConstantDensityInfo
// 0x0010 (0x009C - 0x00AC)
class UActorFactoryFogVolumeConstantDensityInfo : public UActorFactory
{
public:
class UMaterialInterface*                       SelectedMaterial;                       // 0x00A0 (0x0008)
[0x0000000000000000]
unsigned long                                   bNothingSelected : 1;                   // 0x00A8 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ActorFactoryFogVolumeConstantDensityInfo");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryFogVolumeLinearHalfspaceDensityInfo
// 0x0004 (0x00AC - 0x00B0)
class UActorFactoryFogVolumeLinearHalfspaceDensityInfo : public
UActorFactoryFogVolumeConstantDensityInfo
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ActorFactoryFogVolumeLinearHalfspaceDensityInfo");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryFogVolumeSphericalDensityInfo
// 0x0004 (0x00AC - 0x00B0)
class UActorFactoryFogVolumeSphericalDensityInfo : public
UActorFactoryFogVolumeConstantDensityInfo
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ActorFactoryFogVolumeSphericalDensityInfo");
}

return uClassPointer;
};

};

// Class Engine.ApexDestructibleActor
// 0x0060 (0x0268 - 0x02C8)
class AApexDestructibleActor : public AActor
{
public:
class UDynamicLightEnvironmentComponent*          LightEnvironment;                        //
0x0268 (0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
unsigned long                                     bFractureMaterialOverride : 1;              // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                     bPlaySingleFractureMaterialEffect : 1;        // 0x0270
(0x0004) [0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
TArray<class UFractureMaterial*>                  FractureMaterials;                        // 0x0278
(0x0010) [0x0000000000400043] (CPF_Edit | CPF_Const | CPF_EditConstArray |
CPF_NeedCtorLink)
class UApexStaticDestructibleComponent*           StaticDestructibleComponent;                //
0x0288 (0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject |
CPF_EditConst | CPF_Component | CPF_EditInline)
int32_t                                           LOD;                            // 0x0290 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
TArray<uint8_t>                                   VisibilityFactors;                        // 0x0298 (0x0010)
[0x0000000000500000] (CPF_NeedCtorLink)
TArray<class USoundCue*>                          FractureSounds;                           // 0x02A8 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<class UParticleSystem*>                    FractureParticleEffects;                  // 0x02B8
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexDestructibleActor");
}

return uClassPointer;
};

void OnSetMaterial(class USeqAct_SetMaterial* Action);
void eventPostBeginPlay();
void CacheFractureEffects();
void eventSpawnFractureEmitter(class UParticleSystem* EmitterTemplate, struct FVector
SpawnLocation, struct FVector SpawnDirection);
};

// Class Engine.FracturedStaticMeshActor
// 0x0088 (0x0268 - 0x02F0)
class AFracturedStaticMeshActor : public AActor
{
public:
int32_t                                           MaxPartsToSpawnAtOnce;                    // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UFracturedStaticMeshComponent*              FracturedStaticMeshComponent;               //
```

```
0x0270 (0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject |
CPF_EditConst | CPF_Component | CPF_EditInline)
class UFracturedSkinnedMeshComponent*            SkinnedComponent;                          //
0x0278 (0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
TArray<int32_t>                                  ChunkHealth;                              // 0x0280 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                                    bHasShownMissingSoundWarning : 1;            // 0x0290
(0x0004) [0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                                    bBreakChunksOnActorTouch : 1;               // 0x0290
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                    bShouldSaveForCheckpoint : 1;              // 0x0290 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
float                                            ChunkHealthScale;                          // 0x0294 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class UParticleSystem*>                   OverrideFragmentDestroyEffects;             // 0x0298
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                            FractureCullMinDistance;                   // 0x02A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                            FractureCullMaxDistance;                   // 0x02AC (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FDeferredPartToSpawn>              DeferredPartsToSpawn;                      // 0x02B0
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FPhysEffectInfo                           PartImpactEffect;                          // 0x02C0 (0x0018)
[0x0000000000000000]
class USoundCue*                                 ExplosionFractureSound;                    // 0x02D8 (0x0008)
[0x0000000000000000]
class USoundCue*                                 SingleChunkFractureSound;                  // 0x02E0
(0x0008) [0x0000000000000000]
class UMaterialInterface*                        MI_LoseChunkPreviousMaterial;              // 0x02E8
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FracturedStaticMeshActor");
}

return uClassPointer;
};

void eventSetLoseChunkReplacementMaterial();
void eventHideFragmentsToMaximizeMemoryUsage();
void eventHideOneFragment();
void eventResetVisibility();
void eventBreakOffPartsInRadius(struct FVector Origin, float Radius, float RBStrength, unsigned
long bWantPhysChunksAndParticles);
void eventExplode();
void RemoveDecals(int32_t IndexToRemoveDecalsFrom);
bool FractureEffectIsRelevant(unsigned long bForceDedicated, class APawn* EffectInstigator,
uint8_t& bWantPhysChunksAndParticles);
bool eventSpawnDeferredParts();
void eventBreakOffIsolatedIslands(TArray<int32_t> IgnoreFrags, struct FVector ChunkDir,
TArray<class AFracturedStaticMeshPart*> DisableCollWithPart, unsigned long
bWantPhysChunks, TArray<uint8_t>& FragmentVis);
void ApplyCheckpointRecord(struct AFracturedStaticMeshActor_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct AFracturedStaticMeshActor_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
void ResetHealth();
void eventPostBeginPlay();
class AFracturedStaticMeshPart* SpawnPartMulti(TArray<int32_t> ChunkIndices, struct FVector
InitialVel, struct FVector InitialAngVel, float RelativeScale, unsigned long bExplosion);
class AFracturedStaticMeshPart* SpawnPart(int32_t ChunkIndex, struct FVector InitialVel, struct
FVector InitialAngVel, float RelativeScale, unsigned long bExplosion);
};

// Class Engine.FracturedStaticMeshPart
// 0x0038 (0x02F0 - 0x0328)
class AFracturedStaticMeshPart : public AFracturedStaticMeshActor
{
public:
float                          DestroyPartRadiusFactor;              // 0x02F0 (0x0004)
[0x0000000000000000]
class AFracturedStaticMeshActor*           BaseFracturedMeshActor;              // 0x02F8
(0x0008) [0x0000000000002000] (CPF_Transient)
unsigned long                  bHasBeenRecycled : 1;                 // 0x0300 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                  bChangeRBChannelWhenAsleep : 1;            // 0x0300
(0x0004) [0x0000000000000000] [0x00000002]
unsigned long                  bCompositeThatExplodesOnImpact : 1;         // 0x0300
(0x0004) [0x0000000000000000] [0x00000004]
float                          LastSpawnTime;                        // 0x0304 (0x0004)
[0x0000000000000000]
int32_t                         PartPoolIndex;                       // 0x0308 (0x0004)
[0x0000000000000000]
float                          FracPartGravScale;                    // 0x030C (0x0004)
[0x0000000000000000]
uint8_t                         AsleepRBChannel;                     // 0x0310 (0x0001)
[0x0000000000000000]
struct FVector                    OldVelocity;                       // 0x0314 (0x000C)
[0x0000000000000000]
float                          CurrentVibrationLevel;                // 0x0320 (0x0004)
[0x0000000000000000]
float                          LastImpactSoundTime;                  // 0x0324 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FracturedStaticMeshPart");
}

return uClassPointer;
};

void eventBreakOffPartsInRadius(struct FVector Origin, float Radius, float RBStrength, unsigned
long bWantPhysChunksAndParticles);
void eventExplode();
void eventFellOutOfWorld();
void TryToCleanUp();
void RecyclePart(unsigned long bAddToFreePool);
void Initialize();
};

// Class Engine.FractureManager
// 0x0048 (0x0268 - 0x02B0)
class AFractureManager : public AActor
{
public:
int32_t                              FSMPartPoolSize;                         // 0x0268 (0x0004)
[0x0000000000000000]
unsigned long                        bEnableAntiVibration : 1;                // 0x026C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bEnableSpawnChunkEffectForRadialDamage : 1;    // 0x026C
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
float                                DestroyVibrationLevel;                   // 0x0270 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                DestroyMinAngVel;                        // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                ExplosionVelScale;                       // 0x0278 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class AFracturedStaticMeshPart*>          PartPool;                              // 0x0280
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<int32_t>                      FreeParts;                               // 0x0290 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class AFracturedStaticMeshActor*>         ActorsWithDeferredPartsToSpawn;              //
0x02A0 (0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FractureManager");
}

return uClassPointer;
};

void Tick(float DeltaTime);
void eventReturnPartActor(class AFracturedStaticMeshPart* Part);
class AFracturedStaticMeshPart* eventSpawnPartActor(class AFracturedStaticMeshActor*
Parent, struct FVector SpawnLocation, struct FRotator SpawnRotation);
class AFracturedStaticMeshPart* GetFSMPart(class AFracturedStaticMeshActor* Parent, struct
FVector SpawnLocation, struct FRotator SpawnRotation);
void ResetPoolVisibility();
void CreateFSMParts();
void CleanUpFSMParts();
void eventDestroyed();
void eventPreBeginPlay();
float GetFSMFractureCullDistanceScale();
float GetFSMRadialSpawnChanceScale();
float GetFSMDirectSpawnChanceScale();
float GetNumFSMPartsScale();
void eventSpawnChunkDestroyEffect(class UParticleSystem* Effect, struct FBox ChunkBox,
struct FVector ChunkDir, float Scale);
};

// Class Engine.ImageReflection
// 0x0018 (0x0268 - 0x0280)
class AImageReflection : public AActor
{
public:
unsigned long                              bEnabled : 1;                             // 0x0268 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)
class UImageReflectionComponent*                ReflectionComponent;                        // 0x0270
(0x0008) [0x0000000024080008] (CPF_ExportObject | CPF_Component | CPF_EditInline |
CPF_Deprecated)
class UImageBasedReflectionComponent*            ImageReflectionComponent;                    //
0x0278 (0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ImageReflection");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.ImageReflectionSceneCapture
// 0x0008 (0x0280 - 0x0288)
class AImageReflectionSceneCapture : public AImageReflection
{
public:
float                                      DepthRange;                               // 0x0280 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                                   ColorRange;                             // 0x0284 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ImageReflectionSceneCapture");
}

return uClassPointer;
};

};

// Class Engine.ImageReflectionShadowPlane
// 0x0010 (0x0268 - 0x0278)
class AImageReflectionShadowPlane : public AActor
{
public:
unsigned long                           bEnabled : 1;                           // 0x0268 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)
class UImageReflectionShadowPlaneComponent*
ReflectionShadowComponent;                  // 0x0270 (0x0008) [0x0000000004080009] (CPF_Edit
| CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ImageReflectionShadowPlane");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.ImageReflectionComponent
// 0x000B (0x009D - 0x00A8)
class UImageReflectionComponent : public UActorComponent
{
public:
class UTexture2D*                       ReflectionTexture;                      // 0x00A0 (0x0008)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ImageReflectionComponent");
}

return uClassPointer;
};

};

// Class Engine.ImageReflectionShadowPlaneComponent
// 0x0018 (0x0258 - 0x0270)
class UImageReflectionShadowPlaneComponent : public UPrimitiveComponent
{
public:
unsigned long                    bEnabled : 1;                    // 0x0258 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FPlane                    ReflectionPlane;                 // 0x0260 (0x0010)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ImageReflectionShadowPlaneComponent");
}

return uClassPointer;
};

void SetEnabled(unsigned long bSetEnabled);
};

// Class Engine.ApexComponentBase
// 0x0020 (0x0280 - 0x02A0)
class UApexComponentBase : public UMeshComponent
{
public:
struct FPointer                  ComponentBaseResources;          // 0x0280 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FRenderCommandFence_Mirror            ReleaseResourcesFence;            //
0x0288 (0x0004) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
class UApexAsset*                Asset;                           // 0x0290 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
```

```
struct FColor                                   WireframeColor;                              // 0x0298 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                                   bAssetChanged : 1;                           // 0x029C (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexComponentBase");
}

return uClassPointer;
};

};

// Class Engine.ApexDynamicComponent
// 0x0008 (0x02A0 - 0x02A8)
class UApexDynamicComponent : public UApexComponentBase
{
public:
struct FPointer                                 ComponentDynamicResources;                   // 0x02A0
(0x0008) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexDynamicComponent");
}

return uClassPointer;
};

};

// Class Engine.ApexStaticComponent
// 0x0000 (0x02A0 - 0x02A0)
class UApexStaticComponent : public UApexComponentBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexStaticComponent");
}

return uClassPointer;
};

};

// Class Engine.ApexStaticDestructibleComponent
// 0x001C (0x02A0 - 0x02BC)
class UApexStaticDestructibleComponent : public UApexStaticComponent
{
public:
float                                   SleepEnergyThreshold;                      // 0x02A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   SleepDamping;                         // 0x02A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                         ApexDestructibleActor;                    // 0x02A8 (0x0008)
[0x0000000000201000] (CPF_Native)
struct FPointer                         ApexDestructiblePreview;                  // 0x02B0 (0x0008)
[0x0000000000201000] (CPF_Native)
unsigned long                           bIsThumbnailComponent : 1;                // 0x02B8 (0x0004)
[0x0000000000001000] [0x00000001] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexStaticDestructibleComponent");
}

return uClassPointer;
};

};

// Class Engine.FracturedBaseComponent
// 0x002C (0x0300 - 0x032C)
class UFracturedBaseComponent : public UStaticMeshComponent
{
public:
struct FPointer                         ComponentBaseResources;                   // 0x0300 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FRenderCommandFence_Mirror               ReleaseResourcesFence;                    //
0x0308 (0x0004) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<uint8_t>                         VisibleFragments;                      // 0x0310 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
unsigned long                           bVisibilityHasChanged : 1;                // 0x0320 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
```

```
unsigned long                        bVisibilityReset : 1;                    // 0x0320 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                        bInitialVisibilityValue : 1;              // 0x0320 (0x0004)
[0x0000000000000002] [0x00000004] (CPF_Const)
unsigned long                        bUseDynamicIndexBuffer : 1;               // 0x0320 (0x0004)
[0x0000000000000002] [0x00000008] (CPF_Const)
unsigned long                        bUseDynamicIBWithHiddenFragments : 1;        // 0x0320
(0x0004) [0x0000000000000002] [0x00000010] (CPF_Const)
int32_t                         NumResourceIndices;                       // 0x0324 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         bResetStaticMesh;                       // 0x0328 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FracturedBaseComponent");
}

return uClassPointer;
};

int32_t GetNumVisibleFragments();
int32_t GetNumFragments();
bool IsFragmentVisible(int32_t FragmentIndex);
TArray<uint8_t> GetVisibleFragments();
bool SetStaticMesh(class UStaticMesh* NewMesh, unsigned long bForce);
};

// Class Engine.FracturedSkinnedMeshComponent
// 0x0030 (0x032C - 0x035C)
class UFracturedSkinnedMeshComponent : public UFracturedBaseComponent
{
public:
struct FPointer                      ComponentSkinResources;                 // 0x0330 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FMatrix>                    FragmentTransforms;                    // 0x0338 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class UFracturedStaticMeshComponent*>      DependentComponents;                     //
0x0348 (0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
unsigned long                        bBecameVisible : 1;                      // 0x0358 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                        bFragmentTransformsChanged : 1;             // 0x0358
(0x0004) [0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FracturedSkinnedMeshComponent");
}

return uClassPointer;
};

};

// Class Engine.FracturedStaticMeshComponent
// 0x0054 (0x032C - 0x0380)
class UFracturedStaticMeshComponent : public UFracturedBaseComponent
{
public:
TArray<uint8_t>                    FragmentNeighborsVisible;              // 0x0330 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
struct FBox                        VisibleBox;                            // 0x0340 (0x001C)
[0x0000000000000002] (CPF_Const)
unsigned long                      bUseSkinnedRendering : 1;              // 0x035C (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                      bUseVisibleVertsForBounds : 1;         // 0x035C (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                      bTopFragmentsRootNonDestroyable : 1;       // 0x035C
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                      bBottomFragmentsRootNonDestroyable : 1;    // 0x035C
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
float                              TopBottomFragmentDistThreshold;        // 0x0360 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              LoseChunkOutsideMaterialOverride;         // 0x0368
(0x0008) [0x0000000000000001] (CPF_Edit)
float                              FragmentBoundsMaxZ;                    // 0x0370 (0x0004)
[0x0000000000000000]
float                              FragmentBoundsMinZ;                    // 0x0374 (0x0004)
[0x0000000000000000]
class UFracturedSkinnedMeshComponent*        SkinnedComponent;                    //
0x0378 (0x0008) [0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FracturedStaticMeshComponent");
}

return uClassPointer;
};

class UPhysicalMaterial* GetFracturedMeshPhysMaterial();
```

```
void RecreatePhysState();
TArray<int32_t> GetBoundaryHiddenFragments(TArray<int32_t> AdditionalVisibleFragments);
TArray<struct FFragmentGroup> GetFragmentGroups(TArray<int32_t> IgnoreFragments, float
MinConnectionArea);
int32_t GetCoreFragmentIndex();
struct FVector GetFragmentAverageExteriorNormal(int32_t FragmentIndex);
struct FBox GetFragmentBox(int32_t FragmentIndex);
bool IsNoPhysFragment(int32_t FragmentIndex);
bool IsRootFragment(int32_t FragmentIndex);
bool IsFragmentDestroyable(int32_t FragmentIndex);
void SetVisibleFragments(TArray<uint8_t> VisibilityFactors);
};

// Class Engine.ImageBasedReflectionComponent
// 0x0020 (0x0300 - 0x0320)
class UImageBasedReflectionComponent : public UStaticMeshComponent
{
public:
unsigned long                         bEnabled : 1;                       // 0x0300 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bTwoSided : 1;                       // 0x0300 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
class UTexture2D*                        ReflectionTexture;                      // 0x0308 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor                      ReflectionColor;                      // 0x0310 (0x0010)
[0x0000000200000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ImageBasedReflectionComponent");
}

return uClassPointer;
};

void OnUpdatePropertyReflectionColor();
void UpdateImageReflectionParameters();
void SetEnabled(unsigned long bSetEnabled);
};

// Class Engine.InstancedStaticMeshComponent
// 0x006C (0x0300 - 0x036C)
class UInstancedStaticMeshComponent : public UStaticMeshComponent
{
public:
TArray<struct FInstancedStaticMeshInstanceData>    PerInstanceData;                    //
0x0300 (0x0010) [0x0000000020400000] (CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FInstancedStaticMeshInstanceData>    PerInstanceSMData;                    //
0x0310 (0x0010) [0x0000000000001000] (CPF_Native)
```

```
int32_t                              NumPendingLightmaps;                         // 0x0320 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                              ComponentJoinKey;                            // 0x0324 (0x0004)
[0x0000000000000000]
TArray<struct FInstancedStaticMeshMappingInfo>     CachedMappings;                             //
0x0328 (0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                              InstancingRandomSeed;                        // 0x0338 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                              InstanceStartCullDistance;                   // 0x033C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                              InstanceEndCullDistance;                     // 0x0340 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FBitArray_Mirror                       SelectedInstances;                        // 0x0348 (0x0020)
[0x0000000800001002] (CPF_Const | CPF_Native)
unsigned long                             bDontResolveInstancedLightmaps : 1;           // 0x0368
(0x0004) [0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InstancedStaticMeshComponent");
}

return uClassPointer;
};

};

// Class Engine.SplineMeshComponent
// 0x0068 (0x0300 - 0x0368)
class USplineMeshComponent : public UStaticMeshComponent
{
public:
struct FSplineMeshParams                      SplineParams;                               // 0x0300 (0x0058)
[0x0000000000000000]
struct FVector                            SplineXDir;                                 // 0x0358 (0x000C)
[0x0000000000000000]
unsigned long                             bSmoothInterpRollScale : 1;                 // 0x0364 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SplineMeshComponent");
}
return uClassPointer;
};

};

// Class Engine.ApexAsset
// 0x0050 (0x0060 - 0x00B0)
class UApexAsset : public UObject
{
public:
class FString                         OriginalApexName;                  // 0x0060 (0x0010)
[0x0000000004400002] (CPF_Const | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UApexComponentBase*>              ApexComponents;                    // 0x0070
(0x0010) [0x000000000408300A] (CPF_Const | CPF_ExportObject | CPF_Native | CPF_Transient |
CPF_Component | CPF_EditInline)
TArray<class UApexAsset*>                  NamedReferences;                   // 0x0080
(0x0010) [0x0000000814400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink | CPF_EditInline |
CPF_EditInlineUse)
class FString                         SourceFilePath;                    // 0x0090 (0x0010)
[0x0000000800420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)
class FString                         SourceFileTimestamp;               // 0x00A0 (0x0010)
[0x0000000800420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexAsset");
}

return uClassPointer;
};

};

// Class Engine.ApexClothingAsset
// 0x0078 (0x00B0 - 0x0128)
class UApexClothingAsset : public UApexAsset
{
public:
TArray<struct FClothingLodInfo>              LodMaterialInfo;                   // 0x00B0 (0x0010)
[0x0000000000400043] (CPF_Edit | CPF_Const | CPF_EditConstArray | CPF_NeedCtorLink)
struct FPointer                        MApexAsset;                        // 0x00C0 (0x0008)
[0x0000000000001000] (CPF_Native)
TArray<class UMaterialInterface*>              Materials;                         // 0x00C8 (0x0010)
[0x0000000000400043] (CPF_Edit | CPF_Const | CPF_EditConstArray | CPF_NeedCtorLink)
class UApexGenericAsset*                  ApexClothingLibrary;               // 0x00D8
(0x0008) [0x0000000020000002] (CPF_Const | CPF_Deprecated)
unsigned long                          bUseHardwareCloth : 1;             // 0x00E0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                          bFallbackSkinning : 1;             // 0x00E0 (0x0004)
```

```
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                    bSlowStart : 1;                    // 0x00E0 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                    bRecomputeNormals : 1;                    // 0x00E0 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                    bAllowAdaptiveTargetFrequency : 1;            // 0x00E0
(0x0004) [0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                    bResetAfterTeleport : 1;                    // 0x00E0 (0x0004)
[0x0000000000000003] [0x00000020] (CPF_Edit | CPF_Const)
unsigned long                    bUseLocalSpaceSimulation : 1;                // 0x00E0 (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                    bHasUniqueAssetMaterialNames : 1;            // 0x00E0
(0x0004) [0x0000000000000000] [0x00000080]
unsigned long                    IgnoreInitialTrigger : 1;                   // 0x00E0 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
int32_t                          UVChannelForTangentUpdate;                  // 0x00E4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            MaxDistanceBlendTime;                       // 0x00E8 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            ContinuousRotationThreshold;                // 0x00EC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            ContinuousDistanceThreshold;                // 0x00F0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            LodWeightsMaxDistance;                      // 0x00F4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            LodWeightsDistanceWeight;                   // 0x00F8 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            LodWeightsBias;                             // 0x00FC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            LodWeightsBenefitsBias;                     // 0x0100 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                            LODDecayTime;                               // 0x0104 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class USoundCue*                 SoundOnMove;                                // 0x0108 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                 SoundOnRest;                                // 0x0110 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                 SoundWhileMoving;                           // 0x0118 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            SpeedThresholdOnMove;                       // 0x0120 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            SpeedThresholdOnRest;                       // 0x0124 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexClothingAsset");
}
return uClassPointer;
};

};

// Class Engine.ApexDestructibleAsset
// 0x0150 (0x00B0 - 0x0200)
class UApexDestructibleAsset : public UApexAsset
{
public:
struct FPointer                                   MApexAsset;                                  // 0x00B0 (0x0008)
[0x0000000000001000] (CPF_Native)
TArray<class UMaterialInterface*>                 Materials;                                   // 0x00B8 (0x0010)
[0x0000000000400043] (CPF_Edit | CPF_Const | CPF_EditConstArray | CPF_NeedCtorLink)
TArray<class UFractureMaterial*>                  FractureMaterials;                           // 0x00C8
(0x0010) [0x0000000000400043] (CPF_Edit | CPF_Const | CPF_EditConstArray |
CPF_NeedCtorLink)
unsigned long                                     bPlaySingleFractureMaterialEffect : 1;       // 0x00D8
(0x0004) [0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                                     bHasUniqueAssetMaterialNames : 1;            // 0x00D8
(0x0004) [0x0000000000000000] [0x00000002]
unsigned long                                     bDynamic : 1;                                // 0x00D8 (0x0004)
[0x0000000020000000] [0x00000004]  CPF_Deprecated)
class UPhysicalMaterial*                          DefaultPhysMaterial;                         // 0x00E0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FPointer                                   MDestructibleThumbnailComponent;             // 0x00E8
(0x0008) [0x0000000000001000] (CPF_Native)
class FString                                     CrumbleEmitterName;                          // 0x00F0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                                     DustEmitterName;                             // 0x0100 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FNxDestructibleParameters                  DestructibleParameters;                      // 0x0110
(0x00F0) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexDestructibleAsset");
}

return uClassPointer;
};

};

// Class Engine.ApexGenericAsset
// 0x0018 (0x00B0 - 0x00C8)
class UApexGenericAsset : public UApexAsset
{
public:
```

```
struct FPointer                              MApexAsset;                              // 0x00B0 (0x0008)
[0x0000000000001000] (CPF_Native)
TArray<class UMaterialInterface*>                  Materials;                              // 0x00B8 (0x0010)
[0x0000000800400043] (CPF_Edit | CPF_Const | CPF_EditConstArray | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexGenericAsset");
}

return uClassPointer;
};

};

// Class Engine.InterpFilter
// 0x0010 (0x0060 - 0x0070)
class UInterpFilter : public UObject
{
public:
class FString                              Caption;                              // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpFilter");
}

return uClassPointer;
};

};

// Class Engine.InterpFilter_Classes
// 0x0018 (0x0070 - 0x0088)
class UInterpFilter_Classes : public UInterpFilter
{
public:
class UClass*                              ClassToFilterBy;                              // 0x0070 (0x0008)
[0x0000000800000000]
TArray<class UClass*>                              TrackClasses;                              // 0x0078 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpFilter_Classes");
}

return uClassPointer;
};

};

// Class Engine.InterpFilter_Custom
// 0x0010 (0x0070 - 0x0080)
class UInterpFilter_Custom : public UInterpFilter
{
public:
TArray<class UInterpGroup*>            GroupsToInclude;                // 0x0070 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpFilter_Custom");
}

return uClassPointer;
};

};

// Class Engine.InterpGroup
// 0x003C (0x0060 - 0x009C)
class UInterpGroup : public UObject
{
public:
struct FPointer                    VfTable_FInterpEdInputInterface;        // 0x0060 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
TArray<class UInterpTrack*>            InterpTracks;                   // 0x0068 (0x0010)
[0x0000000000400008] (CPF_ExportObject | CPF_NeedCtorLink)
struct FName                       GroupName;                       // 0x0078 (0x0008)
[0x0000000000000000]
struct FColor                      GroupColor;                      // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class UAnimSet*>                GroupAnimSets;                  // 0x0088 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                      bCollapsed : 1;                  // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]
```

```
unsigned long                                  bVisible : 1;                            // 0x0098 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                                  bIsFolder : 1;                           // 0x0098 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                                  bIsParented : 1;                         // 0x0098 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                                  bIsSelected : 1;                         // 0x0098 (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroup");
}

return uClassPointer;
};

};

// Class Engine.InterpGroupAI
// 0x0018 (0x009C - 0x00B4)
class UInterpGroupAI : public UInterpGroup
{
public:
class UClass*                                  PreviewPawnClass;                        // 0x00A0 (0x0008)
[0x0000000800000001] (CPF_Edit)
struct FName                                   StageMarkGroup;                          // 0x00A8 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                                  SnapToRootBoneLocationWhenFinished : 1;       // 0x00B0
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                  bNoEncroachmentCheck : 1;                // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                  bDisableWorldCollision : 1;              // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                  bIgnoreLegacyHeightAdjust : 1;           // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                                  bRecreatePreviewPawn : 1;                // 0x00B0 (0x0004)
[0x0000000800002000] [0x00000010] (CPF_Transient)
unsigned long                                  bRefreshStageMarkGroup : 1;              // 0x00B0 (0x0004)
[0x0000000800002000] [0x00000020] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroupAI");
}

return uClassPointer;
};

};

// Class Engine.InterpGroupCamera
// 0x0050 (0x009C - 0x00EC)
class UInterpGroupCamera : public UInterpGroup
{
public:
class UCameraAnim*                    CameraAnimInst;                    // 0x00A0 (0x0008)
[0x0000000000002000] (CPF_Transient)
struct FCameraPreviewInfo                 Target;                          // 0x00A8 (0x0040)
[0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
float                             CompressTolerance;                  // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroupCamera");
}

return uClassPointer;
};

};

// Class Engine.InterpGroupDirector
// 0x0004 (0x009C - 0x00A0)
class UInterpGroupDirector : public UInterpGroup
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroupDirector");
}

return uClassPointer;
};

};
```

```
// Class Engine.InterpGroupInst
// 0x0028 (0x0060 - 0x0088)
class UInterpGroupInst : public UObject
{
public:
class UInterpGroup*                    Group;                              // 0x0060 (0x0008)
[0x0000000000000000]
class AActor*                       GroupActor;                         // 0x0068 (0x0008)
[0x0000000000000000]
TArray<class UInterpTrackInst*>              TrackInst;                           // 0x0070 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FPointer                     CachedCamOverridePostProcess;              // 0x0080
(0x0008) [0x0000000000003000] (CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroupInst");
}

return uClassPointer;
};

};

// Class Engine.InterpGroupInstAI
// 0x0020 (0x0088 - 0x00A8)
class UInterpGroupInstAI : public UInterpGroupInst
{
public:
class UInterpGroupAI*                   AIGroup;                            // 0x0088 (0x0008)
[0x0000000000002000] (CPF_Transient)
uint8_t                          SavedPhysics;                        // 0x0090 (0x0001)
[0x0000000000000000]
unsigned long                       bSavedNoEncroachCheck : 1;                // 0x0094 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                       bSavedCollideActors : 1;                 // 0x0094 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                       bSavedBlockActors : 1;                   // 0x0094 (0x0004)
[0x0000000000000000] [0x00000004]
class APawn*                        PreviewPawn;                         // 0x0098 (0x0008)
[0x0000000800002000] (CPF_Transient)
class AActor*                       StageMarkActor;                       // 0x00A0 (0x0008)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroupInstAI");
}

return uClassPointer;
};

};

// Class Engine.InterpGroupInstCamera
// 0x0000 (0x0088 - 0x0088)
class UInterpGroupInstCamera : public UInterpGroupInst
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroupInstCamera");
}

return uClassPointer;
};

};

// Class Engine.InterpGroupInstDirector
// 0x0000 (0x0088 - 0x0088)
class UInterpGroupInstDirector : public UInterpGroupInst
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpGroupInstDirector");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackBoolProp
// 0x001C (0x00C4 - 0x00E0)
class UInterpTrackBoolProp : public UInterpTrack
{
public:
TArray<struct FBoolTrackKey>                BoolTrack;                          // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FName                                PropertyName;                       // 0x00D8 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackBoolProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackDirector
// 0x0018 (0x00C4 - 0x00DC)
class UInterpTrackDirector : public UInterpTrack
{
public:
TArray<struct FDirectorTrackCut>            CutTrack;                           // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                               bSimulateCameraCutsOnClients : 1;           // 0x00D8
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackDirector");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackEvent
// 0x0018 (0x00C4 - 0x00DC)
class UInterpTrackEvent : public UInterpTrack
{
public:
```

```
TArray<struct FEventTrackKey>                    EventTrack;                                  // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                                    bFireEventsWhenForwards : 1;                 // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                    bFireEventsWhenBackwards : 1;                // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                    bFireEventsWhenJumpingForwards : 1;          // 0x00D8
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackEvent");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackFaceFX
// 0x003C (0x00C4 - 0x0100)
class UInterpTrackFaceFX : public UInterpTrack
{
public:
TArray<class UFaceFXAnimSet*>                    FaceFXAnimSets;                              // 0x00C8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FFaceFXTrackKey>                   FaceFXSeqs;                                  // 0x00D8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class UFaceFXAsset*                              CachedActorFXAsset;                          // 0x00E8 (0x0008)
[0x0000000000002000] (CPF_Transient)
TArray<struct FFaceFXSoundCueKey>                FaceFXSoundCueKeys;                          // 0x00F0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackFaceFX");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackFloatBase
// 0x0020 (0x00C4 - 0x00E4)
class UInterpTrackFloatBase : public UInterpTrack
{
public:
struct FInterpCurveFloat                    FloatTrack;                             // 0x00C8 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                    CurveTension;                           // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackFloatBase");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackAnimControl
// 0x0030 (0x00E4 - 0x0114)
class UInterpTrackAnimControl : public UInterpTrackFloatBase
{
public:
TArray<class UAnimSet*>                     AnimSets;                               // 0x00E8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FName                             SlotName;                               // 0x00F8 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<struct FAnimControlTrackKey>            AnimSeqs;                               // 0x0100
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                            bEnableRootMotion : 1;                  // 0x0110 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bSkipAnimNotifiers : 1;                 // 0x0110 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackAnimControl");
}

return uClassPointer;
};

};
```

```
// Class Engine.InterpTrackFade
// 0x0008 (0x00E4 - 0x00EC)
class UInterpTrackFade : public UInterpTrackFloatBase
{
public:
unsigned long                        bPersistFade : 1;                    // 0x00E8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackFade");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackFloatMaterialParam
// 0x0028 (0x00E4 - 0x010C)
class UInterpTrackFloatMaterialParam : public UInterpTrackFloatBase
{
public:
TArray<struct FMaterialReferenceList>          Materials;                         // 0x00E8 (0x0010)
[0x0000000000480003] (CPF_Edit | CPF_Const | CPF_Component | CPF_NeedCtorLink)
class UMaterialInterface*                  Material;                          // 0x00F8 (0x0008)
[0x0000000020000002] (CPF_Const | CPF_Deprecated)
struct FName                         ParamName;                         // 0x0100 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                        bNeedsMaterialRefsUpdate : 1;              // 0x0108 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackFloatMaterialParam");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackFloatParticleParam
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackFloatParticleParam : public UInterpTrackFloatBase
{
public:
struct FName                              ParamName;                              // 0x00E8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackFloatParticleParam");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackFloatProp
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackFloatProp : public UInterpTrackFloatBase
{
public:
struct FName                              PropertyName;                           // 0x00E8 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackFloatProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackMorphWeight
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackMorphWeight : public UInterpTrackFloatBase
{
public:
struct FName                              MorphNodeName;                          // 0x00E8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackMorphWeight");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackMoveAxis
// 0x001C (0x00E4 - 0x0100)
class UInterpTrackMoveAxis : public UInterpTrackFloatBase
{
public:
uint8_t                                 MoveAxis;                          // 0x00E8 (0x0001)
[0x0000000000000000]
struct FInterpLookupTrack               LookupTrack;                       // 0x00F0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackMoveAxis");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackSkelControlScale
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackSkelControlScale : public UInterpTrackFloatBase
{
public:
struct FName                            SkelControlName;                   // 0x00E8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.InterpTrackSkelControlScale");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackSkelControlStrength
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackSkelControlStrength : public UInterpTrackFloatBase
{
public:
struct FName                    SkelControlName;                    // 0x00E8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackSkelControlStrength");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackSlomo
// 0x0004 (0x00E4 - 0x00E8)
class UInterpTrackSlomo : public UInterpTrackFloatBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackSlomo");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackHeadTracking
// 0x005C (0x00C4 - 0x0120)
```

```
class UInterpTrackHeadTracking : public UInterpTrack
{
public:
TArray<struct FHeadTrackingKey>                HeadTrackingTrack;                        // 0x00C8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FName>                        TrackControllerName;                     // 0x00D8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                    LookAtActorRadius;                        // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bDisableBeyondLimit : 1;                   // 0x00EC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bLookAtPawns : 1;                          // 0x00EC (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                                    MaxLookAtTime;                             // 0x00F0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    MinLookAtTime;                             // 0x00F4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    MaxInterestTime;                           // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class UClass*>                       ActorClassesToLookAt;                     // 0x0100 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FName>                        TargetBoneNames;                          // 0x0110 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackHeadTracking");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackLinearColorBase
// 0x0020 (0x00C4 - 0x00E4)
class UInterpTrackLinearColorBase : public UInterpTrack
{
public:
struct FInterpCurveLinearColor                LinearColorTrack;                         // 0x00C8 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                    CurveTension;                              // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackLinearColorBase");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackLinearColorProp
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackLinearColorProp : public UInterpTrackLinearColorBase
{
public:
struct FName                            PropertyName;                          // 0x00E8 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackLinearColorProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackMove
// 0x005A (0x00C4 - 0x011E)
class UInterpTrackMove : public UInterpTrack
{
public:
struct FInterpCurveVector                    PosTrack;                              // 0x00C8 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FInterpCurveVector                    EulerTrack;                            // 0x00E0 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FInterpLookupTrack                    LookupTrack;                           // 0x00F8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FName                            LookAtGroupName;                       // 0x0108 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                               LinCurveTension;                       // 0x0110 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               AngCurveTension;                       // 0x0114 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bUseQuatInterpolation : 1;             // 0x0118 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bShowArrowAtKeys : 1;                  // 0x0118 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
```

```
unsigned long                              bDisableMovement : 1;                        // 0x0118 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                              bShowTranslationOnCurveEd : 1;               // 0x0118 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                              bShowRotationOnCurveEd : 1;                  // 0x0118 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                              bHide3DTrack : 1;                            // 0x0118 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                              bUseRawActorTMforRelativeToInitial : 1;      // 0x0118
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
uint8_t                                MoveFrame;                               // 0x011C (0x0001)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
uint8_t                                RotMode;                                 // 0x011D (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackMove");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackNotify
// 0x0034 (0x00C4 - 0x00F8)
class UInterpTrackNotify : public UInterpTrack
{
public:
class UAnimNodeSequence*                      Node;                                    // 0x00C8 (0x0008)
[0x0000000000000000]
struct FName                               ParentNodeName;                          // 0x00D0 (0x0008)
[0x0000000000000000]
class UAnimSequence*                          OuterSequence;                           // 0x00D8 (0x0008)
[0x0000000000000000]
class UAnimSet*                            OuterSet;                                // 0x00E0 (0x0008)
[0x0000000000000000]
TArray<struct FNotifyTrackKey>                NotifyTrack;                             // 0x00E8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackNotify");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackParticleReplay
// 0x001C (0x00C4 - 0x00E0)
class UInterpTrackParticleReplay : public UInterpTrack
{
public:
TArray<struct FParticleReplayTrackKey>          TrackKeys;                              // 0x00C8
(0x0010) [0x0000000004400000] (CPF_NeedCtorLink | CPF_EditInline)
unsigned long                                   bIsCapturingReplay : 1;                 // 0x00D8 (0x0004)
[0x0000000800002002] [0x00000001] (CPF_Const | CPF_Transient)
float                                           FixedTimeStep;                          // 0x00DC (0x0004)
[0x0000000800002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackParticleReplay");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackToggle
// 0x0018 (0x00C4 - 0x00DC)
class UInterpTrackToggle : public UInterpTrack
{
public:
TArray<struct FToggleTrackKey>                  ToggleTrack;                            // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                                   bActivateSystemEachUpdate : 1;          // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                   bActivateWithJustAttachedFlag : 1;      // 0x00D8
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                   bFireEventsWhenForwards : 1;            // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                   bFireEventsWhenBackwards : 1;           // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                                   bFireEventsWhenJumpingForwards : 1;     // 0x00D8
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackToggle");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackVectorBase
// 0x0020 (0x00C4 - 0x00E4)
class UInterpTrackVectorBase : public UInterpTrack
{
public:
struct FInterpCurveVector                VectorTrack;                        // 0x00C8 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)
float                               CurveTension;                        // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackVectorBase");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackAudioMaster
// 0x0004 (0x00E4 - 0x00E8)
class UInterpTrackAudioMaster : public UInterpTrackVectorBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackAudioMaster");
}
return uClassPointer;
};

};

// Class Engine.InterpTrackColorProp
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackColorProp : public UInterpTrackVectorBase
{
public:
struct FName                        PropertyName;                    // 0x00E8 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackColorProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackColorScale
// 0x0004 (0x00E4 - 0x00E8)
class UInterpTrackColorScale : public UInterpTrackVectorBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackColorScale");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackSound
// 0x0018 (0x00E4 - 0x00FC)
class UInterpTrackSound : public UInterpTrackVectorBase
{
public:
```

```
TArray<struct FSoundTrackKey>                  Sounds;                              // 0x00E8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                             bPlayOnReverse : 1;                  // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                             bContinueSoundOnMatineeEnd : 1;           // 0x00F8
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                             bSuppressSubtitles : 1;                // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                             bTreatAsDialogue : 1;                  // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackSound");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackVectorMaterialParam
// 0x0028 (0x00E4 - 0x010C)
class UInterpTrackVectorMaterialParam : public UInterpTrackVectorBase
{
public:
TArray<struct FMaterialReferenceList>          Materials;                           // 0x00E8 (0x0010)
[0x0000000000480003] (CPF_Edit | CPF_Const | CPF_Component | CPF_NeedCtorLink)
class UMaterialInterface*                      Material;                            // 0x00F8 (0x0008)
[0x0000000020000002] (CPF_Const | CPF_Deprecated)
struct FName                              ParamName;                           // 0x0100 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                             bNeedsMaterialRefsUpdate : 1;            // 0x0108 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackVectorMaterialParam");
}

return uClassPointer;
};

};
```

```
// Class Engine.InterpTrackVectorProp
// 0x000C (0x00E4 - 0x00F0)
class UInterpTrackVectorProp : public UInterpTrackVectorBase
{
public:
struct FName                              PropertyName;                             // 0x00E8 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackVectorProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackVisibility
// 0x0018 (0x00C4 - 0x00DC)
class UInterpTrackVisibility : public UInterpTrack
{
public:
TArray<struct FVisibilityTrackKey>              VisibilityTrack;                           // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                             bFireEventsWhenForwards : 1;                // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                             bFireEventsWhenBackwards : 1;               // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                             bFireEventsWhenJumpingForwards : 1;           // 0x00D8
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackVisibility");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInst
// 0x0000 (0x0060 - 0x0060)
class UInterpTrackInst : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInst");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstAnimControl
// 0x001C (0x0060 - 0x007C)
class UInterpTrackInstAnimControl : public UInterpTrackInst
{
public:
float                          LastUpdatePosition;            // 0x0060 (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FVector                 InitPosition;                  // 0x0064 (0x000C)
[0x0000000800002000] (CPF_Transient)
struct FRotator                InitRotation;                  // 0x0070 (0x000C)
[0x0000000800002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstAnimControl");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstAudioMaster
// 0x0000 (0x0060 - 0x0060)
class UInterpTrackInstAudioMaster : public UInterpTrackInst
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstAudioMaster");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstColorScale
// 0x0000 (0x0060 - 0x0060)
class UInterpTrackInstColorScale : public UInterpTrackInst
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstColorScale");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstDirector
// 0x000C (0x0060 - 0x006C)
class UInterpTrackInstDirector : public UInterpTrackInst
{
public:
class AActor*                    OldViewTarget;                 // 0x0060 (0x0008)
[0x0000000000000000]
struct FRenderingPerformanceOverrides       OldRenderingOverrides;           // 0x0068
(0x0004) [0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstDirector");
}
return uClassPointer;
};

};

// Class Engine.InterpTrackInstEvent
// 0x0004 (0x0060 - 0x0064)
class UInterpTrackInstEvent : public UInterpTrackInst
{
public:
float                                   LastUpdatePosition;                             // 0x0060 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstEvent");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstFaceFX
// 0x0008 (0x0060 - 0x0068)
class UInterpTrackInstFaceFX : public UInterpTrackInst
{
public:
unsigned long                           bFirstUpdate : 1;                               // 0x0060 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
float                                   LastUpdatePosition;                             // 0x0064 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstFaceFX");
}

return uClassPointer;
};

};
```

```
// Class Engine.InterpTrackInstFade
// 0x0000 (0x0060 - 0x0060)
class UInterpTrackInstFade : public UInterpTrackInst
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstFade");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstFloatMaterialParam
// 0x0018 (0x0060 - 0x0078)
class UInterpTrackInstFloatMaterialParam : public UInterpTrackInst
{
public:
TArray<struct FFloatMaterialParamMICData>        MICInfos;                              // 0x0060
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class UInterpTrackFloatMaterialParam*            InstancedTrack;                        // 0x0070
(0x0008) [0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstFloatMaterialParam");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstFloatParticleParam
// 0x0004 (0x0060 - 0x0064)
class UInterpTrackInstFloatParticleParam : public UInterpTrackInst
{
public:
float                                            ResetFloat;                            // 0x0060 (0x0004)
[0x0000000000000000]
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstFloatParticleParam");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstHeadTracking
// 0x0074 (0x0060 - 0x00D4)
class UInterpTrackInstHeadTracking : public UInterpTrackInst
{
public:
uint8_t                              Action;                             // 0x0060 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                              UnknownData00[0x50];                // 0x0068 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.InterpTrackInstHeadTracking.CurrentActorMap
class USkeletalMeshComponent*                 Mesh;                               // 0x00B8 (0x0008)
[0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component | CPF_EditInline)
TArray<class USkelControlLookAt*>             TrackControls;                      // 0x00C0
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                                LastUpdatePosition;                 // 0x00D0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstHeadTracking");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstMorphWeight
// 0x0000 (0x0060 - 0x0060)
class UInterpTrackInstMorphWeight : public UInterpTrackInst
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstMorphWeight");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstMove
// 0x0070 (0x0060 - 0x00D0)
class UInterpTrackInstMove : public UInterpTrackInst
{
public:
struct FVector                    ResetLocation;                    // 0x0060 (0x000C)
[0x0000000000000000]
struct FRotator                   ResetRotation;                    // 0x006C (0x000C)
[0x0000000000000000]
uint8_t                         UnknownData00[0x8];               // 0x0078 (0x0008) MISSED
OFFSET
struct FMatrix                    InitialTM;                     // 0x0080 (0x0040)
[0x0000000000000000]
struct FQuat                     InitialQuat;                   // 0x00C0 (0x0010)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstMove");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstNotify
// 0x0004 (0x0060 - 0x0064)
class UInterpTrackInstNotify : public UInterpTrackInst
{
public:
float                          LastUpdatePosition;               // 0x0060 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstNotify");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstParticleReplay
// 0x0004 (0x0060 - 0x0064)
class UInterpTrackInstParticleReplay : public UInterpTrackInst
{
public:
float                                   LastUpdatePosition;                    // 0x0060 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstParticleReplay");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstProperty
// 0x0010 (0x0060 - 0x0070)
class UInterpTrackInstProperty : public UInterpTrackInst
{
public:
class UFunction*                           PropertyUpdateCallback;                  // 0x0060 (0x0008)
[0x0000000000000000]
class UObject*                             PropertyOuterObjectInst;                 // 0x0068 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstProperty");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstBoolProp
// 0x0010 (0x0070 - 0x0080)
class UInterpTrackInstBoolProp : public UInterpTrackInstProperty
{
public:
struct FPointer                         BoolProp;                           // 0x0070 (0x0008)
[0x0000000000000000]
int32_t                                 BitMask;                            // 0x0078 (0x0004)
[0x0000000000000000]
unsigned long                           ResetBool : 1;                      // 0x007C (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstBoolProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstColorProp
// 0x000C (0x0070 - 0x007C)
class UInterpTrackInstColorProp : public UInterpTrackInstProperty
{
public:
struct FPointer                         ColorProp;                          // 0x0070 (0x0008)
[0x0000000000000000]
struct FColor                           ResetColor;                         // 0x0078 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstColorProp");
}
```

```
return uClassPointer;
};

};

// Class Engine.InterpTrackInstFloatProp
// 0x0018 (0x0070 - 0x0088)
class UInterpTrackInstFloatProp : public UInterpTrackInstProperty
{
public:
struct FPointer                    FloatProp;                    // 0x0070 (0x0008)
[0x0000000000000000]
float                          ResetFloat;                    // 0x0078 (0x0004)
[0x0000000000000000]
struct FPointer                    DistributionProp;                    // 0x0080 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstFloatProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstLinearColorProp
// 0x0018 (0x0070 - 0x0088)
class UInterpTrackInstLinearColorProp : public UInterpTrackInstProperty
{
public:
struct FPointer                    ColorProp;                    // 0x0070 (0x0008)
[0x0000000000000000]
struct FLinearColor                    ResetColor;                    // 0x0078 (0x0010)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstLinearColorProp");
}

return uClassPointer;
};
```

```
};

// Class Engine.InterpTrackInstVectorProp
// 0x0014 (0x0070 - 0x0084)
class UInterpTrackInstVectorProp : public UInterpTrackInstProperty
{
public:
struct FPointer                          VectorProp;                              // 0x0070 (0x0008)
[0x0000000000000000]
struct FVector                           ResetVector;                             // 0x0078 (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstVectorProp");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstSkelControlScale
// 0x0000 (0x0060 - 0x0060)
class UInterpTrackInstSkelControlScale : public UInterpTrackInst
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstSkelControlScale");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstSkelControlStrength
// 0x0004 (0x0060 - 0x0064)
class UInterpTrackInstSkelControlStrength : public UInterpTrackInst
{
public:
```

```
unsigned long                          bSavedControlledByAnimMetaData : 1;          // 0x0060
(0x0004) [0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstSkelControlStrength");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstSlomo
// 0x0004 (0x0060 - 0x0064)
class UInterpTrackInstSlomo : public UInterpTrackInst
{
public:
float                                  OldTimeDilation;                         // 0x0060 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstSlomo");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstSound
// 0x0010 (0x0060 - 0x0070)
class UInterpTrackInstSound : public UInterpTrackInst
{
public:
float                                  LastUpdatePosition;                      // 0x0060 (0x0004)
[0x0000000000000000]
class UAudioComponent*                 PlayAudioComp;                           // 0x0068 (0x0008)
[0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstSound");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstToggle
// 0x000C (0x0060 - 0x006C)
class UInterpTrackInstToggle : public UInterpTrackInst
{
public:
uint8_t                              Action;                              // 0x0060 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                LastUpdatePosition;                  // 0x0064 (0x0004)
[0x0000000000000000]
unsigned long                        bSavedActiveState : 1;               // 0x0068 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstToggle");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstVectorMaterialParam
// 0x0018 (0x0060 - 0x0078)
class UInterpTrackInstVectorMaterialParam : public UInterpTrackInst
{
public:
TArray<struct FVectorMaterialParamMICData>     MICInfos;                              // 0x0060
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class UInterpTrackVectorMaterialParam*         InstancedTrack;                        // 0x0070
(0x0008) [0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstVectorMaterialParam");
}

return uClassPointer;
};

};

// Class Engine.InterpTrackInstVisibility
// 0x0008 (0x0060 - 0x0068)
class UInterpTrackInstVisibility : public UInterpTrackInst
{
public:
uint8_t                        Action;                        // 0x0060 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                          LastUpdatePosition;                    // 0x0064 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpTrackInstVisibility");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpression
// 0x0060 (0x0060 - 0x00C0)
class UMaterialExpression : public UObject
{
public:
int32_t                        EditorX;                        // 0x0060 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
int32_t                        EditorY;                        // 0x0064 (0x0004)
[0x0000000020000000]  CPF_Deprecated)
int32_t                        MaterialExpressionEditorX;              // 0x0068 (0x0004)
[0x0000000800000000]
int32_t                        MaterialExpressionEditorY;              // 0x006C (0x0004)
[0x0000000800000000]
int32_t                        MaterialExpressionWidth;               // 0x0070 (0x0004)
[0x0000000800000000]
unsigned long                     bRealtimePreview : 1;               // 0x0074 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                     bNeedToUpdatePreview : 1;               // 0x0074 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
```

```
unsigned long                    bIsParameterExpression : 1;            // 0x0074 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                    bIsHighlighted : 1;                    // 0x0074 (0x0004)
[0x0000000800000000] [0x00000008]
unsigned long                    bShowOutputNameOnPin : 1;              // 0x0074 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                    bHidePreviewWindow : 1;                // 0x0074 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                    bShowInputs : 1;                       // 0x0074 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                    bShowOutputs : 1;                      // 0x0074 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                    bUsedByStaticParameterSet : 1;         // 0x0074 (0x0004)
[0x0000000000000000] [0x00000100]
class UMaterial*                 Material;                              // 0x0078 (0x0008)
[0x0000000000000002] (CPF_Const)
class UMaterialFunction*             Function;                          // 0x0080 (0x0008)
[0x0000000000000002] (CPF_Const)
class FString                    Desc;                                  // 0x0088 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FColor                    BorderColor;                           // 0x0098 (0x0004)
[0x0000000000000000]
TArray<struct FName>                 MenuCategories;                    // 0x00A0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FExpressionOutput>         Outputs;                       // 0x00B0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpression");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionAbs
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionAbs : public UMaterialExpression
{
public:
struct FExpressionInput              Input;                             // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionAbs");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionActorWorldPosition
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionActorWorldPosition : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionActorWorldPosition");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionAdd
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionAdd : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                              // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionAdd");
}

return uClassPointer;
};
```

```
};

// Class Engine.MaterialExpressionAppendVector
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionAppendVector : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                  // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionAppendVector");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionAtan
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionAtan : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionAtan");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionAtan2
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionAtan2 : public UMaterialExpression
```

```
{
public:
struct FExpressionInput                    A;                                    // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                    // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionAtan2");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionBumpOffset
// 0x00B0 (0x00C0 - 0x0170)
class UMaterialExpressionBumpOffset : public UMaterialExpression
{
public:
struct FExpressionInput                    Coordinate;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Height;                                // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    HeightRatioInput;                          // 0x0130 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                 HeightRatio;                            // 0x0168 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 ReferencePlane;                            // 0x016C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionBumpOffset");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionCameraVector
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionCameraVector : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionCameraVector");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionCameraWorldPosition
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionCameraWorldPosition : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionCameraWorldPosition");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionCeil
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionCeil : public UMaterialExpression
{
public:
struct FExpressionInput                    Input;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionCeil");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionClamp
// 0x00A8 (0x00C0 - 0x0168)
class UMaterialExpressionClamp : public UMaterialExpression
{
public:
struct FExpressionInput                    Input;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Min;                                // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Max;                                // 0x0130 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionClamp");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionComment
// 0x0020 (0x00C0 - 0x00E0)
class UMaterialExpressionComment : public UMaterialExpression
{
public:
int32_t                             PosX;                              // 0x00C0 (0x0004)
[0x0000000000000000]
int32_t                             PosY;                              // 0x00C4 (0x0004)
[0x0000000000000000]
int32_t                             SizeX;                             // 0x00C8 (0x0004)
[0x0000000000000000]
int32_t                             SizeY;                             // 0x00CC (0x0004)
[0x0000000000000000]
class FString                          Text;                             // 0x00D0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionComment");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionComponentMask
// 0x003C (0x00C0 - 0x00FC)
class UMaterialExpressionComponentMask : public UMaterialExpression
{
public:
struct FExpressionInput              Input;                    // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                    R : 1;                    // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    G : 1;                    // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    B : 1;                    // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    A : 1;                    // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionComponentMask");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionConstant
// 0x0004 (0x00C0 - 0x00C4)
class UMaterialExpressionConstant : public UMaterialExpression
{
public:
float                            R;                        // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionConstant");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionConstant2Vector
// 0x0008 (0x00C0 - 0x00C8)
class UMaterialExpressionConstant2Vector : public UMaterialExpression
{
public:
float                                  R;                                    // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  G;                                    // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionConstant2Vector");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionConstant3Vector
// 0x000C (0x00C0 - 0x00CC)
class UMaterialExpressionConstant3Vector : public UMaterialExpression
{
public:
float                                  R;                                    // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  G;                                    // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  B;                                    // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionConstant3Vector");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionConstant4Vector
// 0x0010 (0x00C0 - 0x00D0)
class UMaterialExpressionConstant4Vector : public UMaterialExpression
{
public:
float                                   R;                                      // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   G;                                      // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   B;                                      // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   A;                                      // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionConstant4Vector");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionConstantBiasScale
// 0x0040 (0x00C0 - 0x0100)
class UMaterialExpressionConstantBiasScale : public UMaterialExpression
{
public:
struct FExpressionInput                 Input;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                   Bias;                                   // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   Scale;                                  // 0x00FC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionConstantBiasScale");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionConstantClamp
// 0x0040 (0x00C0 - 0x0100)
class UMaterialExpressionConstantClamp : public UMaterialExpression
{
public:
struct FExpressionInput            Input;                      // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                          Min;                      // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          Max;                      // 0x00FC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionConstantClamp");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionCosine
// 0x003C (0x00C0 - 0x00FC)
class UMaterialExpressionCosine : public UMaterialExpression
{
public:
struct FExpressionInput            Input;                      // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                          Period;                     // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionCosine");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionCrossProduct
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionCrossProduct : public UMaterialExpression
{
public:
struct FExpressionInput                     A;                                     // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                     B;                                     // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionCrossProduct");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionCustom
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionCustom : public UMaterialExpression
{
public:
class FString                               Code;                              // 0x00C0 (0x0010)
[0x0000040000400001] (CPF_Edit | CPF_NeedCtorLink)
uint8_t                                   OutputType;                          // 0x00D0 (0x0001)
[0x0000000000000001] (CPF_Edit)
class FString                               Description;                         // 0x00D8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FCustomInput>                 Inputs;                                // 0x00E8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionCustom");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionCustomTexture
// 0x0008 (0x00C0 - 0x00C8)
class UMaterialExpressionCustomTexture : public UMaterialExpression
{
public:
class UTexture*                         Texture;                                // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionCustomTexture");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDepthBiasedAlpha
// 0x0078 (0x00C0 - 0x0138)
class UMaterialExpressionDepthBiasedAlpha : public UMaterialExpression
{
public:
unsigned long                           bNormalize : 1;                         // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                   BiasScale;                              // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FExpressionInput                 Alpha;                                  // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                 Bias;                                   // 0x0100 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDepthBiasedAlpha");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDepthBiasedBlend
// 0x00B0 (0x00C0 - 0x0170)
class UMaterialExpressionDepthBiasedBlend : public UMaterialExpression
{
public:
unsigned long                    bNormalize : 1;                    // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                            BiasScale;                         // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FExpressionInput          RGB;                               // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput          Alpha;                             // 0x0100 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput          Bias;                              // 0x0138 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDepthBiasedBlend");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDepthOfFieldFunction
// 0x0040 (0x00C0 - 0x0100)
class UMaterialExpressionDepthOfFieldFunction : public UMaterialExpression
{
public:
uint8_t                          FunctionValue;                     // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FExpressionInput          Depth;                             // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDepthOfFieldFunction");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDeriveNormalZ
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionDeriveNormalZ : public UMaterialExpression
{
public:
struct FExpressionInput                InXY;                           // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDeriveNormalZ");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDesaturation
// 0x0080 (0x00C0 - 0x0140)
class UMaterialExpressionDesaturation : public UMaterialExpression
{
public:
struct FExpressionInput                Input;                          // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                Percent;                         // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FLinearColor                  LuminanceFactors;                      // 0x0130 (0x0010)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDesaturation");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDestColor
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionDestColor : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDestColor");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDestDepth
// 0x0004 (0x00C0 - 0x00C4)
class UMaterialExpressionDestDepth : public UMaterialExpression
{
public:
unsigned long                    bNormalize : 1;                  // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDestDepth");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDistance
// 0x0070 (0x00C0 - 0x0130)
```

```
class UMaterialExpressionDistance : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                     // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                     // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDistance");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDivide
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionDivide : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                     // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                     // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDivide");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDotProduct
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionDotProduct : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                     // 0x00C0 (0x0038)
```

```
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                         B;                                    // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDotProduct");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDynamicParameter
// 0x0010 (0x00C0 - 0x00D0)
class UMaterialExpressionDynamicParameter : public UMaterialExpression
{
public:
TArray<class FString>                       ParamNames;                              // 0x00C0 (0x0010)
[0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDynamicParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMeshEmitterDynamicParameter
// 0x0000 (0x00D0 - 0x00D0)
class UMaterialExpressionMeshEmitterDynamicParameter : public
UMaterialExpressionDynamicParameter
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionMeshEmitterDynamicParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionEffectsIntensity
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionEffectsIntensity : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionEffectsIntensity");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFloor
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionFloor : public UMaterialExpression
{
public:
struct FExpressionInput                         Input;                                          // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFloor");
}

return uClassPointer;
};

};
```

```
}

// Class Engine.MaterialExpressionFluidNormal
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionFluidNormal : public UMaterialExpression
{
public:
struct FExpressionInput                    Coordinates;                          // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFluidNormal");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFmod
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionFmod : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                               // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                               // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFmod");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFoliageImpulseDirection
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionFoliageImpulseDirection : public UMaterialExpression
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFoliageImpulseDirection");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFoliageNormalizedRotationAxisAndAngle
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionFoliageNormalizedRotationAxisAndAngle : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionFoliageNormalizedRotationAxisAndAngle");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFontSample
// 0x000C (0x00C0 - 0x00CC)
class UMaterialExpressionFontSample : public UMaterialExpression
{
public:
class UFont*                    Font;                    // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                    FontTexturePage;                // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFontSample");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFontSampleParameter
// 0x0024 (0x00CC - 0x00F0)
class UMaterialExpressionFontSampleParameter : public UMaterialExpressionFontSample
{
public:
struct FName                    ParameterName;                    // 0x00D0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FGuid                    ExpressionGUID;                   // 0x00D8 (0x0010)
[0x0000000000000002] (CPF_Const)
struct FName                    Group;                            // 0x00E8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFontSampleParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFrac
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionFrac : public UMaterialExpression
{
public:
struct FExpressionInput         Input;                            // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFrac");
}
```

```
return uClassPointer;
};

};

// Class Engine.MaterialExpressionFresnel
// 0x0040 (0x00C0 - 0x0100)
class UMaterialExpressionFresnel : public UMaterialExpression
{
public:
float                                  Exponent;                              // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FExpressionInput                     Normal;                               // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFresnel");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFunctionInput
// 0x0088 (0x00C0 - 0x0148)
class UMaterialExpressionFunctionInput : public UMaterialExpression
{
public:
struct FExpressionInput                     Preview;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                               InputName;                           // 0x00F8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                               Description;                        // 0x0108 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FGuid                               Id;                                // 0x0118 (0x0010)
[0x0000000000000002] (CPF_Const)
uint8_t                                  InputType;                            // 0x0128 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector4                             PreviewValue;                         // 0x0130 (0x0010)
[0x0000000000000001] (CPF_Edit)
unsigned long                               bUsePreviewValueAsDefault : 1;               // 0x0140 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                               bCompilingFunctionPreview : 1;               // 0x0140 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
int32_t                                  SortPriority;                          // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFunctionInput");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFunctionOutput
// 0x0074 (0x00C0 - 0x0134)
class UMaterialExpressionFunctionOutput : public UMaterialExpression
{
public:
class FString                    OutputName;                    // 0x00C0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                    Description;                   // 0x00D0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                          SortPriority;                  // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FExpressionInput               A;                        // 0x00E8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                    bLastPreviewed : 1;            // 0x0120 (0x0004)
[0x0000000000000000] [0x00000001]
struct FGuid                     Id;                            // 0x0124 (0x0010)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFunctionOutput");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionGameObjectParameter
// 0x0001 (0x00C0 - 0x00C1)
class UMaterialExpressionGameObjectParameter : public UMaterialExpression
{
public:
uint8_t                          GameObjectShaderParamType;          // 0x00C0 (0x0001)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionGameObjectParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionGameParameter
// 0x0001 (0x00C0 - 0x00C1)
class UMaterialExpressionGameParameter : public UMaterialExpression
{
public:
uint8_t                          GameShaderParamType;                  // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionGameParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionIf
// 0x0118 (0x00C0 - 0x01D8)
class UMaterialExpressionIf : public UMaterialExpression
{
public:
struct FExpressionInput                A;                            // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                B;                            // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                AGreaterThanB;                      // 0x0130 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                AEqualsB;                       // 0x0168 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                ALessThanB;                       // 0x01A0 (0x0038)
```

```
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionIf");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLandscapeLayerBlend
// 0x0020 (0x00C0 - 0x00E0)
class UMaterialExpressionLandscapeLayerBlend : public UMaterialExpression
{
public:
TArray<struct FLayerBlendInput>          Layers;                    // 0x00C0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FGuid                             ExpressionGUID;            // 0x00D0 (0x0010)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLandscapeLayerBlend");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLensFlareIntensity
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionLensFlareIntensity : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLensFlareIntensity");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLensFlareOcclusion
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionLensFlareOcclusion : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLensFlareOcclusion");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLensFlareRadialDistance
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionLensFlareRadialDistance : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLensFlareRadialDistance");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLensFlareRayDistance
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionLensFlareRayDistance : public UMaterialExpression
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLensFlareRayDistance");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLensFlareSourceDistance
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionLensFlareSourceDistance : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionLensFlareSourceDistance");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLightmapUVs
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionLightmapUVs : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLightmapUVs");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLightmassReplace
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionLightmassReplace : public UMaterialExpression
{
public:
struct FExpressionInput                    Realtime;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Lightmass;                              // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLightmassReplace");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLightVector
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionLightVector : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLightVector");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionLinearInterpolate
// 0x00A8 (0x00C0 - 0x0168)
class UMaterialExpressionLinearInterpolate : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                    // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                    // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Alpha;                                  // 0x0130 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionLinearInterpolate");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMaterialFunctionCall
// 0x0028 (0x00C0 - 0x00E8)
class UMaterialExpressionMaterialFunctionCall : public UMaterialExpression
{
public:
class UMaterialFunction*                    MaterialFunction;                      // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<struct FFunctionExpressionInput>          FunctionInputs;                        // 0x00C8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FFunctionExpressionOutput>          FunctionOutputs;                        // 0x00D8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMaterialFunctionCall");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMax
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionMax : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                  // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMax");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMeshEmitterVertexColor
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionMeshEmitterVertexColor : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMeshEmitterVertexColor");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMin
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionMin : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                  // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMin");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMultiply
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionMultiply : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                  // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMultiply");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMusicAnalysisParameter
// 0x0001 (0x00C0 - 0x00C1)
class UMaterialExpressionMusicAnalysisParameter : public UMaterialExpression
{
public:
uint8_t                                    MusicAnalysisShaderParamType;            // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMusicAnalysisParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionNormalize
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionNormalize : public UMaterialExpression
{
public:
struct FExpressionInput                    VectorInput;                          // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionNormalize");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionObjectOrientation
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionObjectOrientation : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionObjectOrientation");
}

return uClassPointer;
};

};
```

```
// Class Engine.MaterialExpressionObjectRadius
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionObjectRadius : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionObjectRadius");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionObjectWorldPosition
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionObjectWorldPosition : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionObjectWorldPosition");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionOcclusionPercentage
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionOcclusionPercentage : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionOcclusionPercentage");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionOneMinus
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionOneMinus : public UMaterialExpression
{
public:
struct FExpressionInput                    Input;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionOneMinus");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionPanner
// 0x0078 (0x00C0 - 0x0138)
class UMaterialExpressionPanner : public UMaterialExpression
{
public:
struct FExpressionInput                    Coordinate;                            // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Time;                               // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                              SpeedX;                          // 0x0130 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              SpeedY;                          // 0x0134 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionPanner");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionParameter
// 0x0020 (0x00C0 - 0x00E0)
class UMaterialExpressionParameter : public UMaterialExpression
{
public:
struct FName                    ParameterName;                  // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FGuid                    ExpressionGUID;                 // 0x00C8 (0x0010)
[0x0000000000000002] (CPF_Const)
struct FName                    Group;                          // 0x00D8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionScalarParameter
// 0x0010 (0x00E0 - 0x00F0)
class UMaterialExpressionScalarParameter : public UMaterialExpressionParameter
{
public:
float                           DefaultValue;                   // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                   bRanged : 1;                    // 0x00E4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                           MinParameterValue;              // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           MaxParameterValue;              // 0x00EC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionScalarParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionStaticBoolParameter
// 0x0010 (0x00E0 - 0x00F0)
class UMaterialExpressionStaticBoolParameter : public UMaterialExpressionParameter
{
public:
unsigned long                    DefaultValue : 1;                // 0x00E0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    ExtendedCaptionDisplay : 1;          // 0x00E0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FPointer                  InstanceOverride;                // 0x00E8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionStaticBoolParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionStaticSwitchParameter
// 0x0070 (0x00F0 - 0x0160)
class UMaterialExpressionStaticSwitchParameter : public
UMaterialExpressionStaticBoolParameter
{
public:
struct FExpressionInput              A;                       // 0x00F0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput              B;                       // 0x0128 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionStaticSwitchParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionStaticComponentMaskParameter
// 0x0048 (0x00E0 - 0x0128)
class UMaterialExpressionStaticComponentMaskParameter : public
UMaterialExpressionParameter
{
public:
struct FExpressionInput                    Input;                              // 0x00E0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                              DefaultR : 1;                       // 0x0118 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                              DefaultG : 1;                       // 0x0118 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                              DefaultB : 1;                       // 0x0118 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                              DefaultA : 1;                       // 0x0118 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FPointer                            InstanceOverride;                   // 0x0120 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionStaticComponentMaskParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionVectorParameter
// 0x0034 (0x00E0 - 0x0114)
class UMaterialExpressionVectorParameter : public UMaterialExpressionParameter
{
public:
struct FLinearColor                        DefaultValue;                       // 0x00E0 (0x0010)
[0x0000000000000001] (CPF_Edit)
unsigned long                              bRanged : 1;                        // 0x00F0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FLinearColor                        MinParameterValue;                  // 0x00F4 (0x0010)
[0x0000000000000001] (CPF_Edit)
```

```
struct FLinearColor                    MaxParameterValue;                    // 0x0104 (0x0010)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionVectorParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionVectorParameterReference
// 0x0008 (0x00E0 - 0x00E8)
class UMaterialExpressionVectorParameterReference : public UMaterialExpressionParameter
{
public:
struct FName                           ReferencedParameter;                  // 0x00E0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionVectorParameterReference");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionParticleMacroUV
// 0x0004 (0x00C0 - 0x00C4)
class UMaterialExpressionParticleMacroUV : public UMaterialExpression
{
public:
unsigned long                          bUseViewSpace : 1;                    // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionParticleMacroUV");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionPerInstanceRandom
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionPerInstanceRandom : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionPerInstanceRandom");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionPitchTekTextureSample
// 0x0039 (0x00C0 - 0x00F9)
class UMaterialExpressionPitchTekTextureSample : public UMaterialExpression
{
public:
struct FExpressionInput                    Coordinates;                         // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                          PitchTekTextureType;                    // 0x00F8 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionPitchTekTextureSample");
}

return uClassPointer;
};
```

```
};

// Class Engine.MaterialExpressionPixelDepth
// 0x0004 (0x00C0 - 0x00C4)
class UMaterialExpressionPixelDepth : public UMaterialExpression
{
public:
unsigned long                    bNormalize : 1;                 // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionPixelDepth");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionPower
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionPower : public UMaterialExpression
{
public:
struct FExpressionInput              Base;                       // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput              Exponent;                     // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionPower");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionQualitySwitch
// 0x00A8 (0x00C0 - 0x0168)
class UMaterialExpressionQualitySwitch : public UMaterialExpression
```

```
{
public:
struct FExpressionInput                    High;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Low;                               // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    HandheldQuality;                        // 0x0130 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionQualitySwitch");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionReflectionVector
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionReflectionVector : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionReflectionVector");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionRotateAboutAxis
// 0x00A8 (0x00C0 - 0x0168)
class UMaterialExpressionRotateAboutAxis : public UMaterialExpression
{
public:
struct FExpressionInput                    NormalizedRotationAxisAndAngle;           // 0x00C0
(0x0038) [0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    PositionOnAxis;                         // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
struct FExpressionInput                    Position;                              // 0x0130 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionRotateAboutAxis");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionRotator
// 0x007C (0x00C0 - 0x013C)
class UMaterialExpressionRotator : public UMaterialExpression
{
public:
struct FExpressionInput                    Coordinate;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Time;                              // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                              CenterX;                              // 0x0130 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CenterY;                              // 0x0134 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              Speed;                              // 0x0138 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionRotator");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionSceneDepth
// 0x003C (0x00C0 - 0x00FC)
class UMaterialExpressionSceneDepth : public UMaterialExpression
{
public:
```

```
struct FExpressionInput                    Coordinates;                            // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                         bNormalize : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionSceneDepth");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionSceneTexture
// 0x0040 (0x00C0 - 0x0100)
class UMaterialExpressionSceneTexture : public UMaterialExpression
{
public:
struct FExpressionInput                    Coordinates;                            // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                               SceneTextureType;                      // 0x00F8 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                         ScreenAlign : 1;                        // 0x00FC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionSceneTexture");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionScreenPosition
// 0x0004 (0x00C0 - 0x00C4)
class UMaterialExpressionScreenPosition : public UMaterialExpression
{
public:
unsigned long                         ScreenAlign : 1;                        // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionScreenPosition");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionScreenSize
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionScreenSize : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionScreenSize");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionSine
// 0x003C (0x00C0 - 0x00FC)
class UMaterialExpressionSine : public UMaterialExpression
{
public:
struct FExpressionInput                    Input;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                Period;                              // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionSine");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionSphereMask
// 0x00E8 (0x00C0 - 0x01A8)
class UMaterialExpressionSphereMask : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                  // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Radius;                             // 0x0130 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Hardness;                           // 0x0168 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
float                              AttenuationRadius;                          // 0x01A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              HardnessPercent;                            // 0x01A4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionSphereMask");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionSquareRoot
// 0x0038 (0x00C0 - 0x00F8)
class UMaterialExpressionSquareRoot : public UMaterialExpression
{
public:
struct FExpressionInput                    Input;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionSquareRoot");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionStaticBool
// 0x0004 (0x00C0 - 0x00C4)
class UMaterialExpressionStaticBool : public UMaterialExpression
{
public:
unsigned long                          Value : 1;                               // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionStaticBool");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionStaticSwitch
// 0x00B0 (0x00C0 - 0x0170)
class UMaterialExpressionStaticSwitch : public UMaterialExpression
{
public:
unsigned long                          DefaultValue : 1;                        // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          ExtendedCaptionDisplay : 1;              // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FExpressionInput                    A;                                   // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                   // 0x0100 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    Value;                                 // 0x0138 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionStaticSwitch");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionSubtract
// 0x0070 (0x00C0 - 0x0130)
class UMaterialExpressionSubtract : public UMaterialExpression
{
public:
struct FExpressionInput                    A;                                  // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                    B;                                  // 0x00F8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionSubtract");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTerrainLayerCoords
// 0x0014 (0x00C0 - 0x00D4)
class UMaterialExpressionTerrainLayerCoords : public UMaterialExpression
{
public:
uint8_t                                    MappingType;                        // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                      MappingScale;                       // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      MappingRotation;                    // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      MappingPanU;                        // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      MappingPanV;                        // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTerrainLayerCoords");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTerrainLayerSwitch
// 0x0094 (0x00C0 - 0x0154)
class UMaterialExpressionTerrainLayerSwitch : public UMaterialExpression
{
public:
struct FPointer                      InstanceOverride;                  // 0x00C0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FExpressionInput                   LayerUsed;                       // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FExpressionInput                   LayerNotUsed;                     // 0x0100 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FName                          ParameterName;                     // 0x0138 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                         PreviewUsed : 1;                   // 0x0140 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FGuid                        ExpressionGUID;                    // 0x0144 (0x0010)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTerrainLayerSwitch");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTerrainLayerWeight
// 0x0094 (0x00C0 - 0x0154)
class UMaterialExpressionTerrainLayerWeight : public UMaterialExpression
{
public:
struct FPointer                      InstanceOverride;                  // 0x00C0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FExpressionInput                   Base;                         // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
struct FExpressionInput                        Layer;                                   // 0x0100 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FName                               ParameterName;                           // 0x0138 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                  PreviewWeight;                           // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FGuid                               ExpressionGUID;                          // 0x0144 (0x0010)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTerrainLayerWeight");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTexelSize
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionTexelSize : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTexelSize");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureCoordinate
// 0x0010 (0x00C0 - 0x00D0)
class UMaterialExpressionTextureCoordinate : public UMaterialExpression
{
public:
int32_t                                  CoordinateIndex;                         // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  UTiling;                                 // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                                   VTiling;                                    // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           UnMirrorU : 1;                              // 0x00CC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           UnMirrorV : 1;                              // 0x00CC (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTextureCoordinate");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureObject
// 0x0008 (0x00C0 - 0x00C8)
class UMaterialExpressionTextureObject : public UMaterialExpression
{
public:
class UTexture*                         Texture;                                    // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTextureObject");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSample
// 0x0078 (0x00C0 - 0x0138)
class UMaterialExpressionTextureSample : public UMaterialExpression
{
public:
class UTexture*                         Texture;                                    // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FExpressionInput                 Coordinates;                                // 0x00C8 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
struct FExpressionInput                    TextureObject;                         // 0x0100 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTextureSample");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionDepthBiasBlend
// 0x0040 (0x0138 - 0x0178)
class UMaterialExpressionDepthBiasBlend : public UMaterialExpressionTextureSample
{
public:
unsigned long                              bNormalize : 1;                        // 0x0138 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                      BiasScale;                             // 0x013C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FExpressionInput                    Bias;                                  // 0x0140 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionDepthBiasBlend");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionFlipBookSample
// 0x0000 (0x0138 - 0x0138)
class UMaterialExpressionFlipBookSample : public UMaterialExpressionTextureSample
{
public:

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionFlipBookSample");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMeshSubUV
// 0x0000 (0x0138 - 0x0138)
class UMaterialExpressionMeshSubUV : public UMaterialExpressionTextureSample
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMeshSubUV");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionMeshSubUVBlend
// 0x0000 (0x0138 - 0x0138)
class UMaterialExpressionMeshSubUVBlend : public UMaterialExpressionMeshSubUV
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionMeshSubUVBlend");
}

return uClassPointer;
};

};
```

```
// Class Engine.MaterialExpressionParticleSubUV
// 0x0000 (0x0138 - 0x0138)
class UMaterialExpressionParticleSubUV : public UMaterialExpressionTextureSample
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionParticleSubUV");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameter
// 0x0020 (0x0138 - 0x0158)
class UMaterialExpressionTextureSampleParameter : public UMaterialExpressionTextureSample
{
public:
struct FName                                ParameterName;                          // 0x0138 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FGuid                               ExpressionGUID;                         // 0x0140 (0x0010)
[0x0000000000000002] (CPF_Const)
struct FName                                Group;                                  // 0x0150 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureObjectParameter
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureObjectParameter : public
UMaterialExpressionTextureSampleParameter
{
```

```
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTextureObjectParameter");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameter2D
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureSampleParameter2D : public
UMaterialExpressionTextureSampleParameter
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameter2D");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionAntialiasedTextureMask
// 0x0005 (0x0158 - 0x015D)
class UMaterialExpressionAntialiasedTextureMask : public
UMaterialExpressionTextureSampleParameter2D
{
public:
float                                    Threshold;                               // 0x0158 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  Channel;                                 // 0x015C (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionAntialiasedTextureMask");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameterFlipbook
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureSampleParameterFlipbook : public
UMaterialExpressionTextureSampleParameter2D
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameterFlipbook");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameterMeshSubUV
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureSampleParameterMeshSubUV : public
UMaterialExpressionTextureSampleParameter2D
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameterMeshSubUV");
}

return uClassPointer;
```

```
};

};

// Class Engine.MaterialExpressionTextureSampleParameterMeshSubUVBlend
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureSampleParameterMeshSubUVBlend : public
UMaterialExpressionTextureSampleParameterMeshSubUV
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameterMeshSubUVBlend");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameterSubUV
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureSampleParameterSubUV : public
UMaterialExpressionTextureSampleParameter2D
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameterSubUV");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameterCube
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureSampleParameterCube : public
UMaterialExpressionTextureSampleParameter
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameterCube");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameterMovie
// 0x0000 (0x0158 - 0x0158)
class UMaterialExpressionTextureSampleParameterMovie : public
UMaterialExpressionTextureSampleParameter
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameterMovie");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTextureSampleParameterNormal
// 0x0008 (0x0158 - 0x0160)
class UMaterialExpressionTextureSampleParameterNormal : public
UMaterialExpressionTextureSampleParameter
{
public:
struct FPointer                    InstanceOverride;                 // 0x0158 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.MaterialExpressionTextureSampleParameterNormal");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTime
// 0x0004 (0x00C0 - 0x00C4)
class UMaterialExpressionTime : public UMaterialExpression
{
public:
unsigned long                          bIgnorePause : 1;                       // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTime");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTransform
// 0x003A (0x00C0 - 0x00FA)
class UMaterialExpressionTransform : public UMaterialExpression
{
public:
struct FExpressionInput                  Input;                               // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                                TransformSourceType;                   // 0x00F8 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                                TransformType;                        // 0x00F9 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTransform");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTransformPosition
// 0x003A (0x00C0 - 0x00FA)
class UMaterialExpressionTransformPosition : public UMaterialExpression
{
public:
struct FExpressionInput                  Input;                              // 0x00C0 (0x0038)
[0x0000000000400000] (CPF_NeedCtorLink)
uint8_t                                  TransformSourceType;                // 0x00F8 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                                  TransformType;                      // 0x00F9 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTransformPosition");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionTwoSidedSign
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionTwoSidedSign : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionTwoSidedSign");
}

return uClassPointer;
};
```

```
};

// Class Engine.MaterialExpressionVertexColor
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionVertexColor : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionVertexColor");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionWindDirectionAndSpeed
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionWindDirectionAndSpeed : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionWindDirectionAndSpeed");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionWorldNormal
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionWorldNormal : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionWorldNormal");
}

return uClassPointer;
};

};

// Class Engine.MaterialExpressionWorldPosition
// 0x0000 (0x00C0 - 0x00C0)
class UMaterialExpressionWorldPosition : public UMaterialExpression
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialExpressionWorldPosition");
}

return uClassPointer;
};

};

// Class Engine.MaterialFunction
// 0x0060 (0x0060 - 0x00C0)
class UMaterialFunction : public UObject
{
public:
struct FGuid                        StateId;                        // 0x0060 (0x0010)
[0x0000000000200000]
class UMaterialFunction*                 ParentFunction;                  // 0x0070 (0x0008)
[0x0000000800002000] (CPF_Transient)
class FString                       Description;                      // 0x0078 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                        bExposeToLibrary : 1;                // 0x0088 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bReentrantFlag : 1;                // 0x0088 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
TArray<class FString>                      LibraryCategories;                  // 0x0090 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UMaterialExpression*>             FunctionExpressions;                  // 0x00A0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<class UMaterialExpressionComment*>        FunctionEditorComments;               //
0x00B0 (0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialFunction");
}

return uClassPointer;
};

};

// Class Engine.MaterialInstance
// 0x00A4 (0x0274 - 0x0318)
class UMaterialInstance : public UMaterialInterface
{
public:
class UPhysicalMaterial*                PhysMaterial;                        // 0x0278 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*               Parent;                              // 0x0280 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UTexture2D*                     PhysMaterialMask;                     // 0x0288 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                          PhysMaterialMaskUVChannel;               // 0x0290 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UPhysicalMaterial*                BlackPhysicalMaterial;                  // 0x0298 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UPhysicalMaterial*                WhitePhysicalMaterial;                  // 0x02A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bHasStaticPermutationResource : 1;           // 0x02A8
(0x0004) [0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                       bStaticPermutationDirty : 1;              // 0x02A8 (0x0004)
[0x0000000000003000] [0x00000002] (CPF_Native | CPF_Transient)
unsigned long                       ReentrantFlag : 1;                    // 0x02A8 (0x0004)
[0x0000000000001002] [0x00000004] (CPF_Const | CPF_Native)
unsigned long                       bNeedsMaterialFlattening : 1;             // 0x02A8 (0x0004)
[0x0000000000002002] [0x00000008] (CPF_Const | CPF_Transient)
struct FPointer                     StaticParameters[0x2];                 // 0x02B0 (0x0010)
[0x0000000000201002] (CPF_Const | CPF_Native)
struct FPointer                     StaticPermutationResources[0x2];            // 0x02C0 (0x0010)
[0x0000000000201002] (CPF_Const | CPF_Native)
struct FPointer                     Resources[0x3];                      // 0x02D0 (0x0018)
[0x0000000000201002] (CPF_Const | CPF_Native)
TArray<class UTexture*>                 ReferencedTextures;                   // 0x02E8 (0x0010)
[0x0000000020400002] (CPF_Const | CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FGuid>                  ReferencedTextureGuids;                 // 0x02F8 (0x0010)
[0x0000000800400002] (CPF_Const | CPF_NeedCtorLink)
struct FGuid                      ParentLightingGuid;                   // 0x0308 (0x0010)
[0x0000000000000002] (CPF_Const)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialInstance");
}

return uClassPointer;
};

void SetVectorParameterValues(class FString Prefix, TArray<struct FVectorParameterValue>&
VectorParamValues);
void SetTextureParameterValues(class FString Prefix, TArray<struct FTextureParameterValue>&
TextureParamValues);
void SetScalarParameterValues(class FString Prefix, TArray<struct FScalarParameterValue>&
ScalarParamValues);
void SetFontParameterValues(class FString Prefix, TArray<struct FFontParameterValue>&
FontParamValues);
bool IsInMapOrTransientPackage();
void ClearParameterValues();
void SetFontParameterValue(struct FName ParameterName, class UFont* FontValue, int32_t
FontPage);
bool GetTextureParameterValue(struct FName ParameterName, class UTexture*& Value);
void SetTextureParameterValue(struct FName ParameterName, class UTexture* Value);
void SetScalarCurveParameterValue(struct FName ParameterName, struct FInterpCurveFloat&
Value);
void SetScalarParameterValue(struct FName ParameterName, float Value);
void SetVectorParameterValue(struct FName ParameterName, struct FLinearColor Value);
void SetParent(class UMaterialInterface* NewParent);
};

// Class Engine.MaterialInstanceConstant
// 0x0048 (0x0318 - 0x0360)
class UMaterialInstanceConstant : public UMaterialInstance
{
public:
struct FPointer                          VfTable_IISetParameter;                  // 0x0318 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
TArray<struct FFontParameterValue>              FontParameterValues;                     // 0x0320
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<struct FScalarParameterValue>            ScalarParameterValues;                   // 0x0330
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<struct FTextureParameterValue>           TextureParameterValues;                  // 0x0340
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<struct FVectorParameterValue>            VectorParameterValues;                   // 0x0350
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialInstanceConstant");
}

return uClassPointer;
};

void SetActorParameter(struct FName Key, class AActor* Value);
void SetLinearColorParameter(struct FName Key, struct FLinearColor Value);
void SetVectorParameter(struct FName Key, struct FVector V);
void SetFloatParameter(struct FName Key, float Value);
void SetNameParameter(struct FName Key, struct FName Value);
void ClearParameterValues();
bool GetMobileVectorParameterValue(struct FName ParameterName, struct FLinearColor&
OutValue);
bool GetMobileTextureParameterValue(struct FName ParameterName, class UTexture*&
OutValue);
bool GetMobileScalarParameterValue(struct FName ParameterName, float& OutValue);
void SetFontParameterValue(struct FName ParameterName, class UFont* FontValue, int32_t
FontPage);
void SetVectorParameterValue(struct FName ParameterName, struct FLinearColor Value);
void SetTextureParameterValue(struct FName ParameterName, class UTexture* Value);
void SetScalarParameterValue(struct FName ParameterName, float Value);
void SetParent(class UMaterialInterface* NewParent);
bool GetFontParameterValue(struct FName ParameterName, class UFont*& OutFontValue,
int32_t& OutFontPage);
bool GetTextureParameterValue(struct FName ParameterName, class UTexture*& OutValue);
bool GetScalarParameterValue(struct FName ParameterName, float& OutValue);
bool GetVectorParameterValue(struct FName ParameterName, struct FLinearColor& OutValue);
};

// Class Engine.LandscapeMaterialInstanceConstant
// 0x000C (0x0360 - 0x036C)
class ULandscapeMaterialInstanceConstant : public UMaterialInstanceConstant
{
public:
unsigned long                   bIsLayerThumbnail : 1;            // 0x0360 (0x0004)
[0x0000000000000000] [0x00000001]
int32_t                      DataWeightmapIndex;              // 0x0364 (0x0004)
[0x0000000000000000]
int32_t                      DataWeightmapSize;               // 0x0368 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeMaterialInstanceConstant");
}
```

```
return uClassPointer;
};

};

// Class Engine.MaterialInstanceTimeVarying
// 0x0058 (0x0318 - 0x0370)
class UMaterialInstanceTimeVarying : public UMaterialInstance
{
public:
unsigned long                    bAutoActivateAll : 1;              // 0x0318 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                            Duration;                          // 0x031C (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FFontParameterValueOverTime>       FontParameterValues;                 //
0x0320 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FScalarParameterValueOverTime>      ScalarParameterValues;                //
0x0330 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FTextureParameterValueOverTime>     TextureParameterValues;               //
0x0340 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FVectorParameterValueOverTime>      VectorParameterValues;                //
0x0350 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FLinearColorParameterValueOverTime>
LinearColorParameterValues;              // 0x0360 (0x0010) [0x0000000000400001] (CPF_Edit |
CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialInstanceTimeVarying");
}

return uClassPointer;
};

bool CheckForVectorParameterConflicts(struct FName ParameterName);
float GetMaxDurationFromAllParameters();
void ClearParameterValues();
void SetFontParameterValue(struct FName ParameterName, class UFont* FontValue, int32_t
FontPage);
void SetVectorStartTime(struct FName ParameterName, float Value);
void SetLinearColorCurveParameterValue(struct FName ParameterName, struct
FInterpCurveLinearColor& Value);
void SetLinearColorParameterValue(struct FName ParameterName, struct FLinearColor& Value);
void SetVectorCurveParameterValue(struct FName ParameterName, struct FInterpCurveVector&
Value);
void SetVectorParameterValue(struct FName ParameterName, struct FLinearColor Value);
void SetTextureParameterValue(struct FName ParameterName, class UTexture* Value);
void SetDuration(float Value);
void SetScalarStartTime(struct FName ParameterName, float Value);
void SetScalarCurveParameterValue(struct FName ParameterName, struct FInterpCurveFloat&
Value);
void SetScalarParameterValue(struct FName ParameterName, float Value);
void SetParent(class UMaterialInterface* NewParent);
bool GetTextureParameterValue(struct FName ParameterName, class UTexture*& OutValue);
bool GetScalarCurveParameterValue(struct FName ParameterName, struct FInterpCurveFloat&
OutValue);
bool GetScalarParameterValue(struct FName ParameterName, float& OutValue);
bool GetLinearColorCurveParameterValue(struct FName ParameterName, struct
FInterpCurveLinearColor& OutValue);
bool GetLinearColorParameterValue(struct FName ParameterName, struct FLinearColor&
OutValue);
bool GetVectorCurveParameterValue(struct FName ParameterName, struct FInterpCurveVector&
OutValue);
bool GetVectorParameterValue(struct FName ParameterName, struct FLinearColor& OutValue);
bool GetFontParameterValue(struct FName ParameterName, class UFont*& OutFontValue,
int32_t& OutFontPage);
};

// Class Engine.EmitterCameraLensEffectBase
// 0x003C (0x027C - 0x02B8)
class AEmitterCameraLensEffectBase : public AEmitter
{
public:
class UParticleSystem*                  PS_CameraEffect;                        // 0x0280 (0x0008)
[0x0000000000000000]
class UParticleSystem*                  PS_CameraEffectNonExtremeContent;           // 0x0288
(0x0008) [0x0000000000000000]
float                                   BaseFOV;                                // 0x0290 (0x0004)
[0x0000000000000000]
float                                   DistFromCamera;                         // 0x0294 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                           bAllowMultipleInstances : 1;                // 0x0298 (0x0004)
[0x0000008000000003] [0x00000001] (CPF_Edit | CPF_Const)
TArray<class UClass*>                   EmittersToTreatAsSame;                  // 0x02A0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class ACamera*                          BaseCamera;                             // 0x02B0 (0x0008)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EmitterCameraLensEffectBase");
}

return uClassPointer;
};

void UpdateLocation(float CamFOVDeg, struct FVector& CamLoc, struct FRotator& CamRot);
void ActivateLensEffect();
void PostBeginPlay();
void NotifyRetriggered();
void RegisterCamera(class ACamera* C);
void Destroyed();
};

// Class Engine.ParticleEventManager
// 0x0000 (0x0268 - 0x0268)
class AParticleEventManager : public AActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleEventManager");
}

return uClassPointer;
};

void eventHandleParticleModuleEventSendToGame(class UParticleModuleEventSendToGame*
InEvent, struct FVector& InCollideDirection, struct FVector& InHitLocation, struct FVector&
InHitNormal, struct FName& InBoneName);
};

// Class Engine.ParticleSystemComponent
// 0x0188 (0x0258 - 0x03E0)
class UParticleSystemComponent : public UPrimitiveComponent
{
public:
struct FPointer                         VfTable_IISetParameter;                   // 0x0258 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UParticleSystem*                       Template;                            // 0x0260 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UClass*                           LightEnvironmentClass;                    // 0x0268 (0x0008)
[0x0000000000000000]
class AActor*                           LightEnvironmentSharedInstigator;            // 0x0270 (0x0008)
[0x0000000000002000] (CPF_Transient)
int32_t                             MaxLightEnvironmentPooledReuses;              // 0x0278 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FPointer>                      EmitterInstances;                    // 0x0280 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class UStaticMeshComponent*>              SMComponents;                           // 0x0290
(0x0010) [0x000000000468200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UMaterialInterface*>              SMMaterialInterfaces;                   // 0x02A0
(0x0010) [0x0000000000602002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class USkeletalMeshComponent*>              SkelMeshComponents;                        //
```

```
0x02B0 (0x0010) [0x000000000468200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
TArray<struct FViewParticleEmitterInstanceMotionBlurInfo>
ViewMBInfoArray;                          // 0x02C0 (0x0010) [0x0000000000003002] (CPF_Const |
CPF_Native | CPF_Transient)
unsigned long                             bAutoActivate : 1;                          // 0x02D0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                             bWasCompleted : 1;                          // 0x02D0 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                             bSuppressSpawning : 1;                      // 0x02D0 (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                             bWasDeactivated : 1;                        // 0x02D0 (0x0004)
[0x0000000000002002] [0x00000008] (CPF_Const | CPF_Transient)
unsigned long                             bResetOnDetach : 1;                         // 0x02D0 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                             bUpdateOnDedicatedServer : 1;               // 0x02D0 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                             bJustAttached : 1;                          // 0x02D0 (0x0004)
[0x0000000000002000] [0x00000040] (CPF_Transient)
unsigned long                             bIsActive : 1;                              // 0x02D0 (0x0004)
[0x0000000000002000] [0x00000080] (CPF_Transient)
unsigned long                             bHasBeenActivated : 1;                      // 0x02D0 (0x0004)
[0x0000000000002000] [0x00000100] (CPF_Transient)
unsigned long                             bWarmingUp : 1;                             // 0x02D0 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                             bIsCachedInPool : 1;                        // 0x02D0 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                             bOverrideLODMethod : 1;                     // 0x02D0 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                             bSkipUpdateDynamicDataDuringTick : 1;       // 0x02D0
(0x0004) [0x0000000000000000] [0x00001000]
unsigned long                             bSkipBoundsUpdate : 1;                      // 0x02D0 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                             bUpdateComponentInTick : 1;                 // 0x02D0 (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                             bDeferredBeamUpdate : 1;                    // 0x02D0 (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                             bForcedInActive : 1;                        // 0x02D0 (0x0004)
[0x0000000000002000] [0x00010000] (CPF_Transient)
unsigned long                             bIsWarmingUp : 1;                           // 0x02D0 (0x0004)
[0x0000000000002000] [0x00020000] (CPF_Transient)
unsigned long                             bIsViewRelevanceDirty : 1;                  // 0x02D0 (0x0004)
[0x0000000000002000] [0x00040000] (CPF_Transient)
unsigned long                             bRecacheViewRelevance : 1;                  // 0x02D0 (0x0004)
[0x0000000000002000] [0x00080000] (CPF_Transient)
unsigned long                             bParticleRequiresUpdateInTick : 1;          // 0x02D0 (0x0004)
[0x0000000000002000] [0x00100000] (CPF_Transient)
unsigned long                             bLODUpdatePending : 1;                      // 0x02D0 (0x0004)
[0x0000000000002000] [0x00200000] (CPF_Transient)
unsigned long                             bSkipSpawnCountCheck : 1;                   // 0x02D0 (0x0004)
[0x0000000000002000] [0x00400000] (CPF_Transient)
uint8_t                              LastDetailMode;                                  // 0x02D4 (0x0001)
[0x0000000000002000] (CPF_Transient)
uint8_t                              LODMethod;                                       // 0x02D5 (0x0001)
```

```
[0x0000000000000001] (CPF_Edit)
uint8_t                          ReplayState;                        // 0x02D6 (0x0001)
[0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<struct FParticleSysParam>             InstanceParameters;                    // 0x02D8
(0x0010) [0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)
struct FVector                         OldPosition;                        // 0x02E8 (0x000C)
[0x0000000000000000]
struct FVector                         PartSysVelocity;                      // 0x02F4 (0x000C)
[0x0000000000000000]
float                           WarmupTime;                         // 0x0300 (0x0004)
[0x0000000000000000]
float                           WarmupTickRate;                       // 0x0304 (0x0004)
[0x0000000000000000]
int32_t                          LODLevel;                          // 0x0308 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           SecondsBeforeInactive;                  // 0x030C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           TimeSinceLastForceUpdateTransform;          // 0x0310 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           MaxTimeBeforeForceUpdateTransform;          // 0x0314 (0x0004)
[0x0000000000000000]
int32_t                          EditorLODLevel;                       // 0x0318 (0x0004)
[0x0000000800000000]
int32_t                          EditorDetailMode;                      // 0x031C (0x0004)
[0x0000000800000000]
float                           AccumTickTime;                        // 0x0320 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FMaterialViewRelevance>           CachedViewRelevanceFlags;                //
0x0328 (0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class UParticleSystemReplay*>           ReplayClips;                         // 0x0338
(0x0010) [0x0000000004400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink | CPF_EditInline)
int32_t                          ReplayClipIDNumber;                    // 0x0348 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                          ReplayFrameIndex;                     // 0x034C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                           AccumLODDistanceCheckTime;               // 0x0350 (0x0004)
[0x0000000000002000] (CPF_Transient)
TArray<struct FParticleEventSpawnData>          SpawnEvents;                         // 0x0358
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FParticleEventDeathData>          DeathEvents;                         // 0x0368
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FParticleEventCollideData>         CollisionEvents;                      // 0x0378
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FParticleEventKismetData>          KismetEvents;                        // 0x0388
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FParticleEventAttractorCollideData>  AttractorCollisionEvents;                //
0x0398 (0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FPointer                         ReleaseResourcesFence;               // 0x03A8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
float                           CustomTimeDilation;                    // 0x03B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           EmitterDelay;                        // 0x03B4 (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FVector                         FakeVelocity;                        // 0x03B8 (0x000C)
```

```
[0x0000000000000001] (CPF_Edit)
struct FScriptDelegate                    __OnSystemFinished__Delegate;              // 0x03C8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleSystemComponent");
}

return uClassPointer;
};

void Warmup();
void SetParameters(TArray<struct FParticleSysParam>& Params);
void SetLinearColorParameter(struct FName Key, struct FLinearColor Value);
void SetNameParameter(struct FName Key, struct FName Value);
void SetStopSpawning(int32_t InEmitterIndex, unsigned long bInStopSpawning);
void ResetToDefaults();
void SetActive(unsigned long bNowActive, unsigned long bFlagAsJustAttached);
void ClearAllParameters();
void ClearParameter(struct FName ParameterName, uint8_t ParameterType);
bool GetMaterialParameter(struct FName InName, class UMaterialInterface*& OutMaterial);
bool GetActorParameter(struct FName InName, class AActor*& OutActor);
bool GetColorParameter(struct FName InName, struct FColor& OutColor);
bool GetVectorParameter(struct FName InName, struct FVector& OutVector);
bool GetFloatParameter(struct FName InName, float& OutFloat);
bool GetMeshParameter(struct FName ParameterName, class UStaticMesh*& Param);
void SetMeshParameter(struct FName ParameterName, class UStaticMesh* Param);
void SetMaterialParameter(struct FName ParameterName, class UMaterialInterface* Param);
void SetActorParameter(struct FName ParameterName, class AActor* Param);
void SetColorParameter(struct FName ParameterName, struct FColor Param);
void SetVectorRandParameter(struct FName ParameterName, struct FVector& Param, struct
FVector& ParamLow);
void SetVectorParameter(struct FName ParameterName, struct FVector Param);
void SetFloatRandParameter(struct FName ParameterName, float Param, float ParamLow);
void SetFloatParameter(struct FName ParameterName, float Param);
int32_t GetEditorLODLevel();
int32_t GetLODLevel();
void SetEditorLODLevel(int32_t InLODLevel);
void SetLODLevel(int32_t InLODLevel);
bool SystemHasCompleted();
float GetMaxLifespan();
int32_t DetermineLODLevelForLocation(struct FVector& EffectLocation);
void SetBeamTargetStrength(int32_t EmitterIndex, float NewTargetStrength, int32_t TargetIndex);
void SetBeamTargetTangent(int32_t EmitterIndex, struct FVector NewTangentPoint, int32_t
TargetIndex);
void SetBeamTargetPoint(int32_t EmitterIndex, struct FVector NewTargetPoint, int32_t
TargetIndex);
void SetBeamSourceStrength(int32_t EmitterIndex, float NewSourceStrength, int32_t
SourceIndex);
void SetBeamSourceTangent(int32_t EmitterIndex, struct FVector NewTangentPoint, int32_t
SourceIndex);
void SetBeamSourcePoint(int32_t EmitterIndex, struct FVector NewSourcePoint, int32_t
SourceIndex);
void SetBeamDistance(int32_t EmitterIndex, float Distance);
void SetBeamEndPoint(int32_t EmitterIndex, struct FVector NewEndPoint);
void SetBeamTessellationFactor(int32_t EmitterIndex, float NewFactor);
void SetBeamType(int32_t EmitterIndex, int32_t NewMethod);
void RewindEmitterInstances();
void RewindEmitterInstance(int32_t EmitterIndex);
void SetKillOnCompleted(int32_t EmitterIndex, unsigned long bKill);
void SetKillOnDeactivate(int32_t EmitterIndex, unsigned long bKill);
bool GetSkipBoundsUpdate();
void SetSkipBoundsUpdate(unsigned long bInSkipBoundsUpdate);
bool GetSkipUpdateDynamicDataDuringTick();
void SetSkipUpdateDynamicDataDuringTick(unsigned long
bInSkipUpdateDynamicDataDuringTick);
void KillParticlesInEmitter(struct FName InEmitterName);
void KillParticlesForced();
void DeactivateSystem();
void ActivateSystem(unsigned long bFlagAsJustAttached);
void SetTemplate(class UParticleSystem* NewTemplate);
void OnSystemFinished(class UParticleSystemComponent* PSystem);
};

// Class Engine.DistributionFloatParticleParameter
// 0x0007 (0x00A1 - 0x00A8)
class UDistributionFloatParticleParameter : public UDistributionFloatParameterBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatParticleParameter");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorParticleParameter
// 0x0005 (0x00D3 - 0x00D8)
class UDistributionVectorParticleParameter : public UDistributionVectorParameterBase
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionVectorParticleParameter");
}

return uClassPointer;
};

};

// Class Engine.DistributionVectorConstantCurveParticleParameter
// 0x001B (0x009D - 0x00B8)
class UDistributionVectorConstantCurveParticleParameter : public
UDistributionVectorConstantCurveBase
{
public:
TArray<struct FCurveParameterPoint>          Points;                       // 0x00A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
uint8_t                              InterpMethod;                    // 0x00B0 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bIsEditingCurve : 1;                 // 0x00B4 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.DistributionVectorConstantCurveParticleParameter");
}

return uClassPointer;
};

};

// Class Engine.ParticleEmitter
// 0x0038 (0x0060 - 0x0098)
class UParticleEmitter : public UObject
{
public:
struct FName                           EmitterName;                      // 0x0060 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                              SubUVDataOffset;                   // 0x0068 (0x0004)
[0x0000000000002000] (CPF_Transient)
uint8_t                              EmitterRenderMode;                   // 0x006C (0x0001)
[0x0000000000000001] (CPF_Edit)
```

```
struct FColor                         EmitterEditorColor;                    // 0x0070 (0x0004)
[0x0000000800000001] (CPF_Edit)
TArray<class UParticleLODLevel*>                LODLevels;                            // 0x0078 (0x0010)
[0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
unsigned long                         ConvertedModules : 1;                    // 0x0088 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                         bCollapsed : 1;                        // 0x0088 (0x0004)
[0x0000000800000001] [0x00000002] (CPF_Edit)
unsigned long                         bIsSoloing : 1;                       // 0x0088 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                         bCookedOut : 1;                        // 0x0088 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                         bDisableForLowIntensity : 1;               // 0x0088 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
int32_t                            PeakActiveParticles;                   // 0x008C (0x0004)
[0x0000000000000000]
int32_t                            InitialAllocationCount;                 // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                             MediumDetailSpawnRateScale;                // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleEmitter");
}

return uClassPointer;
};

float GetMaxLifespan(float InComponentDelay);
bool IsEmitterEnabled();
};

// Class Engine.ParticleSpriteEmitter
// 0x0000 (0x0098 - 0x0098)
class UParticleSpriteEmitter : public UParticleEmitter
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleSpriteEmitter");
}
return uClassPointer;
};

};

// Class Engine.ParticleLODLevel
// 0x008C (0x0060 - 0x00EC)
class UParticleLODLevel : public UObject
{
public:
int32_t                                   Level;                                  // 0x0060 (0x0004)
[0x0000000000000002] (CPF_Const)
unsigned long                             bEnabled : 1;                           // 0x0064 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                             ConvertedModules : 1;                   // 0x0064 (0x0004)
[0x0000000000000000] [0x00000002]
class UParticleModuleRequired*            RequiredModule;                         // 0x0068
(0x0008) [0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UParticleModule*>            Modules;                                // 0x0070 (0x0010)
[0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
class UParticleModule*                    TypeDataModule;                         // 0x0080 (0x0008)
[0x0000000000000008] (CPF_ExportObject)
class UParticleModuleSpawn*               SpawnModule;                            // 0x0088
(0x0008) [0x0000000000000008] (CPF_ExportObject)
class UParticleModuleEventGenerator*      EventGenerator;                         // 0x0090
(0x0008) [0x0000000000000008] (CPF_ExportObject)
TArray<class UParticleModuleSpawnBase*>   SpawningModules;                        // 0x0098
(0x0010) [0x0000000000001000] (CPF_Native)
TArray<class UParticleModule*>            SpawnModules;                           // 0x00A8
(0x0010) [0x0000000000001000] (CPF_Native)
TArray<class UParticleModule*>            UpdateModules;                          // 0x00B8
(0x0010) [0x0000000000001000] (CPF_Native)
TArray<class UParticleModuleOrbit*>       OrbitModules;                           // 0x00C8
(0x0010) [0x0000000000001000] (CPF_Native)
TArray<class UParticleModuleEventReceiverBase*>   EventReceiverModules;                   //
0x00D8 (0x0010) [0x0000000000001000] (CPF_Native)
int32_t                                   PeakActiveParticles;                    // 0x00E8 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleLODLevel");
}

return uClassPointer;
};

};
```

```
// Class Engine.ParticleModule
// 0x000C (0x0060 - 0x006C)
class UParticleModule : public UObject
{
public:
unsigned long                   bSpawnModule : 1;                   // 0x0060 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                   bUpdateModule : 1;                  // 0x0060 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                   bFinalUpdateModule : 1;              // 0x0060 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                   bCurvesAsColor : 1;                 // 0x0060 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                   b3DDrawMode : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                   bSupported3DDrawMode : 1;            // 0x0060 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                   bEnabled : 1;                       // 0x0060 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                   bEditable : 1;                      // 0x0060 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                   LODDuplicate : 1;                   // 0x0060 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                   bSupportsRandomSeed : 1;             // 0x0060 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                   bRequiresLoopingNotification : 1;     // 0x0060 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                   bRequiresUpdateInTick : 1;           // 0x0060 (0x0004)
[0x0000000000000000] [0x00000800]
uint8_t                      LODValidity;                      // 0x0064 (0x0001)
[0x0000000000000002] (CPF_Const)
struct FColor                  ModuleEditorColor;                 // 0x0068 (0x0004)
[0x0000000800000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModule");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAccelerationBase
// 0x0008 (0x006C - 0x0074)
class UParticleModuleAccelerationBase : public UParticleModule
{
public:
```

```
unsigned long                    bAlwaysInWorldSpace : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAccelerationBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAcceleration
// 0x0030 (0x0074 - 0x00A4)
class UParticleModuleAcceleration : public UParticleModuleAccelerationBase
{
public:
struct FRawDistributionVector                    Acceleration;                              // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                    bApplyOwnerScale : 1;                    // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAcceleration");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAccelerationOverLifetime
// 0x002C (0x0074 - 0x00A0)
class UParticleModuleAccelerationOverLifetime : public UParticleModuleAccelerationBase
{
public:
struct FRawDistributionVector                    AccelOverLife;                              // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAccelerationOverLifetime");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAttractorBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleAttractorBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAttractorBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAttractorBoneSocket
// 0x0128 (0x0070 - 0x0198)
class UParticleModuleAttractorBoneSocket : public UParticleModuleAttractorBase
{
public:
uint8_t                        FalloffType;                   // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                        DestinationType;                // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                        SelectionMethod;                // 0x0072 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                  bParticleLifeRelative : 1;          // 0x0074 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                  bAttractAlongLengthOfBone : 1;          // 0x0074 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FRawDistributionFloat          FalloffExponent;                 // 0x0078 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat          Range;                       // 0x00A0 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat          Strength;                     // 0x00C8 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
```

```
struct FRawDistributionFloat                    CollisionRadius;                            // 0x00F0 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    DragCoefficient;                            // 0x0118 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    DragRadius;                               // 0x0140 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FVector                              UniversalOffset;                          // 0x0168 (0x000C)
[0x0000000000000001] (CPF_Edit)
TArray<struct FAttractLocationBoneSocketInfo>      SourceLocations;                              // 0x0178
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FName                                SkelMeshActorParamName;                     // 0x0188 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USkeletalMesh*                            EditorSkelMesh;                             // 0x0190 (0x0008)
[0x0000000800000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAttractorBoneSocket");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAttractorLine
// 0x0068 (0x0070 - 0x00D8)
class UParticleModuleAttractorLine : public UParticleModuleAttractorBase
{
public:
struct FVector                              EndPoint0;                                // 0x0070 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                              EndPoint1;                                // 0x007C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                    Range;                                    // 0x0088 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    Strength;                                 // 0x00B0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAttractorLine");
}
return uClassPointer;
};

};

// Class Engine.ParticleModuleAttractorParticle
// 0x0068 (0x0070 - 0x00D8)
class UParticleModuleAttractorParticle : public UParticleModuleAttractorBase
{
public:
struct FName                          EmitterName;                    // 0x0070 (0x0008)
[0x0000000002000009] (CPF_Edit | CPF_ExportObject | CPF_NoClear)
struct FRawDistributionFloat               Range;                       // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                         bStrengthByDistance : 1;           // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bAffectBaseVelocity : 1;           // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                         bRenewSource : 1;                 // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                         bInheritSourceVel : 1;             // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FRawDistributionFloat               Strength;                    // 0x00A8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
uint8_t                            SelectionMethod;                  // 0x00D0 (0x0001)
[0x0000000000000001] (CPF_Edit)
int32_t                            LastSelIndex;                   // 0x00D4 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAttractorParticle");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAttractorPoint
// 0x007C (0x0070 - 0x00EC)
class UParticleModuleAttractorPoint : public UParticleModuleAttractorBase
{
public:
struct FRawDistributionVector              Position;                    // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat               Range;                       // 0x0098 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat               Strength;                    // 0x00C0 (0x0028)
```

```
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                          StrengthByDistance : 1;                    // 0x00E8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bAffectBaseVelocity : 1;                   // 0x00E8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bOverrideVelocity : 1;                    // 0x00E8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bUseWorldSpacePosition : 1;                // 0x00E8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAttractorPoint");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleAttractorSkelVertSurface
// 0x0150 (0x0070 - 0x01C0)
class UParticleModuleAttractorSkelVertSurface : public UParticleModuleAttractorBase
{
public:
uint8_t                             FalloffType;                        // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                             DestinationType;                      // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bParticleLifeRelative : 1;                 // 0x0074 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bEnforceNormalCheck : 1;                   // 0x0074 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FRawDistributionFloat                 FalloffExponent;                    // 0x0078 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 Range;                          // 0x00A0 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 Strength;                        // 0x00C8 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 CollisionRadius;                    // 0x00F0 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 DragCoefficient;                    // 0x0118 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 DragRadius;                       // 0x0140 (0x0028)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FVector                        UniversalOffset;                    // 0x0168 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FName                          SkelMeshActorParamName;                 // 0x0174 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
class USkeletalMesh*                    EditorSkelMesh;                    // 0x0180 (0x0008)
[0x0000000800000001] (CPF_Edit)
TArray<struct FName>                    ValidAssociatedBones;                    // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FVector                    NormalToCompare;                    // 0x0198 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                    NormalCheckToleranceDegrees;                    // 0x01A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                    NormalCheckTolerance;                    // 0x01A8 (0x0004)
[0x0000000000000000]
TArray<int32_t>                    ValidMaterialIndices;                    // 0x01B0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleAttractorSkelVertSurface");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleBeamBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleBeamBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleBeamBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleBeamModifier
// 0x0090 (0x0070 - 0x0100)
class UParticleModuleBeamModifier : public UParticleModuleBeamBase
{
public:
```

```
uint8_t                              ModifierType;                              // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FBeamModifierOptions                    PositionOptions;                              // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                  Position;                                     // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FBeamModifierOptions                    TangentOptions;                               // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                  Tangent;                                      // 0x00A8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                          bAbsoluteTangent : 1;                    // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FBeamModifierOptions                    StrengthOptions;                              // 0x00D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                   Strength;                                     // 0x00D8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleBeamModifier");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleBeamNoise
// 0x00F0 (0x0070 - 0x0160)
class UParticleModuleBeamNoise : public UParticleModuleBeamBase
{
public:
unsigned long                          bLowFreq_Enabled : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bNRScaleEmitterTime : 1;                   // 0x0070 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bSmooth : 1;                             // 0x0070 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bNoiseLock : 1;                          // 0x0070 (0x0004)
[0x0000000000000002] [0x00000008] (CPF_Const)
unsigned long                          bOscillate : 1;                          // 0x0070 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bUseNoiseTangents : 1;                     // 0x0070 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bTargetNoise : 1;                        // 0x0070 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          bApplyNoiseScale : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
int32_t                              Frequency;                               // 0x0074 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
int32_t                          Frequency_LowRange;                    // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                 NoiseRange;                            // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                  NoiseRangeScale;                        // 0x00A8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                 NoiseSpeed;                            // 0x00D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
float                            NoiseLockRadius;                       // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            NoiseLockTime;                         // 0x00FC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            NoiseTension;                          // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                  NoiseTangentStrength;                  // 0x0108
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
int32_t                          NoiseTessellation;                     // 0x0130 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            FrequencyDistance;                     // 0x0134 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                  NoiseScale;                            // 0x0138 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleBeamNoise");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleBeamSource
// 0x0088 (0x0070 - 0x00F8)
class UParticleModuleBeamSource : public UParticleModuleBeamBase
{
public:
uint8_t                          SourceMethod;                          // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          SourceTangentMethod;                   // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                          SourceName;                            // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                         bSourceAbsolute : 1;                    // 0x007C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bLockSource : 1;                        // 0x007C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
```

```
unsigned long                          bLockSourceTangent : 1;                    // 0x007C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bLockSourceStength : 1;                    // 0x007C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FRawDistributionVector                Source;                              // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                SourceTangent;                           // 0x00A8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                SourceStrength;                           // 0x00D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleBeamSource");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleBeamTarget
// 0x009C (0x0070 - 0x010C)
class UParticleModuleBeamTarget : public UParticleModuleBeamBase
{
public:
uint8_t                             TargetMethod;                         // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                             TargetTangentMethod;                      // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                        TargetName;                           // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                Target;                              // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                          bTargetAbsolute : 1;                   // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bLockTarget : 1;                    // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bLockTargetTangent : 1;                   // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bLockTargetStength : 1;                   // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FRawDistributionVector                TargetTangent;                           // 0x00B0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FName                        TargetBone;                           // 0x00D8 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                TargetStrength;                           // 0x00E0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
float                               LockRadius;                           // 0x0108 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleBeamTarget");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleCameraBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleCameraBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleCameraBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleCameraOffset
// 0x002D (0x0070 - 0x009D)
class UParticleModuleCameraOffset : public UParticleModuleCameraBase
{
public:
struct FRawDistributionFloat                    CameraOffset;                           // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                   bSpawnTimeOnly : 1;                     // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
uint8_t                                         UpdateMethod;                           // 0x009C (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleCameraOffset");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleCollisionBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleCollisionBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleCollisionBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleCollision
// 0x0118 (0x0070 - 0x0188)
class UParticleModuleCollision : public UParticleModuleCollisionBase
{
public:
struct FRawDistributionVector                  DampingFactor;                              // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                  DampingFactorRotation;                      // 0x0098
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                   MaxCollisions;                              // 0x00C0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
uint8_t                                        CollisionCompletionOption;                  // 0x00E8 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                                  bApplyPhysics : 1;                          // 0x00EC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                  bNoClip : 1;                                // 0x00EC (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                  bPawnsDoNotDecrementCount : 1;              // 0x00EC
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                  bOnlyVerticalNormalsDecrementCount : 1;     // 0x00EC
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                                  bDropDetail : 1;                            // 0x00EC (0x0004)
```

```
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bCollideOnlyIfVisible : 1;                  // 0x00EC (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bCollideWithWorld : 1;                      // 0x00EC (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          bCollideWithWorldAttractors : 1;            // 0x00EC (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
class UMaterialInterface*                  DecalMaterial;                          // 0x00F0 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                 DecalWidth;                              // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 DecalHeight;                             // 0x00FC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 DecalThickness;                           // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 DecalLifetime;                            // 0x0104 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 DecalDepthBias;                           // 0x0108 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                        DecalBlendRange;                         // 0x010C (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                 ParticleMass;                          // 0x0118 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
float                                 DirScalar;                              // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 VerticalFudgeFactor;                      // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                 DelayAmount;                           // 0x0148 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
float                                 MaxCollisionDistance;                      // 0x0170 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FParticleAttractorCollisionAction>   ParticleAttractorCollisionActions;           //
0x0178 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleCollision");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleCollisionActor
// 0x0014 (0x0188 - 0x019C)
class UParticleModuleCollisionActor : public UParticleModuleCollision
{
public:
```

```
TArray<struct FName>                    ActorsToCollideWith;                    // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                    bCheckPawnCollisions : 1;                    // 0x0198 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleCollisionActor");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleColorBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleColorBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleColorBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleColor
// 0x0054 (0x0070 - 0x00C4)
class UParticleModuleColor : public UParticleModuleColorBase
{
public:
struct FRawDistributionVector                    StartColor;                    // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    StartAlpha;                    // 0x0098 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                    bClampAlpha : 1;                    // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleColor");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleColor_Seeded
// 0x0024 (0x00C4 - 0x00E8)
class UParticleModuleColor_Seeded : public UParticleModuleColor
{
public:
struct FParticleRandomSeedInfo                  RandomSeedInfo;                          // 0x00C8
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleColor_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleColorByParameter
// 0x000C (0x0070 - 0x007C)
class UParticleModuleColorByParameter : public UParticleModuleColorBase
{
public:
struct FName                                    ColorParam;                              // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FColor                                   DefaultColor;                            // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleColorByParameter");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleColorOverLife
// 0x0054 (0x0070 - 0x00C4)
class UParticleModuleColorOverLife : public UParticleModuleColorBase
{
public:
struct FRawDistributionVector                    ColorOverLife;                             // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                     AlphaOverLife;                             // 0x0098 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                    bClampAlpha : 1;                           // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleColorOverLife");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleColorScaleOverDensity
// 0x0050 (0x0070 - 0x00C0)
class UParticleModuleColorScaleOverDensity : public UParticleModuleColorBase
{
public:
struct FRawDistributionVector                    ColorScaleOverDensity;                     // 0x0070
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                     AlphaScaleOverDensity;                     // 0x0098
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleColorScaleOverDensity");
}
```

```
return uClassPointer;
};

};

// Class Engine.ParticleModuleColorScaleOverLife
// 0x0054 (0x0070 - 0x00C4)
class UParticleModuleColorScaleOverLife : public UParticleModuleColorBase
{
public:
struct FRawDistributionVector                    ColorScaleOverLife;                          // 0x0070
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                     AlphaScaleOverLife;                          // 0x0098 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                    bEmitterTime : 1;                            // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleColorScaleOverLife");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleEventBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleEventBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleEventBase");
}

return uClassPointer;
};

};
```

```
// Class Engine.ParticleModuleEventGenerator
// 0x0010 (0x0070 - 0x0080)
class UParticleModuleEventGenerator : public UParticleModuleEventBase
{
public:
TArray<struct FParticleEvent_GenerateInfo>        Events;                                   // 0x0070
(0x0010) [0x0000000002400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink |
CPF_NoClear)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleEventGenerator");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleEventReceiverBase
// 0x000C (0x0070 - 0x007C)
class UParticleModuleEventReceiverBase : public UParticleModuleEventBase
{
public:
uint8_t                                           EventGeneratorType;                       // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                                      EventName;                                // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleEventReceiverBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleEventReceiverKillParticles
// 0x0008 (0x007C - 0x0084)
class UParticleModuleEventReceiverKillParticles : public UParticleModuleEventReceiverBase
{
public:
```

```
unsigned long                         bStopSpawning : 1;                        // 0x0080 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleEventReceiverKillParticles");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleEventReceiverSpawn
// 0x005C (0x007C - 0x00D8)
class UParticleModuleEventReceiverSpawn : public UParticleModuleEventReceiverBase
{
public:
struct FRawDistributionFloat                  SpawnCount;                              // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                         bUseParticleTime : 1;                     // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bUsePSysLocation : 1;                     // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                         bInheritVelocity : 1;                    // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FRawDistributionVector                  InheritVelocityScale;                    // 0x00B0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleEventReceiverSpawn");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleKillBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleKillBase : public UParticleModule
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleKillBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleKillBox
// 0x0054 (0x0070 - 0x00C4)
class UParticleModuleKillBox : public UParticleModuleKillBase
{
public:
struct FRawDistributionVector                  LowerLeftCorner;                          // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                  UpperRightCorner;                         // 0x0098 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                  bAbsolute : 1;                            // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                  bKillInside : 1;                          // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                  bAxisAlignedAndFixedSize : 1;             // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleKillBox");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleKillHeight
// 0x002C (0x0070 - 0x009C)
class UParticleModuleKillHeight : public UParticleModuleKillBase
{
public:
struct FRawDistributionFloat                   Height;                                   // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
```

```
unsigned long                          bAbsolute : 1;                         // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bFloor : 1;                          // 0x0098 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bApplyPSysScale : 1;                   // 0x0098 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleKillHeight");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLifetimeBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleLifetimeBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLifetimeBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLifetime
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleLifetime : public UParticleModuleLifetimeBase
{
public:
struct FRawDistributionFloat                 LifeTime;                            // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLifetime");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLifetime_Seeded
// 0x0020 (0x0098 - 0x00B8)
class UParticleModuleLifetime_Seeded : public UParticleModuleLifetime
{
public:
struct FParticleRandomSeedInfo                    RandomSeedInfo;                              // 0x0098
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLifetime_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleLocationBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationBase");
}

return uClassPointer;
};
```

```
};

// Class Engine.ParticleModuleLocation
// 0x0030 (0x0070 - 0x00A0)
class UParticleModuleLocation : public UParticleModuleLocationBase
{
public:
struct FRawDistributionVector                    StartLocation;                              // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
float                                            DistributeOverNPoints;                      // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                            DistributeThreshold;                        // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocation");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocation_Seeded
// 0x0020 (0x00A0 - 0x00C0)
class UParticleModuleLocation_Seeded : public UParticleModuleLocation
{
public:
struct FParticleRandomSeedInfo                   RandomSeedInfo;                             // 0x00A0
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocation_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationWorldOffset
// 0x0000 (0x00A0 - 0x00A0)
```

```
class UParticleModuleLocationWorldOffset : public UParticleModuleLocation
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationWorldOffset");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationWorldOffset_Seeded
// 0x0020 (0x00A0 - 0x00C0)
class UParticleModuleLocationWorldOffset_Seeded : public
UParticleModuleLocationWorldOffset
{
public:
struct FParticleRandomSeedInfo          RandomSeedInfo;                    // 0x00A0
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationWorldOffset_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationBoneSocket
// 0x003C (0x0070 - 0x00AC)
class UParticleModuleLocationBoneSocket : public UParticleModuleLocationBase
{
public:
uint8_t                                 SourceType;                        // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 SelectionMethod;                   // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                          UniversalOffset;                   // 0x0074 (0x000C)
[0x0000000000000001] (CPF_Edit)
```

```
TArray<struct FLocationBoneSocketInfo>          SourceLocations;                        // 0x0080
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                          bUpdatePositionEachFrame : 1;            // 0x0090 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bOrientMeshEmitters : 1;                 // 0x0090 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bInheritVelocityAtSpawn : 1;             // 0x0090 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FName                           SkelMeshActorParamName;                  // 0x0094 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USkeletalMesh*                       EditorSkelMesh;                      // 0x00A0 (0x0008)
[0x0000000800000001] (CPF_Edit)
float                            LastUpdateTime;                      // 0x00A8 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationBoneSocket");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationDirect
// 0x00A0 (0x0070 - 0x0110)
class UParticleModuleLocationDirect : public UParticleModuleLocationBase
{
public:
struct FRawDistributionVector              Location;                        // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector              LocationOffset;                    // 0x0098 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector              ScaleFactor;                      // 0x00C0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector              Direction;                       // 0x00E8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationDirect");
}
```

```
return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationEmitter
// 0x0018 (0x0070 - 0x0088)
class UParticleModuleLocationEmitter : public UParticleModuleLocationBase
{
public:
struct FName                              EmitterName;                              // 0x0070 (0x0008)
[0x0000000002000009] (CPF_Edit | CPF_ExportObject | CPF_NoClear)
uint8_t                              SelectionMethod;                              // 0x0078 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                              InheritSourceVelocity : 1;                              // 0x007C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                              bInheritSourceRotation : 1;                              // 0x007C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                              InheritSourceVelocityScale;                              // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              InheritSourceRotationScale;                              // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationEmitter");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationEmitterDirect
// 0x0008 (0x0070 - 0x0078)
class UParticleModuleLocationEmitterDirect : public UParticleModuleLocationBase
{
public:
struct FName                              EmitterName;                              // 0x0070 (0x0008)
[0x0000000002000009] (CPF_Edit | CPF_ExportObject | CPF_NoClear)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationEmitterDirect");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationPrimitiveBase
// 0x0058 (0x0070 - 0x00C8)
class UParticleModuleLocationPrimitiveBase : public UParticleModuleLocationBase
{
public:
unsigned long                          Positive_X : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          Positive_Y : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          Positive_Z : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          Negative_X : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          Negative_Y : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          Negative_Z : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          SurfaceOnly : 1;                   // 0x0070 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          Velocity : 1;                      // 0x0070 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
struct FRawDistributionFloat                 VelocityScale;                    // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                 StartLocation;                    // 0x00A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationPrimitiveBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationPrimitiveCylinder
// 0x0059 (0x00C8 - 0x0121)
class UParticleModuleLocationPrimitiveCylinder : public UParticleModuleLocationPrimitiveBase
{
public:
unsigned long                          RadialVelocity : 1;                // 0x00C8 (0x0004)
```

```
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bAdjustForWorldSpace : 1;           // 0x00C8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FRawDistributionFloat              StartRadius;                    // 0x00D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat              StartHeight;                    // 0x00F8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
uint8_t                          HeightAxis;                    // 0x0120 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationPrimitiveCylinder");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationPrimitiveCylinder_Seeded
// 0x0027 (0x0121 - 0x0148)
class UParticleModuleLocationPrimitiveCylinder_Seeded : public
UParticleModuleLocationPrimitiveCylinder
{
public:
struct FParticleRandomSeedInfo             RandomSeedInfo;                  // 0x0128
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ParticleModuleLocationPrimitiveCylinder_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationPrimitiveSphere
// 0x0028 (0x00C8 - 0x00F0)
class UParticleModuleLocationPrimitiveSphere : public UParticleModuleLocationPrimitiveBase
{
```

```
public:
struct FRawDistributionFloat                    StartRadius;                              // 0x00C8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationPrimitiveSphere");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationPrimitiveSphere_Seeded
// 0x0020 (0x00F0 - 0x0110)
class UParticleModuleLocationPrimitiveSphere_Seeded : public
UParticleModuleLocationPrimitiveSphere
{
public:
struct FParticleRandomSeedInfo                  RandomSeedInfo;                           // 0x00F0
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ParticleModuleLocationPrimitiveSphere_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationSkelVertSurface
// 0x0060 (0x0070 - 0x00D0)
class UParticleModuleLocationSkelVertSurface : public UParticleModuleLocationBase
{
public:
uint8_t                                         SourceType;                               // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                                  UniversalOffset;                          // 0x0074 (0x000C)
[0x0000000000000001] (CPF_Edit)
unsigned long                                   bUpdatePositionEachFrame : 1;             // 0x0080 (0x0004)
```

```
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bOrientMeshEmitters : 1;              // 0x0080 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bEnforceNormalCheck : 1;              // 0x0080 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FName                     SkelMeshActorParamName;               // 0x0084 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USkeletalMesh*                 EditorSkelMesh;                   // 0x0090 (0x0008)
[0x0000000800000001] (CPF_Edit)
TArray<struct FName>                 ValidAssociatedBones;             // 0x0098 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FVector                   NormalToCompare;                      // 0x00A8 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                        NormalCheckToleranceDegrees;              // 0x00B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        NormalCheckTolerance;                     // 0x00B8 (0x0004)
[0x0000000000000000]
TArray<int32_t>                   ValidMaterialIndices;                // 0x00C0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationSkelVertSurface");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleLocationStaticVertSurface
// 0x0050 (0x0070 - 0x00C0)
class UParticleModuleLocationStaticVertSurface : public UParticleModuleLocationBase
{
public:
uint8_t                        SourceType;                            // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                   UniversalOffset;                     // 0x0074 (0x000C)
[0x0000000000000001] (CPF_Edit)
unsigned long                    bUpdatePositionEachFrame : 1;         // 0x0080 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bOrientMeshEmitters : 1;              // 0x0080 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bEnforceNormalCheck : 1;              // 0x0080 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FName                     StaticMeshActorParamName;             // 0x0084 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UStaticMesh*                  EditorStaticMesh;                  // 0x0090 (0x0008)
[0x0000000800000001] (CPF_Edit)
```

```
struct FVector                              NormalToCompare;                          // 0x0098 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                  NormalCheckToleranceDegrees;                // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  NormalCheckTolerance;                     // 0x00A8 (0x0004)
[0x0000000000000000]
TArray<int32_t>                             ValidMaterialIndices;                     // 0x00B0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleLocationStaticVertSurface");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSourceMovement
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleSourceMovement : public UParticleModuleLocationBase
{
public:
struct FRawDistributionVector                  SourceMovementScale;                      // 0x0070
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSourceMovement");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMaterialBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleMaterialBase : public UParticleModule
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMaterialBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMaterialByParameter
// 0x0020 (0x0070 - 0x0090)
class UParticleModuleMaterialByParameter : public UParticleModuleMaterialBase
{
public:
TArray<struct FName>                            MaterialParameters;                         // 0x0070 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UMaterialInterface*>               DefaultMaterials;                           // 0x0080
(0x0010) [0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMaterialByParameter");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMeshMaterial
// 0x0010 (0x0070 - 0x0080)
class UParticleModuleMeshMaterial : public UParticleModuleMaterialBase
{
public:
TArray<class UMaterialInterface*>               MeshMaterials;                              // 0x0070
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMeshMaterial");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleOrbitBase
// 0x0008 (0x006C - 0x0074)
class UParticleModuleOrbitBase : public UParticleModule
{
public:
unsigned long                         bUseEmitterTime : 1;                      // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleOrbitBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleOrbit
// 0x0098 (0x0074 - 0x010C)
class UParticleModuleOrbit : public UParticleModuleOrbitBase
{
public:
uint8_t                               ChainMode;                                // 0x0078 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                OffsetAmount;                              // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FOrbitOptions                         OffsetOptions;                           // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                RotationAmount;                            // 0x00B0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FOrbitOptions                         RotationOptions;                          // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                RotationRateAmount;                        // 0x00E0
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FOrbitOptions                         RotationRateOptions;                      // 0x0108 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleOrbit");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleOrientationBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleOrientationBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleOrientationBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleOrientationAxisLock
// 0x0001 (0x0070 - 0x0071)
class UParticleModuleOrientationAxisLock : public UParticleModuleOrientationBase
{
public:
uint8_t                                  LockAxisFlags;                              // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleOrientationAxisLock");
}

return uClassPointer;
};
```

```
};

// Class Engine.ParticleModuleParameterBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleParameterBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleParameterBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleParameterDynamic
// 0x0018 (0x0070 - 0x0088)
class UParticleModuleParameterDynamic : public UParticleModuleParameterBase
{
public:
TArray<struct FEmitterDynamicParameter>          DynamicParams;                            // 0x0070
(0x0010) [0x0000000000480041] (CPF_Edit | CPF_EditConstArray | CPF_Component |
CPF_NeedCtorLink)
int32_t                                UpdateFlags;                            // 0x0080 (0x0004)
[0x0000000000000000]
unsigned long                             bUsesVelocity : 1;                       // 0x0084 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleParameterDynamic");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleParameterDynamic_Seeded
// 0x0020 (0x0088 - 0x00A8)
class UParticleModuleParameterDynamic_Seeded : public UParticleModuleParameterDynamic
```

```
{
public:
struct FParticleRandomSeedInfo                RandomSeedInfo;                          // 0x0088
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleParameterDynamic_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleRequired
// 0x00B4 (0x006C - 0x0120)
class UParticleModuleRequired : public UParticleModule
{
public:
class UMaterialInterface*                    Material;                              // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                                ScreenAlignment;                        // 0x0078 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                SortMode;                             // 0x0079 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                ParticleBurstMethod;                    // 0x007A (0x0001)
[0x0000000000000000]
uint8_t                                InterpolationMethod;                    // 0x007B (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                EmitterNormalsMode;                     // 0x007C (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bAllowImageFlipping : 1;                  // 0x0080 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bSquareImageFlipping : 1;                 // 0x0080 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                            bUseLocalSpace : 1;                     // 0x0080 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                            bKillOnDeactivate : 1;                  // 0x0080 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                            bKillOnCompleted : 1;                   // 0x0080 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                            bRequiresSorting : 1;                   // 0x0080 (0x0004)
[0x0000000020000000] [0x00000020]  CPF_Deprecated)
unsigned long                            bUseLegacyEmitterTime : 1;                // 0x0080 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                            bEnableNearParticleCulling : 1;             // 0x0080 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                            bEnableFarParticleCulling : 1;             // 0x0080 (0x0004)
```

```
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                    bEmitterDurationUseRange : 1;           // 0x0080 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                    bDurationRecalcEachLoop : 1;            // 0x0080 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                    bEmitterDelayUseRange : 1;              // 0x0080 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                    bDelayFirstLoopOnly : 1;                // 0x0080 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                    bScaleUV : 1;                           // 0x0080 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                    bDirectUV : 1;                          // 0x0080 (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                    bOverrideSystemMacroUV : 1;             // 0x0080 (0x0004)
[0x0000000000000001] [0x00008000] (CPF_Edit)
unsigned long                    bUseMaxDrawCount : 1;                   // 0x0080 (0x0004)
[0x0000000000000001] [0x00010000] (CPF_Edit)
unsigned long                    bOrbitModuleAffectsVelocityAlignment : 1;     // 0x0080
(0x0004) [0x0000000000000001] [0x00020000] (CPF_Edit)
float                       NearCullDistance;                       // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       NearFadeDistance;                       // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       FarFadeDistance;                        // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       FarCullDistance;                        // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       EmitterDuration;                        // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       EmitterDurationLow;                     // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                      EmitterLoops;                          // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat           SpawnRate;                        // 0x00A0 (0x0028)
[0x0000000000480000] (CPF_Component | CPF_NeedCtorLink)
TArray<struct FParticleBurst>          BurstList;                      // 0x00C8 (0x0010)
[0x0000000002400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_NoClear)
float                       EmitterDelay;                          // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       EmitterDelayLow;                        // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                      SubImages_Horizontal;                    // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                      SubImages_Vertical;                     // 0x00E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       RandomImageTime;                        // 0x00E8 (0x0004)
[0x0000000000000000]
int32_t                      RandomImageChanges;                      // 0x00EC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                   MacroUVPosition;                     // 0x00F0 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                       MacroUVRadius;                          // 0x00FC (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                      MaxDrawCount;                           // 0x0100 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                              DownsampleThresholdScreenFraction;            // 0x0104 (0x0004)
[0x0000000000000000]
struct FVector                              NormalsSphereCenter;                      // 0x0108 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                              NormalsCylinderDirection;                  // 0x0114 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRequired");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleRotationBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleRotationBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotationBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMeshRotation
// 0x002C (0x0070 - 0x009C)
class UParticleModuleMeshRotation : public UParticleModuleRotationBase
{
public:
struct FRawDistributionVector                   StartRotation;                             // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                              bInheritParent : 1;                       // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMeshRotation");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMeshRotation_Seeded
// 0x0024 (0x009C - 0x00C0)
class UParticleModuleMeshRotation_Seeded : public UParticleModuleMeshRotation
{
public:
struct FParticleRandomSeedInfo                  RandomSeedInfo;                           // 0x00A0
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMeshRotation_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleRotation
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleRotation : public UParticleModuleRotationBase
{
public:
struct FRawDistributionFloat                    StartRotation;                            // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotation");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleRotation_Seeded
// 0x0020 (0x0098 - 0x00B8)
class UParticleModuleRotation_Seeded : public UParticleModuleRotation
{
public:
struct FParticleRandomSeedInfo                    RandomSeedInfo;                              // 0x0098
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotation_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleRotationOverLifetime
// 0x002C (0x0070 - 0x009C)
class UParticleModuleRotationOverLifetime : public UParticleModuleRotationBase
{
public:
struct FRawDistributionFloat                      RotationOverLife;                            // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                     Scale : 1;                                   // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotationOverLifetime");
}

return uClassPointer;
};

};
```

```
// Class Engine.ParticleModuleRotationRateBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleRotationRateBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotationRateBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMeshRotationRate
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleMeshRotationRate : public UParticleModuleRotationRateBase
{
public:
struct FRawDistributionVector                    StartRotationRate;                          // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMeshRotationRate");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMeshRotationRate_Seeded
// 0x0020 (0x0098 - 0x00B8)
class UParticleModuleMeshRotationRate_Seeded : public UParticleModuleMeshRotationRate
{
public:
struct FParticleRandomSeedInfo                   RandomSeedInfo;                             // 0x0098
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMeshRotationRate_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMeshRotationRateMultiplyLife
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleMeshRotationRateMultiplyLife : public UParticleModuleRotationRateBase
{
public:
struct FRawDistributionVector                    LifeMultiplier;                              // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.ParticleModuleMeshRotationRateMultiplyLife");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleMeshRotationRateOverLife
// 0x002C (0x0070 - 0x009C)
class UParticleModuleMeshRotationRateOverLife : public UParticleModuleRotationRateBase
{
public:
struct FRawDistributionVector                    RotRate;                                     // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                          bScaleRotRate : 1;                        // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleMeshRotationRateOverLife");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleRotationRate
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleRotationRate : public UParticleModuleRotationRateBase
{
public:
struct FRawDistributionFloat                    StartRotationRate;                          // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotationRate");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleRotationRate_Seeded
// 0x0020 (0x0098 - 0x00B8)
class UParticleModuleRotationRate_Seeded : public UParticleModuleRotationRate
{
public:
struct FParticleRandomSeedInfo                  RandomSeedInfo;                             // 0x0098
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotationRate_Seeded");
}

return uClassPointer;
};

};
```

```
// Class Engine.ParticleModuleRotationRateMultiplyLife
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleRotationRateMultiplyLife : public UParticleModuleRotationRateBase
{
public:
struct FRawDistributionFloat                    LifeMultiplier;                          // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleRotationRateMultiplyLife");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSizeBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleSizeBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSizeBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSize
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleSize : public UParticleModuleSizeBase
{
public:
struct FRawDistributionVector                   StartSize;                               // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSize");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSize_Seeded
// 0x0020 (0x0098 - 0x00B8)
class UParticleModuleSize_Seeded : public UParticleModuleSize
{
public:
struct FParticleRandomSeedInfo                    RandomSeedInfo;                              // 0x0098
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSize_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSizeMultiplyLife
// 0x002C (0x0070 - 0x009C)
class UParticleModuleSizeMultiplyLife : public UParticleModuleSizeBase
{
public:
struct FRawDistributionVector                    LifeMultiplier;                              // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                    MultiplyX : 1;                               // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                    MultiplyY : 1;                               // 0x0098 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                    MultiplyZ : 1;                               // 0x0098 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSizeMultiplyLife");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSizeMultiplyVelocity
// 0x0044 (0x0070 - 0x00B4)
class UParticleModuleSizeMultiplyVelocity : public UParticleModuleSizeBase
{
public:
struct FRawDistributionVector                VelocityMultiplier;                     // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                            MultiplyX : 1;                       // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            MultiplyY : 1;                       // 0x0098 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                            MultiplyZ : 1;                       // 0x0098 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FVector                           CapMaxSize;                           // 0x009C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                           CapMinSize;                          // 0x00A8 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSizeMultiplyVelocity");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSizeScale
// 0x002C (0x0070 - 0x009C)
class UParticleModuleSizeScale : public UParticleModuleSizeBase
{
public:
struct FRawDistributionVector                SizeScale;                            // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                            EnableX : 1;                         // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
unsigned long                          EnableY : 1;                              // 0x0098 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          EnableZ : 1;                              // 0x0098 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSizeScale");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSizeScaleByTime
// 0x002C (0x0070 - 0x009C)
class UParticleModuleSizeScaleByTime : public UParticleModuleSizeBase
{
public:
struct FRawDistributionVector                SizeScaleByTime;                        // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                          bEnableX : 1;                             // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bEnableY : 1;                             // 0x0098 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bEnableZ : 1;                             // 0x0098 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSizeScaleByTime");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSizeScaleOverDensity
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleSizeScaleOverDensity : public UParticleModuleSizeBase
{
public:
```

```
struct FRawDistributionVector                    SizeScaleOverDensity;                    // 0x0070
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSizeScaleOverDensity");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSpawnBase
// 0x0008 (0x006C - 0x0074)
class UParticleModuleSpawnBase : public UParticleModule
{
public:
unsigned long                                bProcessSpawnRate : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                bProcessBurstList : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSpawnBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSpawn
// 0x006C (0x0074 - 0x00E0)
class UParticleModuleSpawn : public UParticleModuleSpawnBase
{
public:
struct FRawDistributionFloat                    Rate;                                    // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    RateScale;                               // 0x00A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
uint8_t                                    ParticleBurstMethod;                        // 0x00C8 (0x0001)
[0x0000000000000001] (CPF_Edit)
```

```
TArray<struct FParticleBurst>                  BurstList;                              // 0x00D0 (0x0010)
[0x0000000002400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_NoClear)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSpawn");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSpawnPerUnit
// 0x0044 (0x0074 - 0x00B8)
class UParticleModuleSpawnPerUnit : public UParticleModuleSpawnBase
{
public:
float                                          UnitScalar;                             // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                   SpawnPerUnit;                           // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                  bIgnoreSpawnRateWhenMoving : 1;              // 0x00A8
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                  bIgnoreMovementAlongX : 1;                   // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                  bIgnoreMovementAlongY : 1;                   // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                  bIgnoreMovementAlongZ : 1;                   // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
float                                          MovementTolerance;                      // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                          MaxFrameDistance;                       // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                          MinFrameDistance;                       // 0x00B4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSpawnPerUnit");
}

return uClassPointer;
};
```

```
};

// Class Engine.ParticleModuleStoreSpawnTimeBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleStoreSpawnTimeBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleStoreSpawnTimeBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleStoreSpawnTime
// 0x0000 (0x0070 - 0x0070)
class UParticleModuleStoreSpawnTime : public UParticleModuleStoreSpawnTimeBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleStoreSpawnTime");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSubUVBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleSubUVBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSubUVBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSubUV
// 0x002C (0x0070 - 0x009C)
class UParticleModuleSubUV : public UParticleModuleSubUVBase
{
public:
struct FRawDistributionFloat                    SubImageIndex;                           // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                   bUseRealTime : 1;                        // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSubUV");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSubUVMovie
// 0x0038 (0x009C - 0x00D4)
class UParticleModuleSubUVMovie : public UParticleModuleSubUV
{
public:
unsigned long                                   bUseEmitterTime : 1;                     // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FRawDistributionFloat                    FrameRate;                               // 0x00A8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
int32_t                                         StartingFrame;                           // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSubUVMovie");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSubUVDirect
// 0x0050 (0x0070 - 0x00C0)
class UParticleModuleSubUVDirect : public UParticleModuleSubUVBase
{
public:
struct FRawDistributionVector                    SubUVPosition;                              // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                    SubUVSize;                                  // 0x0098 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSubUVDirect");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleSubUVSelect
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleSubUVSelect : public UParticleModuleSubUVBase
{
public:
struct FRawDistributionVector                    SubImageSelect;                             // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleSubUVSelect");
}

return uClassPointer;
```

```
};

};

// Class Engine.ParticleModuleTrailBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleTrailBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTrailBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTrailSource
// 0x0050 (0x0070 - 0x00C0)
class UParticleModuleTrailSource : public UParticleModuleTrailBase
{
public:
uint8_t                                  SourceMethod;                              // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  SelectionMethod;                           // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FName                                  SourceName;                              // 0x0074 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                    SourceStrength;                            // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                 bLockSourceStength : 1;                    // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                 bInheritRotation : 1;                      // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
int32_t                                  SourceOffsetCount;                          // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FVector>                        SourceOffsetDefaults;                      // 0x00B0 (0x0010)
[0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTrailSource");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTrailSpawn
// 0x000C (0x0070 - 0x007C)
class UParticleModuleTrailSpawn : public UParticleModuleTrailBase
{
public:
class UDistributionFloatParticleParameter*        SpawnDistanceMap;                          // 0x0070
(0x0008) [0x0000000006080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_NoClear | CPF_EditInline)
float                                    MinSpawnVelocity;                          // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTrailSpawn");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTrailTaper
// 0x0030 (0x0070 - 0x00A0)
class UParticleModuleTrailTaper : public UParticleModuleTrailBase
{
public:
uint8_t                                   TaperMethod;                               // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                  TaperFactor;                               // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTrailTaper");
}
```

```
return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleTypeDataBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataAnimTrail
// 0x001C (0x0070 - 0x008C)
class UParticleModuleTypeDataAnimTrail : public UParticleModuleTypeDataBase
{
public:
struct FName                                ControlEdgeName;                          // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                                 SheetsPerTrail;                            // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                               bDeadTrailsOnDeactivate : 1;                 // 0x007C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                               bClipSourceSegement : 1;                   // 0x007C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                               bEnablePreviousTangentRecalculation : 1;       // 0x007C
(0x0004) [0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                               bTangentRecalculationEveryFrame : 1;          // 0x007C
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                               bRenderGeometry : 1;                       // 0x007C (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                               bRenderSpawnPoints : 1;                    // 0x007C (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                               bRenderTangents : 1;                       // 0x007C (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                               bRenderTessellation : 1;                   // 0x007C (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
float                                   TilingDistance;                            // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   DistanceTessellationStepSize;               // 0x0084 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                                  TangentTessellationScalar;                  // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataAnimTrail");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataApex
// 0x0010 (0x0070 - 0x0080)
class UParticleModuleTypeDataApex : public UParticleModuleTypeDataBase
{
public:
class UApexGenericAsset*                ApexIOFX;                              // 0x0070 (0x0008)
[0x0000000000000000]
class UApexGenericAsset*                ApexEmitter;                           // 0x0078 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataApex");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataBeam
// 0x00E0 (0x0070 - 0x0150)
class UParticleModuleTypeDataBeam : public UParticleModuleTypeDataBase
{
public:
uint8_t                                BeamMethod;                            // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                EndPointMethod;                        // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat           Distance;                              // 0x0078 (0x0028)
```

```
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                 EndPoint;                                 // 0x00A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
int32_t                                 TessellationFactor;                         // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                  EmitterStrength;                            // 0x00D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                  TargetStrength;                             // 0x00F8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                 EndPointDirection;                          // 0x0120 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
int32_t                                 TextureTile;                              // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                               RenderGeometry : 1;                       // 0x014C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                               RenderDirectLine : 1;                     // 0x014C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                               RenderLines : 1;                          // 0x014C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                               RenderTessellation : 1;                   // 0x014C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataBeam");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataBeam2
// 0x00A8 (0x0070 - 0x0118)
class UParticleModuleTypeDataBeam2 : public UParticleModuleTypeDataBase
{
public:
uint8_t                                 BeamMethod;                               // 0x0070 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                 TaperMethod;                              // 0x0071 (0x0001)
[0x0000000000000001] (CPF_Edit)
int32_t                                 TextureTile;                              // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  TextureTileDistance;                       // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 Sheets;                                   // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 MaxBeamCount;                             // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                                   Speed;                              // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 InterpolationPoints;                // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bAlwaysOn : 1;                      // 0x008C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           RenderGeometry : 1;                 // 0x008C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                           RenderDirectLine : 1;               // 0x008C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           RenderLines : 1;                    // 0x008C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           RenderTessellation : 1;             // 0x008C (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
int32_t                                 UpVectorStepSize;                   // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                            BranchParentName;                   // 0x0094 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                    Distance;                   // 0x00A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    TaperFactor;                // 0x00C8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    TaperScale;                 // 0x00F0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataBeam2");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataMesh
// 0x0024 (0x0070 - 0x0094)
class UParticleModuleTypeDataMesh : public UParticleModuleTypeDataBase
{
public:
class UStaticMesh*                          Mesh;                           // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                            MeshParamName;                      // 0x0078 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                           CastShadows : 1;                    // 0x0080 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                           DoCollisions : 1;                   // 0x0080 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                           bAllowMotionBlur : 1;               // 0x0080 (0x0004)
```

```
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                     bOverrideMaterial : 1;                    // 0x0080 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                     bCameraFacing : 1;                        // 0x0080 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                     bApplyParticleRotationAsSpin : 1;            // 0x0080 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
uint8_t                         MeshAlignment;                            // 0x0084 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                         AxisLockOption;                           // 0x0085 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                         CameraFacingUpAxisOption;                   // 0x0086 (0x0001)
[0x0000000020000000]  CPF_Deprecated)
uint8_t                         CameraFacingOption;                        // 0x0087 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                           Pitch;                                    // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           Roll;                                     // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           Yaw;                                      // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataMesh");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataMeshPhysX
// 0x0028 (0x0094 - 0x00BC)
class UParticleModuleTypeDataMeshPhysX : public UParticleModuleTypeDataMesh
{
public:
class UPhysXParticleSystem*                 PhysXParSys;                          // 0x0098 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                         PhysXRotationMethod;                      // 0x00A0 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                           FluidRotationCoefficient;                  // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPhysXEmitterVerticalLodProperties        VerticalLod;                           // 0x00A8
(0x0010) [0x0000000000000001] (CPF_Edit)
float                           ZOffset;                                  // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataMeshPhysX");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataPhysX
// 0x0018 (0x0070 - 0x0088)
class UParticleModuleTypeDataPhysX : public UParticleModuleTypeDataBase
{
public:
class UPhysXParticleSystem*                    PhysXParSys;                             // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FPhysXEmitterVerticalLodProperties         VerticalLod;                             // 0x0078
(0x0010) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataPhysX");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataRibbon
// 0x0028 (0x0070 - 0x0098)
class UParticleModuleTypeDataRibbon : public UParticleModuleTypeDataBase
{
public:
int32_t                                 MaxTessellationBetweenParticles;            // 0x0070 (0x0004)
[0x0000000000000000]
int32_t                                 SheetsPerTrail;                            // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 MaxTrailCount;                             // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 MaxParticleInTrailCount;                   // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                               bDeadTrailsOnDeactivate : 1;               // 0x0080 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
unsigned long                    bDeadTrailsOnSourceLoss : 1;          // 0x0080 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bClipSourceSegement : 1;              // 0x0080 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    bEnablePreviousTangentRecalculation : 1;      // 0x0080
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bTangentRecalculationEveryFrame : 1;          // 0x0080
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                    bSpawnInitialParticle : 1;            // 0x0080 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                    bRenderGeometry : 1;                  // 0x0080 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                    bRenderSpawnPoints : 1;               // 0x0080 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                    bRenderTangents : 1;                  // 0x0080 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                    bRenderTessellation : 1;              // 0x0080 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                    bEnableTangentDiffInterpScale : 1;            // 0x0080 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
uint8_t                          RenderAxis;                           // 0x0084 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                            TangentSpawningScalar;                // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            TilingDistance;                       // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            DistanceTessellationStepSize;         // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            TangentTessellationScalar;            // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataRibbon");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataTrail
// 0x003C (0x0070 - 0x00AC)
class UParticleModuleTypeDataTrail : public UParticleModuleTypeDataBase
{
public:
unsigned long                    RenderGeometry : 1;                   // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    RenderLines : 1;                      // 0x0070 (0x0004)
```

```
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       RenderTessellation : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                       Tapered : 1;                               // 0x0070 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                       SpawnByDistance : 1;                       // 0x0070 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
int32_t                         TessellationFactor;                            // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat              Tension;                             // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FVector                      SpawnDistance;                             // 0x00A0 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataTrail");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleTypeDataTrail2
// 0x0020 (0x0070 - 0x0090)
class UParticleModuleTypeDataTrail2 : public UParticleModuleTypeDataBase
{
public:
int32_t                         TessellationFactor;                            // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          TessellationFactorDistance;                     // 0x0074 (0x0004)
[0x0000000000000000]
float                          TessellationStrength;                           // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         TextureTile;                                   // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         Sheets;                                        // 0x0080 (0x0004)
[0x0000000000000000]
int32_t                         MaxTrailCount;                                 // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         MaxParticleInTrailCount;                       // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bClipSourceSegement : 1;                   // 0x008C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bClearTangents : 1;                        // 0x008C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       RenderGeometry : 1;                        // 0x008C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
```

```
unsigned long                              RenderDirectLine : 1;                       // 0x008C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                              RenderLines : 1;                            // 0x008C (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                              RenderTessellation : 1;                     // 0x008C (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleTypeDataTrail2");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleUberBase
// 0x0014 (0x006C - 0x0080)
class UParticleModuleUberBase : public UParticleModule
{
public:
TArray<struct FName>                          RequiredModules;                          // 0x0070 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleUberLTISIVCL
// 0x00F0 (0x0080 - 0x0170)
class UParticleModuleUberLTISIVCL : public UParticleModuleUberBase
{
public:
struct FRawDistributionFloat                    LifeTime;                                 // 0x0080 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector                   StartSize;                                // 0x00A8 (0x0028)
```

```
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector                StartVelocity;                          // 0x00D0 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionFloat                 StartVelocityRadial;                    // 0x00F8 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector                ColorOverLife;                          // 0x0120 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionFloat                 AlphaOverLife;                          // 0x0148 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberLTISIVCL");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleUberLTISIVCLIL
// 0x0118 (0x0080 - 0x0198)
class UParticleModuleUberLTISIVCLIL : public UParticleModuleUberBase
{
public:
struct FRawDistributionFloat                 LifeTime;                               // 0x0080 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector                StartSize;                              // 0x00A8 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector                StartVelocity;                          // 0x00D0 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionFloat                 StartVelocityRadial;                    // 0x00F8 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector                ColorOverLife;                          // 0x0120 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionFloat                 AlphaOverLife;                          // 0x0148 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector                StartLocation;                          // 0x0170 (0x0028)
```

```
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberLTISIVCLIL");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleUberLTISIVCLILIRSSBLIRR
// 0x0198 (0x0080 - 0x0218)
class UParticleModuleUberLTISIVCLILIRSSBLIRR : public UParticleModuleUberBase
{
public:
struct FRawDistributionFloat                LifeTime;                          // 0x0080 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector               StartSize;                         // 0x00A8 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector               StartVelocity;                     // 0x00D0 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionFloat                StartVelocityRadial;               // 0x00F8 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector               ColorOverLife;                     // 0x0120 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionFloat                AlphaOverLife;                     // 0x0148 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector               StartLocation;                     // 0x0170 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionFloat                StartRotation;                     // 0x0198 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
struct FRawDistributionVector               SizeLifeMultiplier;                // 0x01C0 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)
unsigned long                           SizeMultiplyX : 1;                     // 0x01E8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           SizeMultiplyY : 1;                     // 0x01E8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
```

```
unsigned long                          SizeMultiplyZ : 1;                     // 0x01E8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FRawDistributionFloat                  StartRotationRate;                      // 0x01F0 (0x0028)
[0x0000000002480009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_NoClear)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberLTISIVCLILIRSSBLIRR");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleUberRainDrops
// 0x0088 (0x0080 - 0x0108)
class UParticleModuleUberRainDrops : public UParticleModuleUberBase
{
public:
float                                 LifetimeMin;                           // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 LifetimeMax;                           // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                         StartSizeMin;                          // 0x0088 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         StartSizeMax;                          // 0x0094 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         StartVelocityMin;                       // 0x00A0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         StartVelocityMax;                       // 0x00AC (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                 StartVelocityRadialMin;                 // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 StartVelocityRadialMax;                 // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                         ColorOverLife;                          // 0x00C0 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                 AlphaOverLife;                         // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bIsUsingCylinder : 1;                   // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bPositive_X : 1;                        // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bPositive_Y : 1;                        // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bPositive_Z : 1;                        // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
```

```
unsigned long                          bNegative_X : 1;                         // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bNegative_Y : 1;                         // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bNegative_Z : 1;                         // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          bSurfaceOnly : 1;                        // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                          bVelocity : 1;                         // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                          bRadialVelocity : 1;                     // 0x00D0 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
float                             PC_VelocityScale;                        // 0x00D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                         PC_StartLocation;                        // 0x00D8 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                             PC_StartRadius;                          // 0x00E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                             PC_StartHeight;                          // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                            PC_HeightAxis;                           // 0x00EC (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                         StartLocationMin;                        // 0x00F0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         StartLocationMax;                        // 0x00FC (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberRainDrops");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleUberRainImpacts
// 0x01A0 (0x0080 - 0x0220)
class UParticleModuleUberRainImpacts : public UParticleModuleUberBase
{
public:
struct FRawDistributionFloat                 LifeTime;                          // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                 StartSize;                         // 0x00A8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                 StartRotation;                       // 0x00D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                          bInheritParent : 1;                      // 0x00F8 (0x0004)
```

```
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          MultiplyX : 1;                           // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          MultiplyY : 1;                           // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          MultiplyZ : 1;                           // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bIsUsingCylinder : 1;                     // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bPositive_X : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bPositive_Y : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          bPositive_Z : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                          bNegative_X : 1;                          // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                          bNegative_Y : 1;                          // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                          bNegative_Z : 1;                          // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                          bSurfaceOnly : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                          bVelocity : 1;                           // 0x00F8 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                          bRadialVelocity : 1;                      // 0x00F8 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
struct FRawDistributionVector                 LifeMultiplier;                          // 0x0100 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 PC_VelocityScale;                        // 0x0128 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                 PC_StartLocation;                        // 0x0150 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 PC_StartRadius;                          // 0x0178 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 PC_StartHeight;                          // 0x01A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
uint8_t                                PC_HeightAxis;                           // 0x01C8 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionVector                 ColorOverLife;                           // 0x01D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                 AlphaOverLife;                           // 0x01F8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberRainImpacts");
}
return uClassPointer;
};

};

// Class Engine.ParticleModuleUberRainSplashA
// 0x00F8 (0x0080 - 0x0178)
class UParticleModuleUberRainSplashA : public UParticleModuleUberBase
{
public:
struct FRawDistributionFloat                    LifeTime;                              // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                   StartSize;                             // 0x00A8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                   StartRotation;                         // 0x00D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                   bInheritParent : 1;                    // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                   MultiplyX : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                   MultiplyY : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                   MultiplyZ : 1;                         // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FRawDistributionVector                   LifeMultiplier;                        // 0x0100 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                   ColorOverLife;                         // 0x0128 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    AlphaOverLife;                         // 0x0150 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberRainSplashA");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleUberRainSplashB
// 0x00F8 (0x0080 - 0x0178)
class UParticleModuleUberRainSplashB : public UParticleModuleUberBase
{
public:
struct FRawDistributionFloat                    LifeTime;                              // 0x0080 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                   StartSize;                             // 0x00A8 (0x0028)
```

```
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                    ColorOverLife;                           // 0x00D0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                     AlphaOverLife;                           // 0x00F8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionVector                    LifeMultiplier;                          // 0x0120 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                    MultiplyX : 1;                           // 0x0148 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                    MultiplyY : 1;                           // 0x0148 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                    MultiplyZ : 1;                           // 0x0148 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FRawDistributionFloat                     StartRotationRate;                       // 0x0150 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleUberRainSplashB");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleVelocityBase
// 0x0008 (0x006C - 0x0074)
class UParticleModuleVelocityBase : public UParticleModule
{
public:
unsigned long                                    bInWorldSpace : 1;                       // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                    bApplyOwnerScale : 1;                    // 0x0070 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleVelocityBase");
}

return uClassPointer;
};
```

```
};

// Class Engine.ParticleModuleVelocity
// 0x0054 (0x0074 - 0x00C8)
class UParticleModuleVelocity : public UParticleModuleVelocityBase
{
public:
struct FRawDistributionVector                    StartVelocity;                            // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    StartVelocityRadial;                       // 0x00A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleVelocity");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleVelocity_Seeded
// 0x0020 (0x00C8 - 0x00E8)
class UParticleModuleVelocity_Seeded : public UParticleModuleVelocity
{
public:
struct FParticleRandomSeedInfo                  RandomSeedInfo;                            // 0x00C8
(0x0020) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleVelocity_Seeded");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleVelocityCone
// 0x0060 (0x0074 - 0x00D4)
class UParticleModuleVelocityCone : public UParticleModuleVelocityBase
{
```

```
public:
struct FRawDistributionFloat                    Angle;                                    // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    Velocity;                                 // 0x00A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FVector                         Direction;                              // 0x00C8 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleVelocityCone");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleVelocityInheritParent
// 0x0030 (0x0074 - 0x00A4)
class UParticleModuleVelocityInheritParent : public UParticleModuleVelocityBase
{
public:
struct FRawDistributionVector                   Scale;                                    // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
float                                  MaxAddedVelocity;                        // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleVelocityInheritParent");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleVelocityOverLifetime
// 0x0030 (0x0074 - 0x00A4)
class UParticleModuleVelocityOverLifetime : public UParticleModuleVelocityBase
{
public:
struct FRawDistributionVector                   VelOverLife;                                  // 0x0078 (0x0028)
```

```
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                        Absolute : 1;                        // 0x00A0 (0x0004)
[0x0000000000000009] [0x00000001] (CPF_Edit | CPF_ExportObject)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleVelocityOverLifetime");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleWorldForcesBase
// 0x0004 (0x006C - 0x0070)
class UParticleModuleWorldForcesBase : public UParticleModule
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleWorldForcesBase");
}

return uClassPointer;
};

};

// Class Engine.ParticleModulePhysicsVolumes
// 0x0029 (0x0070 - 0x0099)
class UParticleModulePhysicsVolumes : public UParticleModuleWorldForcesBase
{
public:
struct FRawDistributionFloat                GlobalInfluence;                        // 0x0070 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
uint8_t                                     LevelInfluenceType;                     // 0x0098 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModulePhysicsVolumes");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleWorldAttractor
// 0x0030 (0x0070 - 0x00A0)
class UParticleModuleWorldAttractor : public UParticleModuleWorldForcesBase
{
public:
unsigned long                          bParticleLifeRelative : 1;              // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FRawDistributionFloat                 AttractorInfluence;                  // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleWorldAttractor");
}

return uClassPointer;
};

};

// Class Engine.ParticleModuleEventSendToGame
// 0x0000 (0x0060 - 0x0060)
class UParticleModuleEventSendToGame : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleModuleEventSendToGame");
}

return uClassPointer;
};
```

```
void DoEvent(struct FVector& InCollideDirection, struct FVector& InHitLocation, struct FVector&
InHitNormal, struct FName& InBoneName);
};

// Class Engine.ParticleSystemReplay
// 0x0018 (0x0060 - 0x0078)
class UParticleSystemReplay : public UObject
{
public:
int32_t                          ClipIDNumber;                    // 0x0060 (0x0004)
[0x0000000000001001] (CPF_Edit | CPF_Native)
TArray<struct FParticleSystemReplayFrame>        Frames;                        // 0x0068
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleSystemReplay");
}

return uClassPointer;
};

};

// Class Engine.PhysXParticleSystem
// 0x0070 (0x0060 - 0x00D0)
class UPhysXParticleSystem : public UObject
{
public:
int32_t                          MaxParticles;                    // 0x0060 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          ParticleSpawnReserve;                // 0x0064 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                          RBChannel;                      // 0x0068 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                          SimulationMethod;                  // 0x0069 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          PacketSizeMultiplier;                // 0x006A (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRBCollisionChannelContainer          RBCollideWithChannels;              // 0x006C
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
float                           CollisionDistance;                 // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           RestitutionWithStaticShapes;            // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           RestitutionWithDynamicShapes;            // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           FrictionWithStaticShapes;              // 0x007C (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                              FrictionWithDynamicShapes;                  // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              StaticFrictionWithStaticShapes;             // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              StaticFrictionWithDynamicShapes;            // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                      bDynamicCollision : 1;                      // 0x008C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                      bDisableGravity : 1;                        // 0x008C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                      bStaticCollision : 1;                       // 0x008C (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                      bTwoWayCollision : 1;                       // 0x008C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                      bDestroy : 1;                               // 0x008C (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                      bSyncFailed : 1;                            // 0x008C (0x0004)
[0x0000000000002000] [0x00000020] (CPF_Transient)
unsigned long                      bIsInGame : 1;                              // 0x008C (0x0004)
[0x0000000000002000] [0x00000040] (CPF_Transient)
float                              MaxMotionDistance;                          // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              Damping;                                    // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                     ExternalAcceleration;                       // 0x0098 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                              RestParticleDistance;                       // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              RestDensity;                                // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              KernelRadiusMultiplier;                     // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              Stiffness;                                  // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              Viscosity;                                  // 0x00B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              CollisionResponseCoefficient;               // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                    CascadeScene;                               // 0x00C0 (0x0008)
[0x0000000000001000] (CPF_Native)
struct FPointer                    PSys;                                       // 0x00C8 (0x0008)
[0x0000000000001000] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PhysXParticleSystem");
}
return uClassPointer;
};

};

// Class Engine.KActor
// 0x00F0 (0x02C8 - 0x03B8)
class AKActor : public ADynamicSMActor
{
public:
unsigned long                          bDamageAppliesImpulse : 1;                   // 0x02C8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bWakeOnLevelStart : 1;                    // 0x02C8 (0x0004)
[0x0000000100000021] [0x00000002] (CPF_Edit | CPF_Net)
unsigned long                          bCurrentSlide : 1;                       // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                          bSlideActive : 1;                        // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                          bEnableStayUprightSpring : 1;                // 0x02C8 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bLimitMaxPhysicsVelocity : 1;               // 0x02C8 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bNeedsRBStateReplication : 1;               // 0x02C8 (0x0004)
[0x0000000000002000] [0x00000040] (CPF_Transient)
unsigned long                          bDisableClientSidePawnInteractions : 1;       // 0x02C8
(0x0004) [0x0000000000000000] [0x00000080]
class UParticleSystemComponent*                ImpactEffectComponent;                    // 0x02D0
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UAudioComponent*                      ImpactSoundComponent;                    // 0x02D8
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UAudioComponent*                      ImpactSoundComponent2;                   // 0x02E0
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                               LastImpactTime;                         // 0x02E8 (0x0004)
[0x0000000000000000]
struct FPhysEffectInfo                       ImpactEffectInfo;                        // 0x02F0 (0x0018)
[0x0000000000000000]
class UParticleSystemComponent*                SlideEffectComponent;                    // 0x0308
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UAudioComponent*                      SlideSoundComponent;                    // 0x0310
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                               LastSlideTime;                          // 0x0318 (0x0004)
[0x0000000000000000]
struct FPhysEffectInfo                       SlideEffectInfo;                         // 0x0320 (0x0018)
[0x0000000000000000]
float                               StayUprightTorqueFactor;                  // 0x0338 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               StayUprightMaxTorque;                    // 0x033C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               MaxPhysicsVelocity;                     // 0x0340 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              UnknownData00[0xC];                     // 0x0344 (0x000C) MISSED
OFFSET
struct FRigidBodyState                        RBState;                             // 0x0350 (0x0040)
[0x0000000000001022] (CPF_Const | CPF_Net | CPF_Native)
```

```
float                                   AngErrorAccumulator;                          // 0x0390 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FVector                          ReplicatedDrawScale3D;                        // 0x0394 (0x000C)
[0x0000000100000020] (CPF_Net)
struct FVector                          InitialLocation;                              // 0x03A0 (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FRotator                         InitialRotation;                              // 0x03AC (0x000C)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.KActor");
}

return uClassPointer;
};

void Reset();
void OnTeleport(class USeqAct_Teleport* inAction);
void OnToggle(class USeqAct_Toggle* Action);
void eventApplyImpulse(struct FVector ImpulseDir, float ImpulseMag, struct FVector HitLocation,
struct FTraceHitInfo HitInfo);
void eventReplicatedEvent(struct FName VarName);
void eventSpawnedByKismet();
void SetPhysicalCollisionProperties();
void eventDestroyed();
void eventFellOutOfWorld();
void eventPostBeginPlay();
void ResolveRBState();
class UPhysicalMaterial* GetKActorPhysMaterial();
};

// Class Engine.KActorFromStatic
// 0x000C (0x03B8 - 0x03C4)
class AKActorFromStatic : public AKActor
{
public:
class AActor*                           MyStaticMeshActor;                            // 0x03B8 (0x0008)
[0x0000000000000000]
float                                   MaxImpulseSpeed;                              // 0x03C0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.KActorFromStatic");
}

return uClassPointer;
};

void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
void eventBump(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitNormal);
void ReceiveImpulse(class APawn* Other, struct FVector HitLocation, struct FVector HitNormal);
void eventApplyImpulse(struct FVector ImpulseDir, float ImpulseMag, struct FVector HitLocation,
struct FTraceHitInfo HitInfo);
static class AKActorFromStatic* MakeDynamic(class UStaticMeshComponent* MovableMesh);
static void MakeStatic();
void BecomeStatic();
void eventOnWakeRBPhysics();
void eventOnSleepRBPhysics();
void DisablePrecomputedLighting();
};

// Class Engine.KActorSpawnable
// 0x0004 (0x03B8 - 0x03BC)
class AKActorSpawnable : public AKActor
{
public:
unsigned long                   bRecycleScaleToZero : 1;            // 0x03B8 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                   bScalingToZero : 1;                 // 0x03B8 (0x0004)
[0x0000000000000000] [0x00000002]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.KActorSpawnable");
}

return uClassPointer;
};

void ResetComponents();
void eventRecycleInternal();
void Recycle();
void Initialize();
};

// Class Engine.KAsset
// 0x0020 (0x0268 - 0x0288)
class AKAsset : public AActor
{
```

```
public:
class USkeletalMeshComponent*                SkeletalMeshComponent;                    // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                        bDamageAppliesImpulse : 1;                // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bWakeOnLevelStart : 1;                 // 0x0270 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                        bBlockPawns : 1;                    // 0x0270 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
class USkeletalMesh*                      ReplicatedMesh;                        // 0x0278 (0x0008)
[0x0000000100002020] (CPF_Net | CPF_Transient)
class UPhysicsAsset*                      ReplicatedPhysAsset;                  // 0x0280 (0x0008)
[0x0000000100002020] (CPF_Net | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.KAsset");
}

return uClassPointer;
};

void DoKismetAttachment(class AActor* Attachment, class USeqAct_AttachToActor* Action);
void OnTeleport(class USeqAct_Teleport* inAction);
void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
void SetMeshAndPhysAsset(class USkeletalMesh* NewMesh, class UPhysicsAsset*
NewPhysAsset);
void eventPostBeginPlay();
};

// Class Engine.RB_ConstraintActor
// 0x0038 (0x0268 - 0x02A0)
class ARB_ConstraintActor : public ARigidBodyBase
{
public:
class AActor*                           ConstraintActor1;                     // 0x0268 (0x0008)
[0x0000000000000001] (CPF_Edit)
class AActor*                           ConstraintActor2;                     // 0x0270 (0x0008)
[0x0000000000000001] (CPF_Edit)
class URB_ConstraintSetup*                  ConstraintSetup;                      // 0x0278 (0x0008)
[0x0000000006400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_NoClear |
CPF_EditInline)
class URB_ConstraintInstance*                ConstraintInstance;                  // 0x0280
(0x0008) [0x0000000006400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink |
CPF_NoClear | CPF_EditInline)
unsigned long                        bDisableCollision : 1;                // 0x0288 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
```

```
unsigned long                          bUpdateActor1RefFrame : 1;                // 0x0288 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bUpdateActor2RefFrame : 1;                // 0x0288 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
class AActor*                          PulleyPivotActor1;                     // 0x0290 (0x0008)
[0x0000000000000001] (CPF_Edit)
class AActor*                          PulleyPivotActor2;                     // 0x0298 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_ConstraintActor");
}

return uClassPointer;
};

void OnToggleConstraintDrive(class USeqAct_ToggleConstraintDrive* Action);
void OnToggle(class USeqAct_Toggle* Action);
void OnDestroy(class USeqAct_Destroy* Action);
void TermConstraint();
void InitConstraint(class AActor* Actor1, class AActor* Actor2, struct FName Actor1Bone, struct
FName Actor2Bone, float BreakThreshold);
void SetDisableCollision(unsigned long NewDisableCollision);
};

// Class Engine.RB_LineImpulseActor
// 0x0019 (0x0268 - 0x0281)
class ARB_LineImpulseActor : public ARigidBodyBase
{
public:
float                                  ImpulseStrength;                       // 0x0268 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                  ImpulseRange;                          // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bVelChange : 1;                        // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bStopAtFirstHit : 1;                   // 0x0270 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bCauseFracture : 1;                    // 0x0270 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
class UArrowComponent*                 Arrow;                                 // 0x0278 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
uint8_t                                ImpulseCount;                          // 0x0280 (0x0001)
[0x0000000100000020] (CPF_Net)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_LineImpulseActor");
}

return uClassPointer;
};

void eventReplicatedEvent(struct FName VarName);
void OnToggle(class USeqAct_Toggle* inAction);
void FireLineImpulse();
};

// Class Engine.RB_RadialImpulseActor
// 0x0011 (0x0268 - 0x0279)
class ARB_RadialImpulseActor : public ARigidBodyBase
{
public:
class UDrawSphereComponent*          RenderComponent;               // 0x0268
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class URB_RadialImpulseComponent*          ImpulseComponent;               // 0x0270
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
uint8_t                    ImpulseCount;                    // 0x0278 (0x0001)
[0x0000000100000020] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_RadialImpulseActor");
}

return uClassPointer;
};

void eventReplicatedEvent(struct FName VarName);
void OnToggle(class USeqAct_Toggle* inAction);
};

// Class Engine.RB_Thruster
// 0x0008 (0x0268 - 0x0270)
class ARB_Thruster : public ARigidBodyBase
{
public:
unsigned long                    bThrustEnabled : 1;               // 0x0268 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                       ThrustStrength;                   // 0x026C (0x0004)
[0x0000000200000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_Thruster");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
};

// Class Engine.WorldAttractor
// 0x0108 (0x0268 - 0x0370)
class AWorldAttractor : public AActor
{
public:
unsigned long                          bEnabled : 1;                          // 0x0268 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                  LoopDuration;                          // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  CurrentTime;                           // 0x0270 (0x0004)
[0x0000000000000000]
uint8_t                                FalloffType;                           // 0x0274 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FMatineeRawDistributionFloat            FalloffExponent;                       // 0x0278
(0x0030) [0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FMatineeRawDistributionFloat            Range;                                 // 0x02A8 (0x0030)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FMatineeRawDistributionFloat            Strength;                              // 0x02D8 (0x0030)
[0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
float                                  CollisionRadius;                       // 0x0308 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FMatineeRawDistributionFloat            DragCoefficient;                       // 0x0310
(0x0030) [0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FMatineeRawDistributionFloat            DragRadius;                            // 0x0340
(0x0030) [0x0000000200480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WorldAttractor");
}

return uClassPointer;
```

```
};

void OnSetWorldAttractorParam(class USeqAct_SetWorldAttractorParam* Action);
};

// Class Engine.RB_ConstraintDrawComponent
// 0x0008 (0x0258 - 0x0260)
class URB_ConstraintDrawComponent : public UPrimitiveComponent
{
public:
class UMaterialInterface*                    LimitMaterial;                         // 0x0258 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_ConstraintDrawComponent");
}

return uClassPointer;
};

};

// Class Engine.RB_RadialImpulseComponent
// 0x0018 (0x0258 - 0x0270)
class URB_RadialImpulseComponent : public UPrimitiveComponent
{
public:
uint8_t                                  ImpulseFalloff;                        // 0x0258 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                    ImpulseStrength;                       // 0x025C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    ImpulseRadius;                         // 0x0260 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bVelChange : 1;                        // 0x0264 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bCauseFracture : 1;                    // 0x0264 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
class UDrawSphereComponent*                  PreviewSphere;                         // 0x0268
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_RadialImpulseComponent");
}

return uClassPointer;
};

void FireImpulse(struct FVector Origin);
};

// Class Engine.RB_Handle
// 0x0077 (0x009D - 0x0114)
class URB_Handle : public UActorComponent
{
public:
class UPrimitiveComponent*                    GrabbedComponent;                        // 0x00A0
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FName                                  GrabbedBoneName;                         // 0x00A8 (0x0008)
[0x0000000000000000]
int32_t                                       SceneIndex;                              // 0x00B0 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
unsigned long                                 bInHardware : 1;                         // 0x00B4 (0x0004)
[0x0000000000003002] [0x00000001] (CPF_Const | CPF_Native | CPF_Transient)
unsigned long                                 bRotationConstrained : 1;                // 0x00B4 (0x0004)
[0x0000000000003002] [0x00000002] (CPF_Const | CPF_Native | CPF_Transient)
unsigned long                                 bInterpolating : 1;                      // 0x00B4 (0x0004)
[0x0000000000000000] [0x00000004]
struct FPointer                               HandleData;                              // 0x00B8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                               KinActorData;                            // 0x00C0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
float                                         LinearDamping;                           // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                         LinearStiffness;                         // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                                LinearStiffnessScale3D;                  // 0x00D0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                                LinearDampingScale3D;                    // 0x00DC (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                         AngularDamping;                          // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                         AngularStiffness;                        // 0x00EC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                                Destination;                             // 0x00F0 (0x000C)
[0x0000000000000000]
struct FVector                                StepSize;                                // 0x00FC (0x000C)
[0x0000000000000000]
struct FVector                                Location;                                // 0x0108 (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.RB_Handle");
}

return uClassPointer;
};

struct FQuat GetOrientation();
void SetOrientation(struct FQuat& NewOrientation);
void UpdateSmoothLocation(struct FVector& NewLocation);
void SetSmoothLocation(struct FVector NewLocation, float MoveTime);
void SetLocation(struct FVector NewLocation);
void ReleaseComponent();
void GrabComponent(class UPrimitiveComponent* Component, struct FName InBoneName,
struct FVector GrabLocation, unsigned long bConstrainRotation);
};

// Class Engine.RB_Spring
// 0x006B (0x009D - 0x0108)
class URB_Spring : public UActorComponent
{
public:
class UPrimitiveComponent*                  Component1;                            // 0x00A0 (0x0008)
[0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FName                             BoneName1;                            // 0x00A8 (0x0008)
[0x0000000000000002] (CPF_Const)
class UPrimitiveComponent*                  Component2;                            // 0x00B0 (0x0008)
[0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FName                             BoneName2;                            // 0x00B8 (0x0008)
[0x0000000000000002] (CPF_Const)
int32_t                                SceneIndex;                          // 0x00C0 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
unsigned long                            bInHardware : 1;                     // 0x00C4 (0x0004)
[0x0000000000001002] [0x00000001] (CPF_Const | CPF_Native)
unsigned long                            bEnableForceMassRatio : 1;                // 0x00C4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FPointer                          SpringData;                          // 0x00C8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                                  TimeSinceActivation;                   // 0x00D0 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                                  MinBodyMass;                         // 0x00D4 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                  SpringSaturateDist;                    // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  SpringMaxForce;                        // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  MaxForceMassRatio;                       // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FInterpCurveFloat                    SpringMaxForceTimeScale;                  // 0x00E8
(0x0018) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                  DampSaturateVel;                       // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  DampMaxForce;                         // 0x0104 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_Spring");
}

return uClassPointer;
};

void Clear();
void SetComponents(class UPrimitiveComponent* InComponent1, struct FName InBoneName1,
struct FVector Position1, class UPrimitiveComponent* InComponent2, struct FName
InBoneName2, struct FVector Position2);
};

// Class Engine.ActorFactoryApexClothing
// 0x0018 (0x00B8 - 0x00D0)
class UActorFactoryApexClothing : public UActorFactorySkeletalMesh
{
public:
TArray<class UApexClothingAsset*>          ClothingAssets;                    // 0x00B8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
uint8_t                                    ClothingRBChannel;                 // 0x00C8 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FRBCollisionChannelContainer        ClothingRBCollideWithChannels;           //
0x00CC (0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryApexClothing");
}

return uClassPointer;
};

};

// Class Engine.ApexDestructibleDamageParameters
// 0x0010 (0x0060 - 0x0070)
class UApexDestructibleDamageParameters : public UObject
{
public:
TArray<struct FDamagePair>                 DamageMap;                         // 0x0060 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexDestructibleDamageParameters");
}

return uClassPointer;
};

};

// Class Engine.FractureMaterial
// 0x0010 (0x0060 - 0x0070)
class UFractureMaterial : public UObject
{
public:
class UParticleSystem*                    FractureEffect;                    // 0x0060 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundCue*                          FractureSound;                     // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FractureMaterial");
}

return uClassPointer;
};

};

// Class Engine.PhysicalMaterial
// 0x0088 (0x0060 - 0x00E8)
class UPhysicalMaterial : public UObject
{
public:
int32_t                                   MaterialIndex;                     // 0x0060 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                     Friction;                          // 0x0064 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                     Restitution;                       // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                             bForceConeFriction : 1;            // 0x006C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
```

```
unsigned long                         bEnableAnisotropicFriction : 1;            // 0x006C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FVector                        AnisoFrictionDir;                          // 0x0070 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                            FrictionV;                                 // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            Density;                                   // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AngularDamping;                            // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            LinearDamping;                             // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            MagneticResponse;                          // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            WindResponse;                              // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            InertiaScale;                              // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            ImpactThreshold;                           // 0x0098 (0x0004)
[0x0000000000000000]
float                            ImpactReFireDelay;                         // 0x009C (0x0004)
[0x0000000000000000]
class UParticleSystem*                ImpactEffect;                              // 0x00A0 (0x0008)
[0x0000000000000000]
class USoundCue*                      ImpactSound;                               // 0x00A8 (0x0008)
[0x0000000000000000]
float                            SlideThreshold;                            // 0x00B0 (0x0004)
[0x0000000000000000]
float                            SlideReFireDelay;                          // 0x00B4 (0x0004)
[0x0000000000000000]
class UParticleSystem*                SlideEffect;                               // 0x00B8 (0x0008)
[0x0000000000000000]
class USoundCue*                      SlideSound;                                // 0x00C0 (0x0008)
[0x0000000000000000]
class USoundCue*                      FractureSoundExplosion;                    // 0x00C8 (0x0008)
[0x0000000000000000]
class USoundCue*                      FractureSoundSingle;                       // 0x00D0 (0x0008)
[0x0000000000000000]
class UPhysicalMaterial*              Parent;                                    // 0x00D8 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UPhysicalMaterialPropertyBase*        PhysicalMaterialProperty;                  // 0x00E0
(0x0008) [0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PhysicalMaterial");
}
```

```
return uClassPointer;
};

class UPhysicalMaterialPropertyBase* GetPhysicalMaterialProperty(class UClass*
DesiredClass);
void FindFractureSounds(class USoundCue*& OutSoundExplosion, class USoundCue*&
OutSoundSingle);
struct FPhysEffectInfo FindPhysEffectInfo(uint8_t Type);
};

// Class Engine.PhysicalMaterialPropertyBase
// 0x0000 (0x0060 - 0x0060)
class UPhysicalMaterialPropertyBase : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PhysicalMaterialPropertyBase");
}

return uClassPointer;
};

};

// Class Engine.PhysicsAsset
// 0x0090 (0x0060 - 0x00F0)
class UPhysicsAsset : public UObject
{
public:
class USkeletalMesh*                    DefaultSkelMesh;                    // 0x0060 (0x0008)
[0x0000000800000002] (CPF_Const)
TArray<class URB_BodySetup*>            BodySetup;                          // 0x0068 (0x0010)
[0x000000000440000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
struct FMap_Mirror                      BodySetupIndexMap;                  // 0x0078 (0x0050)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<int32_t>                         BoundsBodies;                       // 0x00C8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class URB_ConstraintSetup*>      ConstraintSetup;                    // 0x00D8
(0x0010) [0x000000000440000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink |
CPF_EditInline)
class UPhysicsAssetInstance*            DefaultInstance;                    // 0x00E8 (0x0008)
[0x000000000440000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PhysicsAsset");
}

return uClassPointer;
};

int32_t FindBodyIndex(struct FName BodyName);
};

// Class Engine.PhysicsAssetInstance
// 0x009C (0x0060 - 0x00FC)
class UPhysicsAssetInstance : public UObject
{
public:
class AActor*                              Owner;                                 // 0x0060 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                                 RootBodyIndex;                          // 0x0068 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<class URB_BodyInstance*>                 Bodies;                                 // 0x0070 (0x0010)
[0x000000000440000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
TArray<class URB_ConstraintInstance*>            Constraints;                             // 0x0080
(0x0010) [0x000000000440000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink |
CPF_EditInline)
struct FMap_Mirror                           CollisionDisableTable;                    // 0x0090 (0x0050)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                                   LinearSpringScale;                      // 0x00E0 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                   LinearDampingScale;                      // 0x00E4 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                   LinearForceLimitScale;                   // 0x00E8 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                   AngularSpringScale;                      // 0x00EC (0x0004)
[0x0000000000000002] (CPF_Const)
float                                   AngularDampingScale;                      // 0x00F0 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                   AngularForceLimitScale;                   // 0x00F4 (0x0004)
[0x0000000000000002] (CPF_Const)
unsigned long                              bInitBodies : 1;                       // 0x00F8 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PhysicsAssetInstance");
}

return uClassPointer;
```

```
};

class URB_ConstraintInstance* FindConstraintInstance(struct FName ConName, class
UPhysicsAsset* InAsset);
class URB_BodyInstance* FindBodyInstance(struct FName BodyName, class UPhysicsAsset*
InAsset);
void SetFullAnimWeightBonesFixed(unsigned long bNewFixed, class USkeletalMeshComponent*
SkelMesh);
void SetFullAnimWeightBlockRigidBody(unsigned long bNewBlockRigidBody, class
USkeletalMeshComponent* SkelMesh);
void SetNamedBodiesBlockRigidBody(unsigned long bNewBlockRigidBody, TArray<struct
FName> BoneNames, class USkeletalMeshComponent* SkelMesh);
void SetNamedRBBoneSprings(unsigned long bEnable, TArray<struct FName> BoneNames, float
InBoneLinearSpring, float InBoneAngularSpring, class USkeletalMeshComponent*
SkelMeshComp);
void SetNamedMotorsAngularVelocityDrive(unsigned long bEnableSwingDrive, unsigned long
bEnableTwistDrive, TArray<struct FName> BoneNames, class USkeletalMeshComponent*
SkelMeshComp, unsigned long bSetOtherBodiesToComplement);
void SetNamedMotorsAngularPositionDrive(unsigned long bEnableSwingDrive, unsigned long
bEnableTwistDrive, TArray<struct FName> BoneNames, class USkeletalMeshComponent*
SkelMeshComp, unsigned long bSetOtherBodiesToComplement);
void SetAllMotorsAngularDriveParams(float InSpring, float InDamping, float InForceLimit, class
USkeletalMeshComponent* SkelMesh, unsigned long bSkipFullAnimWeightBodies);
void SetAllMotorsAngularVelocityDrive(unsigned long bEnableSwingDrive, unsigned long
bEnableTwistDrive, class USkeletalMeshComponent* SkelMeshComp, unsigned long
bSkipFullAnimWeightBodies);
void SetAllMotorsAngularPositionDrive(unsigned long bEnableSwingDrive, unsigned long
bEnableTwistDrive, class USkeletalMeshComponent* SkelMesh, unsigned long
bSkipFullAnimWeightBodies);
void ForceAllBodiesBelowUnfixed(class UPhysicsAsset* InAsset, class
USkeletalMeshComponent* InSkelMesh, unsigned long InbInstanceAlwaysFullAnimWeight,
struct FName& InBoneName);
void SetNamedBodiesFixed(unsigned long bNewFixed, TArray<struct FName> BoneNames, class
USkeletalMeshComponent* SkelMesh, unsigned long bSetOtherBodiesToComplement, unsigned
long bSkipFullAnimWeightBodies);
void SetAllBodiesFixed(unsigned long bNewFixed);
float GetTotalMassBelowBone(struct FName InBoneName, class UPhysicsAsset* InAsset, class
USkeletalMesh* InSkelMesh);
void SetAngularDriveScale(float InAngularSpringScale, float InAngularDampingScale, float
InAngularForceLimitScale);
void SetLinearDriveScale(float InLinearSpringScale, float InLinearDampingScale, float
InLinearForceLimitScale);
};

// Class Engine.PhysicsLODVerticalEmitter
// 0x0004 (0x0060 - 0x0064)
class UPhysicsLODVerticalEmitter : public UObject
{
public:
int32_t                         ParticlePercentage;                 // 0x0060 (0x0004)
[0x0000000000005000] (CPF_Native | CPF_Config)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PhysicsLODVerticalEmitter");
}

return uClassPointer;
};

};

// Class Engine.RB_BodyInstance
// 0x0084 (0x0060 - 0x00E4)
class URB_BodyInstance : public UObject
{
public:
class UPrimitiveComponent*                    OwnerComponent;                          // 0x0060
(0x0008) [0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)
int32_t                                BodyIndex;                           // 0x0068 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FVector                         Velocity;                            // 0x006C (0x000C)
[0x0000000000000000]
struct FVector                         PreviousVelocity;                    // 0x0078 (0x000C)
[0x0000000000000000]
int32_t                                SceneIndex;                          // 0x0084 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                        BodyData;                            // 0x0088 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FBulletBodyData>                 BulletBodys;                         // 0x0090 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                        BoneSpring;                          // 0x00A0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                        BoneSpringKinActor;                  // 0x00A8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
unsigned long                          bEnableBoneSpringLinear : 1;             // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bEnableBoneSpringAngular : 1;            // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bDisableOnOverextension : 1;             // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                          bNotifyOwnerOnOverextension : 1;            // 0x00B0
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bTeleportOnOverextension : 1;            // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bUseKinActorForBoneSpring : 1;           // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bMakeSpringToBaseCollisionComponent : 1;      // 0x00B0
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          bOnlyCollideWithPawns : 1;               // 0x00B0 (0x0004)
[0x0000000000000003] [0x00000080] (CPF_Edit | CPF_Const)
unsigned long                          bEnableCollisionResponse : 1;            // 0x00B0 (0x0004)
```

```
[0x0000000000000003] [0x00000100] (CPF_Edit | CPF_Const)
unsigned long                          bPushBody : 1;                           // 0x00B0 (0x0004)
[0x0000000000000003] [0x00000200] (CPF_Edit | CPF_Const)
unsigned long                          bForceUnfixed : 1;                       // 0x00B0 (0x0004)
[0x0000000000002000] [0x00000400] (CPF_Transient)
unsigned long                          bInstanceAlwaysFullAnimWeight : 1;            // 0x00B0
(0x0004) [0x0000000000002000] [0x00000800] (CPF_Transient)
float                          BoneLinearSpring;                            // 0x00B4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          BoneLinearDamping;                           // 0x00B8 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          BoneAngularSpring;                           // 0x00BC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          BoneAngularDamping;                          // 0x00C0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          OverextensionThreshold;                      // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          CustomGravityFactor;                         // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          LastEffectPlayedTime;                        // 0x00CC (0x0004)
[0x0000000000002000] (CPF_Transient)
class UPhysicalMaterial*                    PhysMaterialOverride;                    // 0x00D0 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ContactReportForceThreshold;                 // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          InstanceMassScale;                           // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          InstanceDampingScale;                        // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_BodyInstance");
}

return uClassPointer;
};

void UpdateDampingProperties();
void UpdateMassProperties(class URB_BodySetup* Setup);
void SetContactReportForceThreshold(float Threshold);
void EnableCollisionResponse(unsigned long bEnableResponse);
void SetPhysMaterialOverride(class UPhysicalMaterial* NewPhysMaterial);
void SetBlockRigidBody(unsigned long bNewBlockRigidBody);
void SetBoneSpringTarget(unsigned long bTeleport, struct FMatrix& InBoneTarget);
void SetBoneSpringParams(float InLinearSpring, float InLinearDamping, float InAngularSpring,
float InAngularDamping);
void EnableBoneSpring(unsigned long bInEnableLinear, unsigned long bInEnableAngular, struct
FMatrix& InBoneTarget);
```

```
struct FVector GetUnrealWorldVelocityAtPoint(struct FVector Point);
struct FVector GetUnrealWorldAngularVelocity();
struct FVector GetUnrealWorldVelocity();
struct FMatrix GetUnrealWorldTM();
class UPhysicsAssetInstance* GetPhysicsAssetInstance();
bool IsValidBodyInstance();
bool IsFixed();
void SetFixed(unsigned long bNewFixed);
float GetBodyMass();
};

// Class Engine.RB_ConstraintInstance
// 0x0080 (0x0060 - 0x00E0)
class URB_ConstraintInstance : public UObject
{
public:
class AActor*                    Owner;                          // 0x0060 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class UPrimitiveComponent*              OwnerComponent;                      // 0x0068
(0x0008) [0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)
int32_t                      ConstraintIndex;                     // 0x0070 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                      SceneIndex;                       // 0x0074 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
unsigned long                    bInHardware : 1;                   // 0x0078 (0x0004)
[0x0000000000001002] [0x00000001] (CPF_Const | CPF_Native)
unsigned long                    bLinearXPositionDrive : 1;              // 0x0078 (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                    bLinearXVelocityDrive : 1;              // 0x0078 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                    bLinearYPositionDrive : 1;              // 0x0078 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                    bLinearYVelocityDrive : 1;              // 0x0078 (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                    bLinearZPositionDrive : 1;              // 0x0078 (0x0004)
[0x0000000000000003] [0x00000020] (CPF_Edit | CPF_Const)
unsigned long                    bLinearZVelocityDrive : 1;              // 0x0078 (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                    bSwingPositionDrive : 1;               // 0x0078 (0x0004)
[0x0000000000000003] [0x00000080] (CPF_Edit | CPF_Const)
unsigned long                    bSwingVelocityDrive : 1;               // 0x0078 (0x0004)
[0x0000000000000003] [0x00000100] (CPF_Edit | CPF_Const)
unsigned long                    bTwistPositionDrive : 1;               // 0x0078 (0x0004)
[0x0000000000000003] [0x00000200] (CPF_Edit | CPF_Const)
unsigned long                    bTwistVelocityDrive : 1;               // 0x0078 (0x0004)
[0x0000000000000003] [0x00000400] (CPF_Edit | CPF_Const)
unsigned long                    bAngularSlerpDrive : 1;               // 0x0078 (0x0004)
[0x0000000000000003] [0x00000800] (CPF_Edit | CPF_Const)
unsigned long                    bTerminated : 1;                   // 0x0078 (0x0004)
[0x0000000000000000] [0x00001000]
struct FPointer                   ConstraintData;                    // 0x0080 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FVector                    LinearPositionTarget;                  // 0x0088 (0x000C)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FVector                    LinearVelocityTarget;                 // 0x0094 (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                             LinearDriveSpring;                  // 0x00A0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                             LinearDriveDamping;                   // 0x00A4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                             LinearDriveForceLimit;                // 0x00A8 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FQuat                      AngularPositionTarget;                // 0x00B0 (0x0010)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FVector                    AngularVelocityTarget;                // 0x00C0 (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                             AngularDriveSpring;                 // 0x00CC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                             AngularDriveDamping;                  // 0x00D0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                             AngularDriveForceLimit;               // 0x00D4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FPointer                   DummyKinActor;                        // 0x00D8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_ConstraintInstance");
}

return uClassPointer;
};

void MoveKinActorTransform(struct FMatrix& NewTM);
void SetLinearLimitSize(float NewLimitSize);
void SetAngularDOFLimitScale(float InSwing1LimitScale, float InSwing2LimitScale, float
InTwistLimitScale, class URB_ConstraintSetup* InSetup);
void SetAngularDriveParams(float InSpring, float InDamping, float InForceLimit);
void SetAngularVelocityTarget(struct FVector InVelTarget);
void SetAngularPositionTarget(struct FQuat& InPosTarget);
void SetLinearDriveParams(float InSpring, float InDamping, float InForceLimit);
void SetLinearVelocityTarget(struct FVector InVelTarget);
void SetLinearPositionTarget(struct FVector InPosTarget);
void SetAngularVelocityDrive(unsigned long bEnableSwingDrive, unsigned long
bEnableTwistDrive);
void SetAngularPositionDrive(unsigned long bEnableSwingDrive, unsigned long
bEnableTwistDrive);
void SetLinearVelocityDrive(unsigned long bEnableXDrive, unsigned long bEnableYDrive,
unsigned long bEnableZDrive);
void SetLinearPositionDrive(unsigned long bEnableXDrive, unsigned long bEnableYDrive,
unsigned long bEnableZDrive);
struct FVector GetConstraintLocation();
```

```
class UPhysicsAssetInstance* GetPhysicsAssetInstance();
void TermConstraint();
void InitConstraint(class UPrimitiveComponent* PrimComp1, class UPrimitiveComponent*
PrimComp2, class URB_ConstraintSetup* Setup, float Scale, class AActor* InOwner, class
UPrimitiveComponent* InPrimComp, unsigned long bMakeKinForBody1);
};

// Class Engine.RB_ConstraintSetup
// 0x00C4 (0x0060 - 0x0124)
class URB_ConstraintSetup : public UObject
{
public:
struct FName                    JointName;                    // 0x0060 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FName                    ConstraintBone1;                // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                    ConstraintBone2;                // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                  Pos1;                        // 0x0078 (0x000C)
[0x0000000000000000]
struct FVector                  PriAxis1;                     // 0x0084 (0x000C)
[0x0000000000000000]
struct FVector                  SecAxis1;                     // 0x0090 (0x000C)
[0x0000000000000000]
struct FVector                  Pos2;                        // 0x009C (0x000C)
[0x0000000000000000]
struct FVector                  PriAxis2;                     // 0x00A8 (0x000C)
[0x0000000000000000]
struct FVector                  SecAxis2;                     // 0x00B4 (0x000C)
[0x0000000000000000]
struct FVector                  PulleyPivot1;                  // 0x00C0 (0x000C)
[0x0000000000000000]
struct FVector                  PulleyPivot2;                  // 0x00CC (0x000C)
[0x0000000000000000]
unsigned long                   bEnableProjection : 1;            // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                   bLinearLimitSoft : 1;             // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                   bLinearBreakable : 1;             // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                   bSwingLimited : 1;               // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                   bTwistLimited : 1;               // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                   bSwingLimitSoft : 1;             // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                   bTwistLimitSoft : 1;             // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                   bAngularBreakable : 1;            // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                   bIsPulley : 1;                 // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                   bMaintainMinDistance : 1;          // 0x00D8 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
```

```
struct FLinearDOFSetup                    LinearXSetup;                         // 0x00DC (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FLinearDOFSetup                    LinearYSetup;                         // 0x00E4 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FLinearDOFSetup                    LinearZSetup;                         // 0x00EC (0x0008)
[0x0000000000000001] (CPF_Edit)
float                           LinearLimitStiffness;                  // 0x00F4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           LinearLimitDamping;                    // 0x00F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           LinearBreakThreshold;                  // 0x00FC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           Swing1LimitAngle;                      // 0x0100 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           Swing2LimitAngle;                      // 0x0104 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           TwistLimitAngle;                       // 0x0108 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           SwingLimitStiffness;                   // 0x010C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           SwingLimitDamping;                     // 0x0110 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           TwistLimitStiffness;                   // 0x0114 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           TwistLimitDamping;                     // 0x0118 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           AngularBreakThreshold;                 // 0x011C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           PulleyRatio;                           // 0x0120 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_ConstraintSetup");
}

return uClassPointer;
};

};

// Class Engine.RB_BSJointSetup
// 0x0004 (0x0124 - 0x0128)
class URB_BSJointSetup : public URB_ConstraintSetup
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_BSJointSetup");
}

return uClassPointer;
};

};

// Class Engine.RB_DistanceJointSetup
// 0x0004 (0x0124 - 0x0128)
class URB_DistanceJointSetup : public URB_ConstraintSetup
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_DistanceJointSetup");
}

return uClassPointer;
};

};

// Class Engine.RB_HingeSetup
// 0x0004 (0x0124 - 0x0128)
class URB_HingeSetup : public URB_ConstraintSetup
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_HingeSetup");
}

return uClassPointer;
};

};
```

```
// Class Engine.RB_PrismaticSetup
// 0x0004 (0x0124 - 0x0128)
class URB_PrismaticSetup : public URB_ConstraintSetup
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_PrismaticSetup");
}

return uClassPointer;
};

};

// Class Engine.RB_PulleyJointSetup
// 0x0004 (0x0124 - 0x0128)
class URB_PulleyJointSetup : public URB_ConstraintSetup
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_PulleyJointSetup");
}

return uClassPointer;
};

};

// Class Engine.RB_SkelJointSetup
// 0x0004 (0x0124 - 0x0128)
class URB_SkelJointSetup : public URB_ConstraintSetup
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_SkelJointSetup");
}

return uClassPointer;
};

};

// Class Engine.RB_StayUprightSetup
// 0x0004 (0x0124 - 0x0128)
class URB_StayUprightSetup : public URB_ConstraintSetup
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_StayUprightSetup");
}

return uClassPointer;
};

};

// Class Engine.NxGenericForceFieldBrush
// 0x00E4 (0x02A4 - 0x0388)
class ANxGenericForceFieldBrush : public AVolume
{
public:
int32_t                          ExcludeChannel;                        // 0x02A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRBCollisionChannelContainer            CollideWithChannels;                     // 0x02AC
(0x0004) [0x0000000000000001] (CPF_Edit)
uint8_t                          RBChannel;                           // 0x02B0 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                          Coordinates;                         // 0x02B1 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                      Constant;                           // 0x02B4 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                      PositionMultiplierX;                     // 0x02C0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                      PositionMultiplierY;                     // 0x02CC (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                      PositionMultiplierZ;                     // 0x02D8 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                      PositionTarget;                       // 0x02E4 (0x000C)
[0x0000000000000001] (CPF_Edit)
```

```
struct FVector                          VelocityMultiplierX;                    // 0x02F0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          VelocityMultiplierY;                    // 0x02FC (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          VelocityMultiplierZ;                    // 0x0308 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          VelocityTarget;                       // 0x0314 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          Noise;                            // 0x0320 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          FalloffLinear;                       // 0x032C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          FalloffQuadratic;                      // 0x0338 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                              TorusRadius;                        // 0x0344 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                         ForceField;                         // 0x0348 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                      ConvexMeshes;                        // 0x0350 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                      ExclusionShapes;                      // 0x0360 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                      ExclusionShapePoses;                    // 0x0370 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                         LinearKernel;                        // 0x0380 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxGenericForceFieldBrush");
}

return uClassPointer;
};

};

// Class Engine.RB_ForceFieldExcludeVolume
// 0x000C (0x02A4 - 0x02B0)
class ARB_ForceFieldExcludeVolume : public AVolume
{
public:
int32_t                             ForceFieldChannel;                     // 0x02A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             SceneIndex;                         // 0x02AC (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_ForceFieldExcludeVolume");
}

return uClassPointer;
};

};

// Class Engine.NxForceField
// 0x0054 (0x0268 - 0x02BC)
class ANxForceField : public AActor
{
public:
int32_t                                ExcludeChannel;                        // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bForceActive : 1;                      // 0x026C (0x0004)
[0x0000000000000021] [0x00000001] (CPF_Edit | CPF_Net)
struct FRBCollisionChannelContainer            CollideWithChannels;                   // 0x0270
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                                RBChannel;                             // 0x0274 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FPointer                        ForceField;                            // 0x0278 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                     ConvexMeshes;                          // 0x0280 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                     ExclusionShapes;                       // 0x0290 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                     ExclusionShapePoses;                   // 0x02A0 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                        U2NRotation;                           // 0x02B0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                                SceneIndex;                            // 0x02B8 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceField");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* inAction);
void DoInitRBPhys();
```

```
};

// Class Engine.NxCylindricalForceField
// 0x0034 (0x02BC - 0x02F0)
class ANxCylindricalForceField : public ANxForceField
{
public:
float                             RadialStrength;                    // 0x02C0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             RotationalStrength;                // 0x02C4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             LiftStrength;                      // 0x02C8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             ForceRadius;                       // 0x02CC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             ForceTopRadius;                    // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             LiftFalloffHeight;                 // 0x02D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             EscapeVelocity;                    // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             ForceHeight;                       // 0x02DC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             HeightOffset;                      // 0x02E0 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                          UseSpecialRadialForce : 1;            // 0x02E4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FPointer                        Kernel;                         // 0x02E8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxCylindricalForceField");
}

return uClassPointer;
};

};

// Class Engine.NxCylindricalForceFieldCapsule
// 0x0008 (0x02F0 - 0x02F8)
class ANxCylindricalForceFieldCapsule : public ANxCylindricalForceField
{
public:
class UDrawCapsuleComponent*              RenderComponent;                    // 0x02F0
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxCylindricalForceFieldCapsule");
}

return uClassPointer;
};

void DoInitRBPhys();
};

// Class Engine.NxForceFieldGeneric
// 0x00C4 (0x02BC - 0x0380)
class ANxForceFieldGeneric : public ANxForceField
{
public:
class UForceFieldShape*                Shape;                              // 0x02C0 (0x0008)
[0x0000000004000001] (CPF_Edit | CPF_EditInline)
class UActorComponent*                 DrawComponent;                      // 0x02C8 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                                  RoughExtentX;                       // 0x02D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  RoughExtentY;                       // 0x02D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  RoughExtentZ;                       // 0x02D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                Coordinates;                        // 0x02DC (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                         Constant;                           // 0x02E0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         PositionMultiplierX;                // 0x02EC (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         PositionMultiplierY;                // 0x02F8 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         PositionMultiplierZ;                // 0x0304 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         PositionTarget;                     // 0x0310 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         VelocityMultiplierX;                // 0x031C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         VelocityMultiplierY;                // 0x0328 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         VelocityMultiplierZ;                // 0x0334 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         VelocityTarget;                     // 0x0340 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         Noise;                              // 0x034C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         FalloffLinear;                      // 0x0358 (0x000C)
```

```
[0x0000000000000001] (CPF_Edit)
struct FVector                    FalloffQuadratic;               // 0x0364 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                             TorusRadius;                    // 0x0370 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                   LinearKernel;                   // 0x0378 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldGeneric");
}

return uClassPointer;
};

void DoInitRBPhys();
};

// Class Engine.NxForceFieldRadial
// 0x002C (0x02BC - 0x02E8)
class ANxForceFieldRadial : public ANxForceField
{
public:
class UForceFieldShape*           Shape;                          // 0x02C0 (0x0008)
[0x0000000004000001] (CPF_Edit | CPF_EditInline)
class UActorComponent*            DrawComponent;                  // 0x02C8 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                             ForceStrength;                  // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             ForceRadius;                    // 0x02D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                             SelfRotationStrength;           // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
uint8_t                           ForceFalloff;                   // 0x02DC (0x0001)
[0x0000000000080009] (CPF_Edit | CPF_ExportObject | CPF_Component)
struct FPointer                   Kernel;                         // 0x02E0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldRadial");
}
return uClassPointer;
};

void DoInitRBPhys();
};

// Class Engine.NxForceFieldTornado
// 0x004C (0x02BC - 0x0308)
class ANxForceFieldTornado : public ANxForceField
{
public:
class UForceFieldShape*                 Shape;                          // 0x02C0 (0x0008)
[0x0000000004000001] (CPF_Edit | CPF_EditInline)
class UActorComponent*                  DrawComponent;                  // 0x02C8 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                                   RadialStrength;                 // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   RotationalStrength;             // 0x02D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   LiftStrength;                   // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceRadius;                    // 0x02DC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceTopRadius;                 // 0x02E0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   LiftFalloffHeight;              // 0x02E4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   EscapeVelocity;                 // 0x02E8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceHeight;                    // 0x02EC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   HeightOffset;                   // 0x02F0 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                           BSpecialRadialForceMode : 1;    // 0x02F4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                   SelfRotationStrength;           // 0x02F8 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FPointer                         Kernel;                         // 0x0300 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldTornado");
}

return uClassPointer;
};

void DoInitRBPhys();
```

```
};

// Class Engine.NxGenericForceField
// 0x00A4 (0x02BC - 0x0360)
class ANxGenericForceField : public ANxForceField
{
public:
uint8_t                              Coordinates;                          // 0x02C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                       Constant;                             // 0x02C4 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       PositionMultiplierX;                  // 0x02D0 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       PositionMultiplierY;                  // 0x02DC (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       PositionMultiplierZ;                  // 0x02E8 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       PositionTarget;                       // 0x02F4 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       VelocityMultiplierX;                  // 0x0300 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       VelocityMultiplierY;                  // 0x030C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       VelocityMultiplierZ;                  // 0x0318 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       VelocityTarget;                       // 0x0324 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       Noise;                                // 0x0330 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       FalloffLinear;                        // 0x033C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                       FalloffQuadratic;                     // 0x0348 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                TorusRadius;                          // 0x0354 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                      LinearKernel;                         // 0x0358 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxGenericForceField");
}

return uClassPointer;
};

};

// Class Engine.NxGenericForceFieldBox
// 0x0014 (0x0360 - 0x0374)
class ANxGenericForceFieldBox : public ANxGenericForceField
{
public:
class UDrawBoxComponent*                    RenderComponent;                        // 0x0360
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FVector                              BoxExtent;                              // 0x0368 (0x000C)
[0x0000000200000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxGenericForceFieldBox");
}

return uClassPointer;
};

void DoInitRBPhys();
};

// Class Engine.NxGenericForceFieldCapsule
// 0x0010 (0x0360 - 0x0370)
class ANxGenericForceFieldCapsule : public ANxGenericForceField
{
public:
class UDrawCapsuleComponent*                RenderComponent;                        // 0x0360
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                                       CapsuleHeight;                          // 0x0368 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                       CapsuleRadius;                          // 0x036C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxGenericForceFieldCapsule");
}

return uClassPointer;
};

};

// Class Engine.NxRadialForceField
// 0x0024 (0x02BC - 0x02E0)
```

```
class ANxRadialForceField : public ANxForceField
{
public:
class UDrawSphereComponent*                    RenderComponent;                            // 0x02C0
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                                     ForceStrength;                            // 0x02C8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                     ForceRadius;                              // 0x02CC (0x0004)
[0x0000000200000001] (CPF_Edit)
uint8_t                                   ForceFalloff;                             // 0x02D0 (0x0001)
[0x0000000000080009] (CPF_Edit | CPF_ExportObject | CPF_Component)
struct FPointer                           LinearKernel;                             // 0x02D8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxRadialForceField");
}

return uClassPointer;
};

};

// Class Engine.NxRadialCustomForceField
// 0x0010 (0x02E0 - 0x02F0)
class ANxRadialCustomForceField : public ANxRadialForceField
{
public:
float                                     SelfRotationStrength;                     // 0x02E0 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FPointer                           Kernel;                                   // 0x02E8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxRadialCustomForceField");
}

return uClassPointer;
};

};
```

```
// Class Engine.NxTornadoAngularForceField
// 0x003C (0x02BC - 0x02F8)
class ANxTornadoAngularForceField : public ANxForceField
{
public:
float                          RadialStrength;                    // 0x02C0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          RotationalStrength;                // 0x02C4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          LiftStrength;                      // 0x02C8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          ForceRadius;                       // 0x02CC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          ForceTopRadius;                    // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          LiftFalloffHeight;                 // 0x02D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          EscapeVelocity;                    // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          ForceHeight;                       // 0x02DC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          HeightOffset;                      // 0x02E0 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                       BSpecialRadialForceMode : 1;          // 0x02E4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                          SelfRotationStrength;              // 0x02E8 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FPointer                     Kernel;                            // 0x02F0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxTornadoAngularForceField");
}

return uClassPointer;
};

};

// Class Engine.NxTornadoAngularForceFieldCapsule
// 0x0008 (0x02F8 - 0x0300)
class ANxTornadoAngularForceFieldCapsule : public ANxTornadoAngularForceField
{
public:
class UDrawCapsuleComponent*              RenderComponent;                        // 0x02F8
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxTornadoAngularForceFieldCapsule");
}

return uClassPointer;
};

};

// Class Engine.NxTornadoForceField
// 0x0034 (0x02BC - 0x02F0)
class ANxTornadoForceField : public ANxForceField
{
public:
float                                    RadialStrength;                            // 0x02C0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    RotationalStrength;                         // 0x02C4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    LiftStrength;                              // 0x02C8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    ForceRadius;                               // 0x02CC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    ForceTopRadius;                             // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    LiftFalloffHeight;                          // 0x02D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    EscapeVelocity;                             // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    ForceHeight;                               // 0x02DC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    HeightOffset;                              // 0x02E0 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                                    BSpecialRadialForceMode : 1;                // 0x02E4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FPointer                                  Kernel;                                    // 0x02E8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxTornadoForceField");
}

return uClassPointer;
};

};

// Class Engine.NxTornadoForceFieldCapsule
// 0x0008 (0x02F0 - 0x02F8)
class ANxTornadoForceFieldCapsule : public ANxTornadoForceField
{
public:
class UDrawCapsuleComponent*                    RenderComponent;                            // 0x02F0
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxTornadoForceFieldCapsule");
}

return uClassPointer;
};

};

// Class Engine.NxForceFieldSpawnable
// 0x0008 (0x0268 - 0x0270)
class ANxForceFieldSpawnable : public AActor
{
public:
class UNxForceFieldComponent*                   ForceFieldComponent;                        // 0x0268
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldSpawnable");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* inAction);
};

// Class Engine.RB_CylindricalForceActor
// 0x0034 (0x0268 - 0x029C)
class ARB_CylindricalForceActor : public ARigidBodyBase
{
public:
class UDrawCylinderComponent*                   RenderComponent;                             // 0x0268
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
float                                   RadialStrength;                           // 0x0270 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   RotationalStrength;                       // 0x0274 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   LiftStrength;                             // 0x0278 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   LiftFalloffHeight;                        // 0x027C (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   EscapeVelocity;                           // 0x0280 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceRadius;                              // 0x0284 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceTopRadius;                           // 0x0288 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceHeight;                              // 0x028C (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   HeightOffset;                             // 0x0290 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                                bForceActive : 1;                         // 0x0294 (0x0004)
[0x0000000000000021] [0x00000001] (CPF_Edit | CPF_Net)
unsigned long                                bForceApplyToCloth : 1;                   // 0x0294 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                bForceApplyToFluid : 1;                   // 0x0294 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                bForceApplyToRigidBodies : 1;             // 0x0294 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                                bForceApplyToProjectiles : 1;             // 0x0294 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
struct FRBCollisionChannelContainer             CollideWithChannels;                         // 0x0298
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_CylindricalForceActor");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* inAction);
};
```

```
// Class Engine.RB_RadialForceActor
// 0x0024 (0x0268 - 0x028C)
class ARB_RadialForceActor : public ARigidBodyBase
{
public:
class UDrawSphereComponent*          RenderComponent;                    // 0x0268
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                              ForceStrength;                    // 0x0270 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                              ForceRadius;                      // 0x0274 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                              SwirlStrength;                    // 0x0278 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                              SpinTorque;                       // 0x027C (0x0004)
[0x0000000200000001] (CPF_Edit)
uint8_t                             ForceFalloff;                    // 0x0280 (0x0001)
[0x0000000000080009] (CPF_Edit | CPF_ExportObject | CPF_Component)
uint8_t                             RadialForceMode;                   // 0x0281 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                         bForceActive : 1;                 // 0x0284 (0x0004)
[0x0000000000000021] [0x00000001] (CPF_Edit | CPF_Net)
unsigned long                         bForceApplyToCloth : 1;              // 0x0284 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                         bForceApplyToFluid : 1;              // 0x0284 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                         bForceApplyToRigidBodies : 1;           // 0x0284 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                         bForceApplyToProjectiles : 1;           // 0x0284 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
struct FRBCollisionChannelContainer          CollideWithChannels;                 // 0x0288
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_RadialForceActor");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* inAction);
};

// Class Engine.NxForceFieldComponent
// 0x0070 (0x0258 - 0x02C8)
class UNxForceFieldComponent : public UPrimitiveComponent
{
public:
class UForceFieldShape*                Shape;                            // 0x0258 (0x0008)
```

```
[0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
class UActorComponent*                    DrawComponent;                        // 0x0260 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
int32_t                              ExcludeChannel;                       // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bForceActive : 1;                      // 0x026C (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bDestroyWhenInactive : 1;                 // 0x026C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FRBCollisionChannelContainer             CollideWithChannels;                        // 0x0270
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
float                                Duration;                             // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                          ForceField;                             // 0x0278 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                       ConvexMeshes;                           // 0x0280 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                       ExclusionShapes;                         // 0x0290 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FPointer>                       ExclusionShapePoses;                     // 0x02A0 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                              SceneIndex;                            // 0x02B0 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                                ElapsedTime;                           // 0x02B4 (0x0004)
[0x0000000000000000]
class UPrimitiveComponent*                   RenderComponent;                          // 0x02B8
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FPointer                          RBPhysScene;                              // 0x02C0 (0x0008)
[0x0000000000001000] (CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldComponent");
}

return uClassPointer;
};

void DoInitRBPhys();
};

// Class Engine.NxForceFieldCylindricalComponent
// 0x0030 (0x02C8 - 0x02F8)
class UNxForceFieldCylindricalComponent : public UNxForceFieldComponent
{
public:
float                                RadialStrength;                         // 0x02C8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                RotationalStrength;                      // 0x02CC (0x0004)
```

```
[0x0000000200000001] (CPF_Edit)
float                          LiftStrength;                  // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          ForceRadius;                   // 0x02D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          ForceTopRadius;                // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          LiftFalloffHeight;             // 0x02DC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          EscapeVelocity;                // 0x02E0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          ForceHeight;                   // 0x02E4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          HeightOffset;                  // 0x02E8 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                  UseSpecialRadialForce : 1;     // 0x02EC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FPointer                Kernel;                        // 0x02F0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldCylindricalComponent");
}

return uClassPointer;
};

};

// Class Engine.NxForceFieldGenericComponent
// 0x00B0 (0x02C8 - 0x0378)
class UNxForceFieldGenericComponent : public UNxForceFieldComponent
{
public:
float                          RoughExtentX;                  // 0x02C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          RoughExtentY;                  // 0x02CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          RoughExtentZ;                  // 0x02D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                        Coordinates;                   // 0x02D4 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FVector                 Constant;                      // 0x02D8 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                 PositionMultiplierX;           // 0x02E4 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                 PositionMultiplierY;           // 0x02F0 (0x000C)
[0x0000000000000001] (CPF_Edit)
```

```
struct FVector                          PositionMultiplierZ;                    // 0x02FC (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          PositionTarget;                         // 0x0308 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          VelocityMultiplierX;                    // 0x0314 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          VelocityMultiplierY;                    // 0x0320 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          VelocityMultiplierZ;                    // 0x032C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          VelocityTarget;                         // 0x0338 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          Noise;                                  // 0x0344 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          FalloffLinear;                          // 0x0350 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                          FalloffQuadratic;                       // 0x035C (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                   TorusRadius;                            // 0x0368 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                         Kernel;                                 // 0x0370 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldGenericComponent");
}

return uClassPointer;
};

};

// Class Engine.NxForceFieldRadialComponent
// 0x0018 (0x02C8 - 0x02E0)
class UNxForceFieldRadialComponent : public UNxForceFieldComponent
{
public:
float                                   ForceStrength;                          // 0x02C8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceRadius;                            // 0x02CC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   SelfRotationStrength;                   // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
uint8_t                                 ForceFalloff;                           // 0x02D4 (0x0001)
[0x0000000000080009] (CPF_Edit | CPF_ExportObject | CPF_Component)
struct FPointer                         Kernel;                                 // 0x02D8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldRadialComponent");
}

return uClassPointer;
};

};

// Class Engine.NxForceFieldTornadoComponent
// 0x0038 (0x02C8 - 0x0300)
class UNxForceFieldTornadoComponent : public UNxForceFieldComponent
{
public:
float                                   RadialStrength;                         // 0x02C8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   RotationalStrength;                     // 0x02CC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   LiftStrength;                           // 0x02D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceRadius;                            // 0x02D4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceTopRadius;                         // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   LiftFalloffHeight;                      // 0x02DC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   EscapeVelocity;                         // 0x02E0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ForceHeight;                            // 0x02E4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   HeightOffset;                           // 0x02E8 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                           BSpecialRadialForceMode : 1;            // 0x02EC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                   SelfRotationStrength;                   // 0x02F0 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FPointer                         Kernel;                                 // 0x02F8 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NxForceFieldTornadoComponent");
}
return uClassPointer;
};

};

// Class Engine.ForceFieldShape
// 0x0000 (0x0060 - 0x0060)
class UForceFieldShape : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForceFieldShape");
}

return uClassPointer;
};

class UPrimitiveComponent* eventGetDrawComponent();
void eventFillByCylinder(float BottomRadius, float TopRadius, float Height, float HeightOffset);
void eventFillByCapsule(float Height, float Radius);
void eventFillByBox(struct FVector Dimension);
void eventFillBySphere(float Radius);
};

// Class Engine.ForceFieldShapeBox
// 0x0008 (0x0060 - 0x0068)
class UForceFieldShapeBox : public UForceFieldShape
{
public:
class UDrawBoxComponent*                    Shape;                              // 0x0060 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForceFieldShapeBox");
}

return uClassPointer;
};

class UPrimitiveComponent* eventGetDrawComponent();
```

```
void eventFillByCylinder(float BottomRadius, float TopRadius, float Height, float HeightOffset);
void eventFillByCapsule(float Height, float Radius);
void eventFillByBox(struct FVector Extent);
void eventFillBySphere(float Radius);
struct FVector eventGetRadii();
};

// Class Engine.ForceFieldShapeCapsule
// 0x0008 (0x0060 - 0x0068)
class UForceFieldShapeCapsule : public UForceFieldShape
{
public:
class UDrawCapsuleComponent*                    Shape;                                  // 0x0060 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForceFieldShapeCapsule");
}

return uClassPointer;
};

class UPrimitiveComponent* eventGetDrawComponent();
void eventFillByCylinder(float BottomRadius, float TopRadius, float Height, float HeightOffset);
void eventFillByCapsule(float Height, float Radius);
void eventFillByBox(struct FVector Extent);
void eventFillBySphere(float Radius);
float eventGetRadius();
float eventGetHeight();
};

// Class Engine.ForceFieldShapeSphere
// 0x0008 (0x0060 - 0x0068)
class UForceFieldShapeSphere : public UForceFieldShape
{
public:
class UDrawSphereComponent*                     Shape;                                  // 0x0060 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForceFieldShapeSphere");
}
return uClassPointer;
};

class UPrimitiveComponent* eventGetDrawComponent();
void eventFillByCylinder(float BottomRadius, float TopRadius, float Height, float HeightOffset);
void eventFillByCapsule(float Height, float Radius);
void eventFillByBox(struct FVector Extent);
void eventFillBySphere(float Radius);
float eventGetRadius();
};

// Class Engine.PrefabInstance
// 0x0100 (0x0268 - 0x0368)
class APrefabInstance : public AActor
{
public:
class UPrefab*                          TemplatePrefab;                          // 0x0268 (0x0008)
[0x0000000000000002] (CPF_Const)
int32_t                          TemplateVersion;                          // 0x0270 (0x0004)
[0x0000000000000002] (CPF_Const)
uint8_t                          UnknownData00[0x50];                     // 0x0278 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.PrefabInstance.ArchetypeToInstanceMap
class UPrefabSequence*                          SequenceInstance;                          // 0x02C8 (0x0008)
[0x0000000000000002] (CPF_Const)
int32_t                          PI_PackageVersion;                          // 0x02D0 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                          PI_LicenseePackageVersion;                  // 0x02D4 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<uint8_t>                          PI_Bytes;                          // 0x02D8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class UObject*>                          PI_CompleteObjects;                          // 0x02E8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class UObject*>                          PI_ReferencedObjects;                          // 0x02F8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class FString>                          PI_SavedNames;                          // 0x0308 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
uint8_t                          UnknownData01[0x50];                     // 0x0318 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.PrefabInstance.PI_ObjectMap

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrefabInstance");
}

return uClassPointer;
};

};
```

```
// Class Engine.Prefab
// 0x003C (0x0060 - 0x009C)
class UPrefab : public UObject
{
public:
int32_t                          PrefabVersion;                     // 0x0060 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<class UObject*>                PrefabArchetypes;                   // 0x0068 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<class UObject*>                RemovedArchetypes;                   // 0x0078 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
class UPrefabSequence*                 PrefabSequence;                   // 0x0088 (0x0008)
[0x0000000000000002] (CPF_Const)
class UTexture2D*                   PrefabPreview;                    // 0x0090 (0x0008)
[0x0000000800000002] (CPF_Const)
unsigned long                     bWorldspacePrefab : 1;               // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                     bAutoUpdatePrefabInstances : 1;          // 0x0098 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Prefab");
}

return uClassPointer;
};

};

// Class Engine.SequenceObject
// 0x0078 (0x0060 - 0x00D8)
class USequenceObject : public UObject
{
public:
int32_t                          ObjInstanceVersion;                  // 0x0060 (0x0004)
[0x0000000000000002] (CPF_Const)
class USequence*                    ParentSequence;                    // 0x0068 (0x0008)
[0x0000000001000002] (CPF_Const)
int32_t                          ObjPosX;                        // 0x0070 (0x0004)
[0x0000000800000000]
int32_t                          ObjPosY;                        // 0x0074 (0x0004)
[0x0000000800000000]
class FString                      ObjName;                        // 0x0078 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                      ObjCategory;                      // 0x0088 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
TArray<class FString>                 ObjRemoveInProject;                  // 0x0098 (0x0010)
```

```
[0x0000000800400000] (CPF_NeedCtorLink)
struct FColor                          ObjColor;                              // 0x00A8 (0x0004)
[0x0000000800000000]
class FString                          ObjComment;                            // 0x00B0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                          bDeletable : 1;                        // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                          bDrawFirst : 1;                        // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                          bDrawLast : 1;                         // 0x00C0 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                          bOutputObjCommentToScreen : 1;               // 0x00C0
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bSuppressAutoComment : 1;                   // 0x00C0 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
int32_t                             DrawWidth;                             // 0x00C4 (0x0004)
[0x0000000000000000]
int32_t                             DrawHeight;                            // 0x00C8 (0x0004)
[0x0000000000000000]
class USequenceObject*                      PIESequenceObject;                      // 0x00D0 (0x0008)
[0x0000000C00002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceObject");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
bool eventShouldClearNameOnPasting();
bool eventIsPastingIntoLevelSequenceAllowed();
bool eventIsValidLevelSequenceObject();
class AWorldInfo* GetWorldInfo();
void ScriptLog(class FString LogText, unsigned long bWarning);
};

// Class Engine.SequenceFrame
// 0x0028 (0x00D8 - 0x0100)
class USequenceFrame : public USequenceObject
{
public:
int32_t                             SizeX;                              // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             SizeY;                              // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             BorderWidth;                           // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
unsigned long                    bDrawBox : 1;                    // 0x00E4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bFilled : 1;                    // 0x00E4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bTileFill : 1;                    // 0x00E4 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
struct FColor                    BorderColor;                    // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                    FillColor;                    // 0x00EC (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                    FillTexture;                    // 0x00F0 (0x0008)
[0x0000000800000001] (CPF_Edit)
class UMaterial*                    FillMaterial;                    // 0x00F8 (0x0008)
[0x0000000800000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceFrame");
}

return uClassPointer;
};

};

// Class Engine.SequenceFrameWrapped
// 0x0000 (0x0100 - 0x0100)
class USequenceFrameWrapped : public USequenceFrame
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceFrameWrapped");
}

return uClassPointer;
};

};

// Class Engine.SequenceOp
// 0x0068 (0x00D8 - 0x0140)
class USequenceOp : public USequenceObject
```

```
{
public:
unsigned long                    bIsActivated : 1;                    // 0x00D8 (0x0004)
[0x0000000C01202000] [0x00000001] (CPF_Transient)
unsigned long                    bIsCurrentDebuggerOp : 1;                 // 0x00D8 (0x0004)
[0x0000000C01202000] [0x00000002] (CPF_Transient)
unsigned long                    bActive : 1;                      // 0x00D8 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                    bLatentExecution : 1;                   // 0x00D8 (0x0004)
[0x0000000000000002] [0x00000008] (CPF_Const)
unsigned long                    bAutoActivateOutputLinks : 1;              // 0x00D8 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                    bHaveMovingVarConnector : 1;               // 0x00D8 (0x0004)
[0x0000000800002000] [0x00000020] (CPF_Transient)
unsigned long                    bHaveMovingInputConnector : 1;             // 0x00D8
(0x0004) [0x0000000800002000] [0x00000040] (CPF_Transient)
unsigned long                    bHaveMovingOutputConnector : 1;             // 0x00D8
(0x0004) [0x0000000800002000] [0x00000080] (CPF_Transient)
unsigned long                    bPendingVarConnectorRecalc : 1;             // 0x00D8
(0x0004) [0x0000000800002000] [0x00000100] (CPF_Transient)
unsigned long                    bPendingInputConnectorRecalc : 1;           // 0x00D8
(0x0004) [0x0000000800002000] [0x00000200] (CPF_Transient)
unsigned long                    bPendingOutputConnectorRecalc : 1;          // 0x00D8
(0x0004) [0x0000000800002000] [0x00000400] (CPF_Transient)
unsigned long                    bIsBreakpointSet : 1;                  // 0x00D8 (0x0004)
[0x0000000800000000] [0x00000800]
unsigned long                    bIsHiddenBreakpointSet : 1;                // 0x00D8 (0x0004)
[0x0000000C01202000] [0x00001000] (CPF_Transient)
float                       PIEActivationTime;                    // 0x00DC (0x0004)
[0x0000000800002000] (CPF_Transient)
class USequenceOp*                       ActivatorSeqOp;                     // 0x00E0 (0x0008)
[0x0000000800002000] (CPF_Transient)
int32_t                      LastActivatedInputLink;                  // 0x00E8 (0x0004)
[0x0000000800002000] (CPF_Transient)
int32_t                      LastActivatedOutputLink;                  // 0x00EC (0x0004)
[0x0000000800002000] (CPF_Transient)
TArray<struct FSeqOpInputLink>              InputLinks;                         // 0x00F0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FSeqOpOutputLink>              OutputLinks;                        // 0x0100 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FSeqVarLink>                VariableLinks;                       // 0x0110 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FSeqEventLink>               EventLinks;                         // 0x0120 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                      PlayerIndex;                       // 0x0130 (0x0004)
[0x0000000001002000] (CPF_Transient)
uint8_t                      GamepadID;                        // 0x0134 (0x0001)
[0x0000000001002000] (CPF_Transient)
int32_t                      ActivateCount;                      // 0x0138 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                      SearchTag;                        // 0x013C (0x0004)
[0x0000000000202002] (CPF_Const | CPF_Transient)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceOp");
}

return uClassPointer;
};

void ForceActivateOutput(int32_t OutputIdx);
void ForceActivateInput(int32_t InputIdx);
class AController* GetController(class AActor* TheActor);
class APawn* GetPawn(class AActor* TheActor);
void Reset();
void PublishLinkedVariableValues();
void PopulateLinkedVariableValues();
void eventVersionUpdated(int32_t OldVersion, int32_t NewVersion);
void eventDeactivated();
void eventActivated();
bool ActivateNamedOutputLink(class FString LinkDesc);
bool ActivateOutputLink(int32_t OutputIdx);
void LinkedVariables(class UClass* VarClass, class FString inDesc, class USequenceVariable*&
OutVariable);
void GetBoolVars(class FString inDesc, TArray<uint8_t>& boolVars);
void GetInterpDataVars(class FString inDesc, TArray<class UInterpData*>& outIData);
void GetObjectVarsW(class FString inDesc, TArray<class UObject*>& objVars);
void GetLinkedObjects(class UClass* ObjectType, unsigned long bRecurse, TArray<class
USequenceObject*>& out_Objects);
bool HasLinkedOps(unsigned long bConsiderInputLinks);
};

// Class Engine.Sequence
// 0x008C (0x0140 - 0x01CC)
class USequence : public USequenceOp
{
public:
struct FPointer                     LogFile;                        // 0x0140 (0x0008)
[0x0000000000000002] (CPF_Const)
TArray<class USequenceObject*>           SequenceObjects;                 // 0x0148
(0x0010) [0x000000000040000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink)
TArray<class USequenceOp*>               ActiveSequenceOps;               // 0x0158
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class USequence*>                 NestedSequences;                // 0x0168
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class USequenceEvent*>            UnregisteredEvents;              // 0x0178
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FActivateOp>               DelayedActivatedOps;            // 0x0188
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class USequenceOp*>               DelayedLatentOps;                // 0x0198
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
unsigned long                       bEnabled : 1;                    // 0x01A8 (0x0004)
```

```
[0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<struct FQueuedActivationInfo>            QueuedActivations;                        // 0x01B0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
int32_t                               DefaultViewX;                         // 0x01C0 (0x0004)
[0x0000000000000000]
int32_t                               DefaultViewY;                         // 0x01C4 (0x0004)
[0x0000000000000000]
float                                 DefaultViewZoom;                        // 0x01C8 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Sequence");
}

return uClassPointer;
};

void SetEnabled(unsigned long bInEnabled);
void Reset();
void FindSeqObjectsByName(class FString SeqObjName, unsigned long bCheckComment,
unsigned long bRecursive, unsigned long bUseFullLevelName, TArray<class
USequenceObject*>& OutputObjects);
void FindSeqObjectsByClass(class UClass* DesiredClass, unsigned long bRecursive,
TArray<class USequenceObject*>& OutputObjects);
};

// Class Engine.PrefabSequence
// 0x000C (0x01CC - 0x01D8)
class UPrefabSequence : public USequence
{
public:
class APrefabInstance*                    OwnerPrefab;                          // 0x01D0 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrefabSequence");
}

return uClassPointer;
};

class APrefabInstance* GetOwnerPrefab();
```

```
void SetOwnerPrefab(class APrefabInstance* InOwner);
};

// Class Engine.PrefabSequenceContainer
// 0x0004 (0x01CC - 0x01D0)
class UPrefabSequenceContainer : public USequence
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrefabSequenceContainer");
}

return uClassPointer;
};

};

// Class Engine.SequenceAction
// 0x0020 (0x0140 - 0x0160)
class USequenceAction : public USequenceOp
{
public:
struct FName                              HandlerName;                              // 0x0140 (0x0008)
[0x0000000000000000]
unsigned long                             bCallHandler : 1;                         // 0x0148 (0x0004)
[0x0000000000000000] [0x00000001]
TArray<class UObject*>                         Targets;                             // 0x0150 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceAction");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ActivateRemoteEvent
// 0x0014 (0x0160 - 0x0174)
class USeqAct_ActivateRemoteEvent : public USequenceAction
```

```
{
public:
class AActor*                        Instigator;                        // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                         EventName;                         // 0x0168 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                        bStatusIsOk : 1;                   // 0x0170 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ActivateRemoteEvent");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_AndGate
// 0x0028 (0x0160 - 0x0188)
class USeqAct_AndGate : public USequenceAction
{
public:
unsigned long                        bOpen : 1;                         // 0x0160 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
TArray<unsigned long>                LinkedOutputFiredStatus;           // 0x0168
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FPointer>              LinkedOutputs;                     // 0x0178 (0x0010)
[0x0000000000003000] (CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AndGate");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ApplySoundNode
// 0x0010 (0x0160 - 0x0170)
```

```
class USeqAct_ApplySoundNode : public USequenceAction
{
public:
class USoundCue*                    PlaySound;                    // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USoundNode*                    ApplyNode;                    // 0x0168 (0x0008)
[0x0000000004000001] (CPF_Edit | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ApplySoundNode");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_AttachToEvent
// 0x0004 (0x0160 - 0x0164)
class USeqAct_AttachToEvent : public USequenceAction
{
public:
unsigned long                    bPreferController : 1;              // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AttachToEvent");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_CameraFade
// 0x0030 (0x0160 - 0x0190)
class USeqAct_CameraFade : public USequenceAction
{
public:
struct FColor                    FadeColor;                    // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                    FadeAlpha;                    // 0x0164 (0x0008)
[0x0000000020000000]  CPF_Deprecated)
float                          FadeOpacity;                     // 0x016C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FadeTime;                        // 0x0170 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                  bPersistFade : 1;                // 0x0174 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                  bFadeAudio : 1;                  // 0x0174 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                          FadeTimeRemaining;               // 0x0178 (0x0004)
[0x0000000000000000]
TArray<class APlayerController*>          CachedPCs;                    // 0x0180 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_CameraFade");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_CameraLookAt
// 0x003C (0x0160 - 0x019C)
class USeqAct_CameraLookAt : public USequenceAction
{
public:
unsigned long                  bAffectCamera : 1;               // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                  bAlwaysFocus : 1;                // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                  bAdjustCamera : 1;               // 0x0160 (0x0004)
[0x0000000020000000] [0x00000004]  CPF_Deprecated)
unsigned long                  bTurnInPlace : 1;                // 0x0160 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                  bIgnoreTrace : 1;                // 0x0160 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                  bAffectHead : 1;                 // 0x0160 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                  bRotatePlayerWithCamera : 1;            // 0x0160 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                  bToggleGodMode : 1;                // 0x0160 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                  bLeaveCameraRotation : 1;            // 0x0160 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                  bDisableInput : 1;               // 0x0160 (0x0004)
```

```
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                    bUsedTimer : 1;                    // 0x0160 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                    bCheckLineOfSight : 1;                  // 0x0160 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
struct FVector2D                  InterpSpeedRange;                    // 0x0164 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                  InFocusFOV;                       // 0x016C (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                     FocusBoneName;                       // 0x0174 (0x0008)
[0x0000000000000001] (CPF_Edit)
class FString                   TextDisplay;                       // 0x0180 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                       TotalTime;                        // 0x0190 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       CameraFOV;                         // 0x0194 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                       RemainingTime;                       // 0x0198 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_CameraLookAt");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_CameraShake
// 0x0028 (0x0160 - 0x0188)
class USeqAct_CameraShake : public USequenceAction
{
public:
class UCameraShake*                Shake;                          // 0x0160 (0x0008)
[0x0000008004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
float                       ShakeScale;                        // 0x0168 (0x0004)
[0x0000008000000001] (CPF_Edit)
unsigned long                    bDoControllerVibration : 1;               // 0x016C (0x0004)
[0x0000008000000001] [0x00000001] (CPF_Edit)
unsigned long                    bRadialShake : 1;                   // 0x016C (0x0004)
[0x0000008000000001] [0x00000002] (CPF_Edit)
unsigned long                    bOrientTowardRadialEpicenter : 1;           // 0x016C (0x0004)
[0x0000008000000001] [0x00000004] (CPF_Edit)
float                       RadialShake_InnerRadius;               // 0x0170 (0x0004)
[0x0000008000000001] (CPF_Edit)
float                       RadialShake_OuterRadius;               // 0x0174 (0x0004)
```

```
[0x0000008000000001] (CPF_Edit)
float                           RadialShake_Falloff;                    // 0x0178 (0x0004)
[0x0000008000000001] (CPF_Edit)
uint8_t                         PlaySpace;                              // 0x017C (0x0001)
[0x0000008000000001] (CPF_Edit)
class AActor*                   LocationActor;                          // 0x0180 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_CameraShake");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_ChangeCollision
// 0x0005 (0x0160 - 0x0165)
class USeqAct_ChangeCollision : public USequenceAction
{
public:
unsigned long                   bCollideActors : 1;                     // 0x0160 (0x0004)
[0x0000000000020003] [0x00000001] (CPF_Edit | CPF_Const | CPF_EditConst)
unsigned long                   bBlockActors : 1;                       // 0x0160 (0x0004)
[0x0000000000020003] [0x00000002] (CPF_Edit | CPF_Const | CPF_EditConst)
unsigned long                   bIgnoreEncroachers : 1;                 // 0x0160 (0x0004)
[0x0000000000020003] [0x00000004] (CPF_Edit | CPF_Const | CPF_EditConst)
uint8_t                         CollisionType;                          // 0x0164 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ChangeCollision");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};
```

```
// Class Engine.SeqAct_CommitMapChange
// 0x0000 (0x0160 - 0x0160)
class USeqAct_CommitMapChange : public USequenceAction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_CommitMapChange");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ConvertToString
// 0x001C (0x0160 - 0x017C)
class USeqAct_ConvertToString : public USequenceAction
{
public:
unsigned long                            bIncludeVarComment : 1;                  // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
class FString                            VarSeparator;                            // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                                  NumberOfInputs;                          // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ConvertToString");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_DrawText
// 0x0048 (0x0160 - 0x01A8)
class USeqAct_DrawText : public USequenceAction
{
public:
```

```
float                                   DisplayTimeSeconds;                        // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bDisplayOnObject : 1;                      // 0x0164 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FKismetDrawTextInfo                      DrawTextInfo;                              // 0x0168 (0x0040)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_DrawText");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_FinishSequence
// 0x0010 (0x0160 - 0x0170)
class USeqAct_FinishSequence : public USequenceAction
{
public:
class FString                           OutputLabel;                               // 0x0160 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_FinishSequence");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_Gate
// 0x000C (0x0160 - 0x016C)
class USeqAct_Gate : public USequenceAction
{
public:
unsigned long                           bOpen : 1;                                 // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                                 AutoCloseCount;                            // 0x0164 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
int32_t                         CurrentCloseCount;                // 0x0168 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Gate");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_GetDistance
// 0x0004 (0x0160 - 0x0164)
class USeqAct_GetDistance : public USequenceAction
{
public:
float                          Distance;                      // 0x0160 (0x0004)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_GetDistance");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_GetLocationAndRotation
// 0x002C (0x0160 - 0x018C)
class USeqAct_GetLocationAndRotation : public USequenceAction
{
public:
struct FVector                  Location;                     // 0x0160 (0x000C)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FVector                  RotationVector;                 // 0x016C (0x000C)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FVector                  Rotation;                     // 0x0178 (0x000C)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FName                    SocketOrBoneName;                 // 0x0184 (0x0008)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_GetLocationAndRotation");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_GetProperty
// 0x0008 (0x0160 - 0x0168)
class USeqAct_GetProperty : public USequenceAction
{
public:
struct FName                    PropertyName;                  // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_GetProperty");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_GetVectorComponents
// 0x0018 (0x0160 - 0x0178)
class USeqAct_GetVectorComponents : public USequenceAction
{
public:
struct FVector                  InVector;                      // 0x0160 (0x000C)
[0x0000000000000000]
float                      X;                        // 0x016C (0x0004)
[0x0000000000000000]
float                      Y;                        // 0x0170 (0x0004)
[0x0000000000000000]
float                      Z;                        // 0x0174 (0x0004)
[0x0000000000000000]
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_GetVectorComponents");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_GetVelocity
// 0x0010 (0x0160 - 0x0170)
class USeqAct_GetVelocity : public USequenceAction
{
public:
float                                   VelocityMag;                           // 0x0160 (0x0004)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FVector                          VelocityVect;                          // 0x0164 (0x000C)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_GetVelocity");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_HeadTrackingControl
// 0x00A8 (0x0160 - 0x0208)
class USeqAct_HeadTrackingControl : public USequenceAction
{
public:
TArray<struct FName>                    TrackControllerName;                   // 0x0160 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                   LookAtActorRadius;                     // 0x0170 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bDisableBeyondLimit : 1;               // 0x0174 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bLookAtPawns : 1;                      // 0x0174 (0x0004)
```

```
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                              MaxLookAtTime;                      // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              MinLookAtTime;                      // 0x017C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              MaxInterestTime;                    // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class UClass*>                         ActorClassesToLookAt;                    // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FName>                          TargetBoneNames;                         // 0x0198 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UObject*>                        LookAtTargets;                        // 0x01A8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
uint8_t                             UnknownData00[0x50];                    // 0x01B8 (0x0050)
UNKNOWN PROPERTY: MapProperty
Engine.SeqAct_HeadTrackingControl.ActorToComponentMap

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_HeadTrackingControl");
}

return uClassPointer;
};

void eventActivated();
static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_IsInObjectList
// 0x0004 (0x0160 - 0x0164)
class USeqAct_IsInObjectList : public USequenceAction
{
public:
unsigned long                       bCheckForAllObjects : 1;                 // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bObjectFound : 1;                      // 0x0160 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_IsInObjectList");
}
return uClassPointer;
};

};

// Class Engine.SeqAct_Latent
// 0x0018 (0x0160 - 0x0178)
class USeqAct_Latent : public USequenceAction
{
public:
TArray<class AActor*>                    LatentActors;                      // 0x0160 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                        bAborted : 1;                    // 0x0170 (0x0004)
[0x0000000000000000] [0x00000001]
float                             LatentActivationTime;                 // 0x0174 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Latent");
}

return uClassPointer;
};

bool eventUpdate(float DeltaTime);
void AbortFor(class AActor* latentActor);
};

// Class Engine.SeqAct_ActorFactory
// 0x0060 (0x0178 - 0x01D8)
class USeqAct_ActorFactory : public USeqAct_Latent
{
public:
unsigned long                        bEnabled : 1;                    // 0x0178 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bIsSpawning : 1;                   // 0x0178 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                        bCheckSpawnCollision : 1;              // 0x0178 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
class UActorFactory*                     Factory;                        // 0x0180 (0x0008)
[0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
uint8_t                            PointSelection;                   // 0x0188 (0x0001)
[0x0000000000000001] (CPF_Edit)
TArray<class AActor*>                    SpawnPoints;                      // 0x0190 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FVector>                   SpawnLocations;                     // 0x01A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FVector>                   SpawnOrientations;                   // 0x01B0 (0x0010)
```

```
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                         SpawnCount;                         // 0x01C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           SpawnDelay;                         // 0x01C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         LastSpawnIdx;                       // 0x01C8 (0x0004)
[0x0000000000000000]
int32_t                         CurrentSpawnIdx;                    // 0x01CC (0x0004)
[0x0000000000000000]
int32_t                         SpawnedCount;                       // 0x01D0 (0x0004)
[0x0000000000000000]
float                           RemainingDelay;                     // 0x01D4 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ActorFactory");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_ActorFactoryEx
// 0x0000 (0x01D8 - 0x01D8)
class USeqAct_ActorFactoryEx : public USeqAct_ActorFactory
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ActorFactoryEx");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_AIMoveToActor
// 0x002C (0x0178 - 0x01A4)
class USeqAct_AIMoveToActor : public USeqAct_Latent
```

```
{
public:
unsigned long                    bInterruptable : 1;                  // 0x0178 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bPickClosest : 1;                    // 0x0178 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
TArray<class AActor*>                Destination;                     // 0x0180 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                        MovementSpeedModifier;                   // 0x0190 (0x0004)
[0x0000000000000001] (CPF_Edit)
class AActor*                    LookAt;                          // 0x0198 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                       LastDestinationChoice;                  // 0x01A0 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AIMoveToActor");
}

return uClassPointer;
};

class AActor* PickDestination(class AActor* Requestor);
static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_Delay
// 0x0014 (0x0178 - 0x018C)
class USeqAct_Delay : public USeqAct_Latent
{
public:
unsigned long                    bDelayActive : 1;                    // 0x0178 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                    bStartWillRestart : 1;                 // 0x0178 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                        DefaultDuration;                     // 0x017C (0x0004)
[0x0000000000000002] (CPF_Const)
float                        Duration;                        // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        LastUpdateTime;                      // 0x0184 (0x0004)
[0x0000000000000002] (CPF_Const)
float                        RemainingTime;                       // 0x0188 (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Delay");
}

return uClassPointer;
};

void ResetDelayActive();
void Reset();
};

// Class Engine.SeqAct_DelaySwitch
// 0x0010 (0x0178 - 0x0188)
class USeqAct_DelaySwitch : public USeqAct_Latent
{
public:
int32_t                                   LinkCount;                               // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                   CurrentIdx;                              // 0x017C (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                     SwitchDelay;                             // 0x0180 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                                     NextLinkTime;                              // 0x0184 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_DelaySwitch");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ForceGarbageCollection
// 0x0000 (0x0178 - 0x0178)
class USeqAct_ForceGarbageCollection : public USeqAct_Latent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ForceGarbageCollection");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_Interp
// 0x0110 (0x0178 - 0x0288)
class USeqAct_Interp : public USeqAct_Latent
{
public:
uint8_t                                    UnknownData00[0x50];                      // 0x0178 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.SeqAct_Interp.SavedActorTransforms
uint8_t                                    UnknownData01[0x50];                      // 0x01C8 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.SeqAct_Interp.SavedActorVisibilities
float                                      PlayRate;                                 // 0x0218 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      Position;                                 // 0x021C (0x0004)
[0x0000000000000000]
float                                      ForceStartPosition;                       // 0x0220 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                              bIsPlaying : 1;                           // 0x0224 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                              bPaused : 1;                              // 0x0224 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                              bIsBeingEdited : 1;                       // 0x0224 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                              bLooping : 1;                             // 0x0224 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                              bRewindOnPlay : 1;                        // 0x0224 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                              bNoResetOnRewind : 1;                     // 0x0224 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                              bRewindIfAlreadyPlaying : 1;              // 0x0224 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                              bReversePlayback : 1;                     // 0x0224 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                              bInterpForPathBuilding : 1;               // 0x0224 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                              bForceStartPos : 1;                       // 0x0224 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                              bDisableRadioFilter : 1;                  // 0x0224 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                              bClientSideOnly : 1;                      // 0x0224 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                              bSkipUpdateIfNotVisible : 1;              // 0x0224 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                              bIsSkippable : 1;                         // 0x0224 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                              bActivateDumpMovieOnStart : 1;            // 0x0224 (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
```

```
unsigned long                       bShouldShowGore : 1;                    // 0x0224 (0x0004)
[0x0000000000002000] [0x00008000] (CPF_Transient)
TArray<class ACoverLink*>                   LinkedCover;                       // 0x0228 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class UInterpData*                       InterpData;                        // 0x0238 (0x0008)
[0x0000000000000008] (CPF_ExportObject)
TArray<class UInterpGroupInst*>                GroupInst;                         // 0x0240 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class UClass*                       ReplicatedActorClass;                  // 0x0250 (0x0008)
[0x0000000000000002] (CPF_Const)
class AMatineeActor*                       ReplicatedActor;                      // 0x0258 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                       PreferredSplitScreenNum;                // 0x0260 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FCameraCutInfo>                CameraCuts;                          // 0x0268 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                       TerminationTime;                      // 0x0278 (0x0004)
[0x0000000000000000]
struct FRenderingPerformanceOverrides            RenderingOverrides;                  // 0x027C
(0x0004) [0x0000000000044001] (CPF_Edit | CPF_Config | CPF_GlobalConfig)
uint8_t                       ConstantCameraAnim;                  // 0x0280 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                       ConstantCameraAnimRate;                // 0x0284 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Interp");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
void Reset();
void AddPlayerToDirectorTracks(class APlayerController* PC);
void Stop();
void SetPosition(float NewPosition, unsigned long bJump);
};

// Class Engine.SeqAct_LevelStreamingBase
// 0x0004 (0x0178 - 0x017C)
class USeqAct_LevelStreamingBase : public USeqAct_Latent
{
public:
unsigned long                       bMakeVisibleAfterLoad : 1;              // 0x0178 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bShouldBlockOnLoad : 1;                // 0x0178 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_LevelStreamingBase");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_LevelStreaming
// 0x0018 (0x017C - 0x0194)
class USeqAct_LevelStreaming : public USeqAct_LevelStreamingBase
{
public:
class ULevelStreaming*              Level;                          // 0x0180 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FName                        LevelName;                      // 0x0188 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bStatusIsOk : 1;                // 0x0190 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_LevelStreaming");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_MultiLevelStreaming
// 0x0018 (0x017C - 0x0194)
class USeqAct_MultiLevelStreaming : public USeqAct_LevelStreamingBase
{
public:
TArray<struct FLevelStreamingNameCombo>         Levels;                         // 0x0180
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                       bUnloadAllOtherLevels : 1;          // 0x0190 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bStatusIsOk : 1;                // 0x0190 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_MultiLevelStreaming");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_LevelVisibility
// 0x0014 (0x0178 - 0x018C)
class USeqAct_LevelVisibility : public USeqAct_Latent
{
public:
class ULevelStreaming*                    Level;                              // 0x0178 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                              LevelName;                          // 0x0180 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                             bStatusIsOk : 1;                    // 0x0188 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_LevelVisibility");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_PlaySound
// 0x0028 (0x0178 - 0x01A0)
class USeqAct_PlaySound : public USeqAct_Latent
{
public:
class USoundCue*                          PlaySound;                          // 0x0178 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                     ExtraDelay;                         // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                             bDelayReached : 1;                  // 0x0184 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
```

```
unsigned long                    bSuppressSubtitles : 1;            // 0x0184 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bStopped : 1;                      // 0x0184 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
float                            SoundDuration;                     // 0x0188 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                            FadeInTime;                        // 0x018C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            FadeOutTime;                       // 0x0190 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            VolumeMultiplier;                  // 0x0194 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            PitchMultiplier;                   // 0x0198 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            BeforeEndTime;                     // 0x019C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_PlaySound");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_PrepareMapChange
// 0x001C (0x0178 - 0x0194)
class USeqAct_PrepareMapChange : public USeqAct_Latent
{
public:
struct FName                     MainLevelName;                     // 0x0178 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<struct FName>             InitiallyLoadedSecondaryLevelNames;          // 0x0180
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                    bIsHighPriority : 1;               // 0x0190 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bStatusIsOk : 1;                   // 0x0190 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SeqAct_PrepareMapChange");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetDOFParams
// 0x0058 (0x0178 - 0x01D0)
class USeqAct_SetDOFParams : public USeqAct_Latent
{
public:
float                          FalloffExponent;                     // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BlurKernelSize;                      // 0x017C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MaxNearBlurAmount;                   // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MinBlurAmount;                       // 0x0184 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          MaxFarBlurAmount;                    // 0x0188 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FocusInnerRadius;                    // 0x018C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FocusDistance;                       // 0x0190 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                      FocusPosition;                  // 0x0194 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                          InterpolateSeconds;                  // 0x01A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          InterpolateElapsed;                  // 0x01A4 (0x0004)
[0x0000000000000000]
float                          OldFalloffExponent;                  // 0x01A8 (0x0004)
[0x0000000000000000]
float                          OldBlurKernelSize;                   // 0x01AC (0x0004)
[0x0000000000000000]
float                          OldMaxNearBlurAmount;                // 0x01B0 (0x0004)
[0x0000000000000000]
float                          OldMinBlurAmount;                    // 0x01B4 (0x0004)
[0x0000000000000000]
float                          OldMaxFarBlurAmount;                 // 0x01B8 (0x0004)
[0x0000000000000000]
float                          OldFocusInnerRadius;                 // 0x01BC (0x0004)
[0x0000000000000000]
float                          OldFocusDistance;                    // 0x01C0 (0x0004)
[0x0000000000000000]
struct FVector                      OldFocusPosition;               // 0x01C4 (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetDOFParams");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetMotionBlurParams
// 0x0010 (0x0178 - 0x0188)
class USeqAct_SetMotionBlurParams : public USeqAct_Latent
{
public:
float                                   MotionBlurAmount;                          // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   InterpolateSeconds;                        // 0x017C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   InterpolateElapsed;                        // 0x0180 (0x0004)
[0x0000000000000000]
float                                   OldMotionBlurAmount;                       // 0x0184 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetMotionBlurParams");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_StreamInTextures
// 0x0040 (0x0178 - 0x01B8)
class USeqAct_StreamInTextures : public USeqAct_Latent
{
public:
unsigned long                                   bLocationBased : 1;                        // 0x0178 (0x0004)
[0x0000000020000000] [0x00000001]  CPF_Deprecated)
unsigned long                                   bStreamingActive : 1;                      // 0x0178 (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                                   bHasTriggeredAllLoaded : 1;                // 0x0178 (0x0004)
[0x0000000000000002] [0x00000004] (CPF_Const)
float                                   Seconds;                                   // 0x017C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   StreamingDistanceMultiplier;               // 0x0180 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
int32_t                              NumWantingResourcesID;                    // 0x0184 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                StopTimestamp;                            // 0x0188 (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<class UObject*>                         LocationActors;                          // 0x0190 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UMaterialInterface*>              ForceMaterials;                          // 0x01A0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FTextureGroupContainer                  CinematicTextureGroups;                  // 0x01B0
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                              SelectedCinematicTextureGroups;             // 0x01B4 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_StreamInTextures");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_WaitForLevelsVisible
// 0x0014 (0x0178 - 0x018C)
class USeqAct_WaitForLevelsVisible : public USeqAct_Latent
{
public:
TArray<struct FName>                           LevelNames;                              // 0x0178 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                             bShouldBlockOnLoad : 1;                  // 0x0188 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_WaitForLevelsVisible");
}

return uClassPointer;
};

void eventActivated();
```

```
bool CheckLevelsVisible();
};

// Class Engine.SeqAct_Log
// 0x0028 (0x0160 - 0x0188)
class USeqAct_Log : public USequenceAction
{
public:
unsigned long                    bOutputToScreen : 1;                    // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bIncludeObjComment : 1;                 // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                            TargetDuration;                         // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                   TargetOffset;                           // 0x0168 (0x000C)
[0x0000000000000001] (CPF_Edit)
class FString                    LogMessage;                             // 0x0178 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Log");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_FeatureTest
// 0x002C (0x0188 - 0x01B4)
class USeqAct_FeatureTest : public USeqAct_Log
{
public:
class FString                    FreezeAtParameters;                     // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                            ScreenShotDelay;                        // 0x0198 (0x0004)
[0x0000000000000001] (CPF_Edit)
class FString                    ScreenShotName;                         // 0x01A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                            RemainingScreenShotDelay;               // 0x01B0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_FeatureTest");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ModifyCover
// 0x0018 (0x0160 - 0x0178)
class USeqAct_ModifyCover : public USequenceAction
{
public:
TArray<int32_t>                         Slots;                          // 0x0160 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
uint8_t                            ManualCoverType;                     // 0x0170 (0x0001)
[0x0000000000000001] (CPF_Edit)
unsigned long                      bManualAdjustPlayersOnly : 1;            // 0x0174 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ModifyCover");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ParticleEventGenerator
// 0x0054 (0x0160 - 0x01B4)
class USeqAct_ParticleEventGenerator : public USequenceAction
{
public:
unsigned long                      bEnabled : 1;                       // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                      bUseEmitterLocation : 1;                // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
class AActor*                      Instigator;                         // 0x0168 (0x0008)
[0x0000000000000000]
TArray<class FString>                   EventNames;                         // 0x0170 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                            EventTime;                          // 0x0180 (0x0004)
[0x0000000000000000]
struct FVector                     EventLocation;                       // 0x0184 (0x000C)
[0x0000000000000000]
```

```
struct FVector                              EventDirection;                         // 0x0190 (0x000C)
[0x0000000000000000]
struct FVector                              EventVelocity;                          // 0x019C (0x000C)
[0x0000000000000000]
struct FVector                              EventNormal;                            // 0x01A8 (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ParticleEventGenerator");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_PlayCameraAnim
// 0x0028 (0x0160 - 0x0188)
class USeqAct_PlayCameraAnim : public USequenceAction
{
public:
class UCameraAnim*                               CameraAnim;                            // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                               bLoop : 1;                              // 0x0168 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                               bRandomStartTime : 1;                   // 0x0168 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                                  BlendInTime;                            // 0x016C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  BlendOutTime;                           // 0x0170 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  Rate;                                // 0x0174 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  IntensityScale;                         // 0x0178 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  PlaySpace;                             // 0x017C (0x0001)
[0x0000000000000001] (CPF_Edit)
class AActor*                               UserDefinedSpaceActor;                  // 0x0180 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SeqAct_PlayCameraAnim");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_PlayFaceFXAnim
// 0x0038 (0x0160 - 0x0198)
class USeqAct_PlayFaceFXAnim : public USequenceAction
{
public:
class UFaceFXAnimSet*                FaceFXAnimSetRef;                // 0x0160
(0x0008) [0x0000000000000001] (CPF_Edit)
class FString                        FaceFXGroupName;                 // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                        FaceFXAnimName;                  // 0x0178 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class USoundCue*                     SoundCueToPlay;                  // 0x0188 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UAkEvent*                      AkEventToPlay;                   // 0x0190 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_PlayFaceFXAnim");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_PlayMusicTrack
// 0x0030 (0x0160 - 0x0190)
class USeqAct_PlayMusicTrack : public USequenceAction
{
public:
struct FMusicTrackStruct             MusicTrack;                      // 0x0160 (0x0030)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_PlayMusicTrack");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_Possess
// 0x000C (0x0160 - 0x016C)
class USeqAct_Possess : public USequenceAction
{
public:
class APawn*                         PawnToPossess;                        // 0x0160 (0x0008)
[0x0000000000002000] (CPF_Transient)
unsigned long                        bKillOldPawn : 1;                     // 0x0168 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Possess");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_RangeSwitch
// 0x0010 (0x0160 - 0x0170)
class USeqAct_RangeSwitch : public USequenceAction
{
public:
TArray<struct FSwitchRange>          Ranges;                               // 0x0160 (0x0010)
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_RangeSwitch");
}

return uClassPointer;
};
```

```
};

// Class Engine.SeqAct_SetActiveAnimChild
// 0x0010 (0x0160 - 0x0170)
class USeqAct_SetActiveAnimChild : public USequenceAction
{
public:
struct FName                    NodeName;                    // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                         ChildIndex;                  // 0x0168 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           BlendTime;                   // 0x016C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetActiveAnimChild");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetApexClothingParam
// 0x0004 (0x0160 - 0x0164)
class USeqAct_SetApexClothingParam : public USequenceAction
{
public:
unsigned long                   bEnableApexClothingSimulation : 1;        // 0x0160
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetApexClothingParam");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetBlockRigidBody
// 0x0000 (0x0160 - 0x0160)
class USeqAct_SetBlockRigidBody : public USequenceAction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetBlockRigidBody");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetCameraTarget
// 0x0018 (0x0160 - 0x0178)
class USeqAct_SetCameraTarget : public USequenceAction
{
public:
class AActor*                     CameraTarget;                     // 0x0160 (0x0008)
[0x0000000000002000] (CPF_Transient)
struct FViewTargetTransitionParams          TransitionParams;                    // 0x0168
(0x0010) [0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetCameraTarget");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SetMaterial
// 0x000C (0x0160 - 0x016C)
class USeqAct_SetMaterial : public USequenceAction
{
public:
class UMaterialInterface*                NewMaterial;                     // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                           MaterialIndex;                    // 0x0168 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetMaterial");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetMatInstScalarParam
// 0x0014 (0x0160 - 0x0174)
class USeqAct_SetMatInstScalarParam : public USequenceAction
{
public:
class UMaterialInstanceConstant*                MatInst;                              // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                                    ParamName;                            // 0x0168 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                           ScalarValue;                          // 0x0170 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetMatInstScalarParam");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetMesh
// 0x0018 (0x0160 - 0x0178)
class USeqAct_SetMesh : public USequenceAction
{
public:
class USkeletalMesh*                            NewSkeletalMesh;                      // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UStaticMesh*                              NewStaticMesh;                        // 0x0168 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                                         MeshType;                             // 0x0170 (0x0001)
```

```
[0x0000000000000001] (CPF_Edit)
unsigned long                         bIsAllowedToMove : 1;                     // 0x0174 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bAllowDecalsToReattach : 1;                 // 0x0174 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetMesh");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetPhysics
// 0x0001 (0x0160 - 0x0161)
class USeqAct_SetPhysics : public USequenceAction
{
public:
uint8_t                                    newPhysics;                           // 0x0160 (0x0001)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetPhysics");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetRigidBodyIgnoreVehicles
// 0x0000 (0x0160 - 0x0160)
class USeqAct_SetRigidBodyIgnoreVehicles : public USequenceAction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetRigidBodyIgnoreVehicles");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetSequenceVariable
// 0x0000 (0x0160 - 0x0160)
class USeqAct_SetSequenceVariable : public USequenceAction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetSequenceVariable");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_AccessObjectList
// 0x000C (0x0160 - 0x016C)
class USeqAct_AccessObjectList : public USeqAct_SetSequenceVariable
{
public:
class UObject*                         OutputObject;                          // 0x0160 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
int32_t                          ObjectIndex;                          // 0x0168 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AccessObjectList");
}

return uClassPointer;
};
```

```
};

// Class Engine.SeqAct_AddFloat
// 0x0010 (0x0160 - 0x0170)
class USeqAct_AddFloat : public USeqAct_SetSequenceVariable
{
public:
float                                   ValueA;                                   // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   ValueB;                                   // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FloatResult;                              // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                                 IntResult;                                // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AddFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_AddInt
// 0x0010 (0x0160 - 0x0170)
class USeqAct_AddInt : public USeqAct_SetSequenceVariable
{
public:
int32_t                                 ValueA;                                   // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 ValueB;                                   // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FloatResult;                              // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                                 IntResult;                                // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AddInt");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_CastToFloat
// 0x0008 (0x0160 - 0x0168)
class USeqAct_CastToFloat : public USeqAct_SetSequenceVariable
{
public:
int32_t                          Value;                              // 0x0160 (0x0004)
[0x0000000000000000]
float                            FloatResult;                        // 0x0164 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_CastToFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_CastToInt
// 0x000C (0x0160 - 0x016C)
class USeqAct_CastToInt : public USeqAct_SetSequenceVariable
{
public:
unsigned long                          bTruncate : 1;                        // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                            Value;                              // 0x0164 (0x0004)
[0x0000000000000000]
int32_t                          IntResult;                          // 0x0168 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_CastToInt");
}
```

```
return uClassPointer;
};

};

// Class Engine.SeqAct_DivideFloat
// 0x0010 (0x0160 - 0x0170)
class USeqAct_DivideFloat : public USeqAct_SetSequenceVariable
{
public:
float                                   ValueA;                                  // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   ValueB;                                  // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FloatResult;                             // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                                 IntResult;                               // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_DivideFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_DivideInt
// 0x0010 (0x0160 - 0x0170)
class USeqAct_DivideInt : public USeqAct_SetSequenceVariable
{
public:
int32_t                                 ValueA;                                  // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 ValueB;                                  // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FloatResult;                             // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                                 IntResult;                               // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_DivideInt");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ModifyObjectList
// 0x0004 (0x0160 - 0x0164)
class USeqAct_ModifyObjectList : public USeqAct_SetSequenceVariable
{
public:
int32_t                                    ListEntriesCount;                          // 0x0160 (0x0004)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ModifyObjectList");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_MultiplyFloat
// 0x0010 (0x0160 - 0x0170)
class USeqAct_MultiplyFloat : public USeqAct_SetSequenceVariable
{
public:
float                                      ValueA;                                    // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      ValueB;                                    // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      FloatResult;                               // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                                    IntResult;                                 // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_MultiplyFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_MultiplyInt
// 0x0010 (0x0160 - 0x0170)
class USeqAct_MultiplyInt : public USeqAct_SetSequenceVariable
{
public:
int32_t                                 ValueA;                                 // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 ValueB;                                 // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FloatResult;                            // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                                 IntResult;                              // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_MultiplyInt");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetBool
// 0x0004 (0x0160 - 0x0164)
class USeqAct_SetBool : public USeqAct_SetSequenceVariable
{
public:
unsigned long                           DefaultValue : 1;                       // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetBool");
}
```

```
return uClassPointer;
};

};

// Class Engine.SeqAct_SetFloat
// 0x0018 (0x0160 - 0x0178)
class USeqAct_SetFloat : public USeqAct_SetSequenceVariable
{
public:
float                           Target;                         // 0x0160 (0x0004)
[0x0000000000000000]
TArray<float>                       Value;                          // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetFloat");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SetInt
// 0x0018 (0x0160 - 0x0178)
class USeqAct_SetInt : public USeqAct_SetSequenceVariable
{
public:
int32_t                          Target;                         // 0x0160 (0x0004)
[0x0000000000000000]
TArray<int32_t>                      Value;                          // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetInt");
}

return uClassPointer;
};
```

```
static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SetLocation
// 0x0028 (0x0160 - 0x0188)
class USeqAct_SetLocation : public USeqAct_SetSequenceVariable
{
public:
unsigned long                    bSetLocation : 1;                 // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bSetRotation : 1;                 // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FVector                   LocationValue;                    // 0x0164 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FRotator                  RotationValue;                    // 0x0170 (0x000C)
[0x0000000000000001] (CPF_Edit)
class UObject*                   Target;                       // 0x0180 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetLocation");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SetObject
// 0x0010 (0x0160 - 0x0170)
class USeqAct_SetObject : public USeqAct_SetSequenceVariable
{
public:
class UObject*                   DefaultValue;                     // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UObject*                   Value;                        // 0x0168 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetObject");
}
```

```
return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SetString
// 0x0020 (0x0160 - 0x0180)
class USeqAct_SetString : public USeqAct_SetSequenceVariable
{
public:
class FString                          Target;                          // 0x0160 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                          Value;                           // 0x0170 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetString");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SubtractFloat
// 0x0010 (0x0160 - 0x0170)
class USeqAct_SubtractFloat : public USeqAct_SetSequenceVariable
{
public:
float                             ValueA;                          // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                             ValueB;                          // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                             FloatResult;                       // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                            IntResult;                         // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SubtractFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SubtractInt
// 0x0010 (0x0160 - 0x0170)
class USeqAct_SubtractInt : public USeqAct_SetSequenceVariable
{
public:
int32_t                        ValueA;                    // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                        ValueB;                    // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                         FloatResult;                // 0x0168 (0x0004)
[0x0000000000000000]
int32_t                        IntResult;                 // 0x016C (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SubtractInt");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetVectorComponents
// 0x0018 (0x0160 - 0x0178)
class USeqAct_SetVectorComponents : public USequenceAction
{
public:
struct FVector                    OutVector;                  // 0x0160 (0x000C)
[0x0000000000000000]
float                         X;                        // 0x016C (0x0004)
[0x0000000000000000]
float                         Y;                        // 0x0170 (0x0004)
[0x0000000000000000]
float                         Z;                        // 0x0174 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetVectorComponents");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetWorldAttractorParam
// 0x0090 (0x0160 - 0x01F0)
class USeqAct_SetWorldAttractorParam : public USequenceAction
{
public:
TArray<class AWorldAttractor*>                  Attractor;                              // 0x0160 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                               bEnabledField : 1;                      // 0x0170 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                               bFalloffTypeField : 1;                  // 0x0170 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                               bFalloffExponentField : 1;              // 0x0170 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                               bRangeField : 1;                        // 0x0170 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                               bStrengthField : 1;                     // 0x0170 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                               bEnabled : 1;                           // 0x0170 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
uint8_t                                 FalloffType;                            // 0x0174 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                    FalloffExponent;                        // 0x0178 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    Range;                                  // 0x01A0 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    Strength;                               // 0x01C8 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetWorldAttractorParam");
}

return uClassPointer;
};

};
```

```
// Class Engine.SeqAct_Switch
// 0x0020 (0x0160 - 0x0180)
class USeqAct_Switch : public USequenceAction
{
public:
int32_t                                 LinkCount;                                // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 IncrementAmount;                          // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bLooping : 1;                             // 0x0168 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bAutoDisableLinks : 1;                    // 0x0168 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
TArray<int32_t>                         Indices;                                  // 0x0170 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Switch");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_RandomSwitch
// 0x0010 (0x0180 - 0x0190)
class USeqAct_RandomSwitch : public USeqAct_Switch
{
public:
TArray<int32_t>                         AutoDisabledIndices;                      // 0x0180 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_RandomSwitch");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
```

```
};

// Class Engine.SeqAct_Timer
// 0x0008 (0x0160 - 0x0168)
class USeqAct_Timer : public USequenceAction
{
public:
float                              ActivationTime;                      // 0x0160 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                              Time;                           // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Timer");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_Toggle
// 0x0000 (0x0160 - 0x0160)
class USeqAct_Toggle : public USequenceAction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Toggle");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_Trace
// 0x0040 (0x0160 - 0x01A0)
class USeqAct_Trace : public USequenceAction
{
public:
unsigned long                           bTraceActors : 1;                    // 0x0160 (0x0004)
```

```
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bTraceWorld : 1;                        // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FVector                         TraceExtent;                            // 0x0164 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         StartOffset;                            // 0x0170 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                         EndOffset;                              // 0x017C (0x000C)
[0x0000000000000001] (CPF_Edit)
class UObject*                          HitObject;                             // 0x0188 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
float                             Distance;                              // 0x0190 (0x0004)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FVector                         HitLocation;                            // 0x0194 (0x000C)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Trace");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SequenceCondition
// 0x0000 (0x0140 - 0x0140)
class USequenceCondition : public USequenceOp
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceCondition");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_CompareBool
// 0x0004 (0x0140 - 0x0144)
class USeqCond_CompareBool : public USequenceCondition
{
public:
unsigned long                    bResult : 1;                    // 0x0140 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_CompareBool");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqCond_CompareFloat
// 0x0008 (0x0140 - 0x0148)
class USeqCond_CompareFloat : public USequenceCondition
{
public:
float                            ValueA;                         // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            ValueB;                         // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_CompareFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_CompareInt
// 0x0008 (0x0140 - 0x0148)
class USeqCond_CompareInt : public USequenceCondition
{
public:
int32_t                          ValueA;                         // 0x0140 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
int32_t                              ValueB;                              // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_CompareInt");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_CompareObject
// 0x0000 (0x0140 - 0x0140)
class USeqCond_CompareObject : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_CompareObject");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_GetServerType
// 0x0000 (0x0140 - 0x0140)
class USeqCond_GetServerType : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_GetServerType");
```

```
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqCond_Increment
// 0x000C (0x0140 - 0x014C)
class USeqCond_Increment : public USequenceCondition
{
public:
int32_t                              IncrementAmount;                          // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                              ValueA;                          // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                              ValueB;                          // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_Increment");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_IncrementFloat
// 0x000C (0x0140 - 0x014C)
class USeqCond_IncrementFloat : public USequenceCondition
{
public:
float                              IncrementAmount;                          // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ValueA;                          // 0x0144 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              ValueB;                          // 0x0148 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IncrementFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_IsAlive
// 0x0000 (0x0140 - 0x0140)
class USeqCond_IsAlive : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IsAlive");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_IsBenchmarking
// 0x0000 (0x0140 - 0x0140)
class USeqCond_IsBenchmarking : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IsBenchmarking");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqCond_IsConsole
// 0x0000 (0x0140 - 0x0140)
class USeqCond_IsConsole : public USequenceCondition
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IsConsole");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_IsInCombat
// 0x0000 (0x0140 - 0x0140)
class USeqCond_IsInCombat : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IsInCombat");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_IsLoggedIn
// 0x0004 (0x0140 - 0x0144)
class USeqCond_IsLoggedIn : public USequenceCondition
{
public:
int32_t                                NumNeededLoggedIn;                       // 0x0140 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IsLoggedIn");
}

return uClassPointer;
};

bool eventCheckLogins();
};

// Class Engine.SeqCond_IsPIE
// 0x0000 (0x0140 - 0x0140)
class USeqCond_IsPIE : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IsPIE");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqCond_IsSameTeam
// 0x0000 (0x0140 - 0x0140)
class USeqCond_IsSameTeam : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_IsSameTeam");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_MatureLanguage
// 0x0000 (0x0140 - 0x0140)
```

```
class USeqCond_MatureLanguage : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_MatureLanguage");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_ShowGore
// 0x0000 (0x0140 - 0x0140)
class USeqCond_ShowGore : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_ShowGore");
}

return uClassPointer;
};

};

// Class Engine.SeqCond_SwitchBase
// 0x0000 (0x0140 - 0x0140)
class USeqCond_SwitchBase : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_SwitchBase");
```

```
}

return uClassPointer;
};

void eventRemoveValueEntry(int32_t RemoveIndex);
void eventInsertValueEntry(int32_t InsertIndex);
bool eventIsFallThruEnabled(int32_t ValueIndex);
void eventVerifyDefaultCaseValue();
};

// Class Engine.SeqCond_SwitchClass
// 0x0010 (0x0140 - 0x0150)
class USeqCond_SwitchClass : public USeqCond_SwitchBase
{
public:
TArray<struct FSwitchClassInfo>                  ClassArray;                              // 0x0140 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_SwitchClass");
}

return uClassPointer;
};

void eventRemoveValueEntry(int32_t RemoveIndex);
void eventInsertValueEntry(int32_t InsertIndex);
bool eventIsFallThruEnabled(int32_t ValueIndex);
void eventVerifyDefaultCaseValue();
};

// Class Engine.SeqCond_SwitchObject
// 0x0010 (0x0140 - 0x0150)
class USeqCond_SwitchObject : public USeqCond_SwitchBase
{
public:
TArray<struct FSwitchObjectCase>                 SupportedValues;                         // 0x0140 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_SwitchObject");
```

```
}

return uClassPointer;
};

void eventRemoveValueEntry(int32_t RemoveIndex);
void eventInsertValueEntry(int32_t InsertIndex);
bool eventIsFallThruEnabled(int32_t ValueIndex);
void eventVerifyDefaultCaseValue();
};

// Class Engine.SeqCond_SwitchPlatform
// 0x0000 (0x0140 - 0x0140)
class USeqCond_SwitchPlatform : public USequenceCondition
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqCond_SwitchPlatform");
}

return uClassPointer;
};

};

// Class Engine.SequenceEvent
// 0x003C (0x0140 - 0x017C)
class USequenceEvent : public USequenceOp
{
public:
TArray<class USequenceEvent*>                  DuplicateEvts;                          // 0x0140 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class AActor*                                   Originator;                             // 0x0150 (0x0008)
[0x0000000000000000]
class AActor*                                   Instigator;                             // 0x0158 (0x0008)
[0x0000000000000000]
float                                           ActivationTime;                         // 0x0160 (0x0004)
[0x0000000000000000]
int32_t                                         TriggerCount;                           // 0x0164 (0x0004)
[0x0000000000000000]
int32_t                                         MaxTriggerCount;                        // 0x0168 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                           ReTriggerDelay;                         // 0x016C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                                   bEnabled : 1;                           // 0x0170 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                   bPlayerOnly : 1;                        // 0x0170 (0x0004)
```

```
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bRegistered : 1;                    // 0x0170 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                    bClientSideOnly : 1;                  // 0x0170 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
uint8_t                          Priority;                           // 0x0174 (0x0001)
[0x0000000000000001] (CPF_Edit)
int32_t                          MaxWidth;                             // 0x0178 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceEvent");
}

return uClassPointer;
};

void eventToggled();
void Reset();
bool CheckActivate(class AActor* InOriginator, class AActor* InInstigator, unsigned long bTest,
unsigned long bPushTop, TArray<int32_t>& ActivateIndices);
void eventRegisterEvent();
};

// Class Engine.SeqEvent_AISeeEnemy
// 0x0008 (0x017C - 0x0184)
class USeqEvent_AISeeEnemy : public USequenceEvent
{
public:
float                            MaxSightDistance;                     // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_AISeeEnemy");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_AnalogInput
// 0x001C (0x017C - 0x0198)
class USeqEvent_AnalogInput : public USequenceEvent
{
public:
unsigned long                                bTrapInput : 1;                          // 0x0180 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                                  AllowedPlayerIndex;                          // 0x0184 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FName>                          InputNames;                                 // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_AnalogInput");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_AnimNotify
// 0x000C (0x017C - 0x0188)
class USeqEvent_AnimNotify : public USequenceEvent
{
public:
struct FName                                 NotifyName;                              // 0x0180 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_AnimNotify");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_Console
// 0x001C (0x017C - 0x0198)
class USeqEvent_Console : public USequenceEvent
{
public:
```

```
struct FName                                    ConsoleEventName;                            // 0x0180 (0x0008)
[0x0000000000000001] (CPF_Edit)
class FString                                   EventDesc;                                   // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_Console");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_ConstraintBroken
// 0x0004 (0x017C - 0x0180)
class USeqEvent_ConstraintBroken : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_ConstraintBroken");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_Destroyed
// 0x0004 (0x017C - 0x0180)
class USeqEvent_Destroyed : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_Destroyed");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_Input
// 0x001C (0x017C - 0x0198)
class USeqEvent_Input : public USequenceEvent
{
public:
unsigned long                    bTrapInput : 1;                 // 0x0180 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                        AllowedPlayerIndex;              // 0x0184 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FName>                 InputNames;                      // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_Input");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_LevelBeginning
// 0x0004 (0x017C - 0x0180)
class USeqEvent_LevelBeginning : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_LevelBeginning");
}

return uClassPointer;
};
```

```
};

// Class Engine.SeqEvent_LevelLoaded
// 0x0004 (0x017C - 0x0180)
class USeqEvent_LevelLoaded : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_LevelLoaded");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqEvent_LevelStartup
// 0x0004 (0x017C - 0x0180)
class USeqEvent_LevelStartup : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_LevelStartup");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_Mover
// 0x0008 (0x017C - 0x0184)
class USeqEvent_Mover : public USequenceEvent
{
public:
float                                   StayOpenTime;                             // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_Mover");
}

return uClassPointer;
};

void NotifyFinishedOpen();
void NotifyDetached(class AActor* Other);
void NotifyAttached(class AActor* Other);
void NotifyEncroachingOn(class AActor* Hit);
void eventRegisterEvent();
};

// Class Engine.SeqEvent_ParticleEvent
// 0x0038 (0x017C - 0x01B4)
class USeqEvent_ParticleEvent : public USequenceEvent
{
public:
uint8_t                                  EventType;                              // 0x0180 (0x0001)
[0x0000000000000000]
struct FVector                           EventPosition;                          // 0x0184 (0x000C)
[0x0000000000000000]
float                                    EventEmitterTime;                       // 0x0190 (0x0004)
[0x0000000000000000]
struct FVector                           EventVelocity;                          // 0x0194 (0x000C)
[0x0000000000000000]
float                                    EventParticleTime;                      // 0x01A0 (0x0004)
[0x0000000000000000]
struct FVector                           EventNormal;                            // 0x01A4 (0x000C)
[0x0000000000000000]
unsigned long                            UseRelfectedImpactVector : 1;           // 0x01B0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_ParticleEvent");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};
```

```
// Class Engine.SeqEvent_RemoteEvent
// 0x0010 (0x017C - 0x018C)
class USeqEvent_RemoteEvent : public USequenceEvent
{
public:
struct FName                          EventName;                          // 0x0180 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                         bStatusIsOk : 1;                    // 0x0188 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_RemoteEvent");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqEvent_RigidBodyCollision
// 0x0008 (0x017C - 0x0184)
class USeqEvent_RigidBodyCollision : public USequenceEvent
{
public:
float                                 MinCollisionVelocity;               // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_RigidBodyCollision");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_SeeDeath
// 0x0004 (0x017C - 0x0180)
class USeqEvent_SeeDeath : public USequenceEvent
{
```

```
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_SeeDeath");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_SequenceActivated
// 0x0014 (0x017C - 0x0190)
class USeqEvent_SequenceActivated : public USequenceEvent
{
public:
class FString                          InputLabel;                          // 0x0180 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_SequenceActivated");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_Touch
// 0x003C (0x017C - 0x01B8)
class USeqEvent_Touch : public USequenceEvent
{
public:
TArray<class UClass*>                  ClassProximityTypes;                 // 0x0180 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UClass*>                  IgnoredClassProximityTypes;              // 0x0190
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                          bForceOverlapping : 1;               // 0x01A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<class AActor*>                  TouchedList;                         // 0x01A8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_Touch");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
void eventToggled();
bool CheckUnTouchActivate(class AActor* InOriginator, class AActor* InInstigator, unsigned long
bTest);
bool CheckTouchActivate(class AActor* InOriginator, class AActor* InInstigator, unsigned long
bTest);
};

// Class Engine.SeqEvent_TouchInput
// 0x0014 (0x017C - 0x0190)
class USeqEvent_TouchInput : public USequenceEvent
{
public:
unsigned long                    bTrapInput : 1;                 // 0x0180 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                      AllowedPlayerIndex;             // 0x0184 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                      AllowedTouchIndex;              // 0x0188 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                      AllowedTouchpadIndex;             // 0x018C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_TouchInput");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_Used
// 0x0044 (0x017C - 0x01C0)
class USeqEvent_Used : public USequenceEvent
{
```

```
public:
unsigned long                          bAimToInteract : 1;                      // 0x0180 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                  InteractDistance;                        // 0x0184 (0x0004)
[0x0000000000000001] (CPF_Edit)
class FString                          InteractText;                            // 0x0188 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class UTexture2D*                      InteractIcon;                            // 0x0198 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<class UClass*>                  ClassProximityTypes;                     // 0x01A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UClass*>                  IgnoredClassProximityTypes;              // 0x01B0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_Used");
}

return uClassPointer;
};

};

// Class Engine.SequenceVariable
// 0x0008 (0x00D8 - 0x00E0)
class USequenceVariable : public USequenceObject
{
public:
struct FName                           VarName;                                 // 0x00D8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SequenceVariable");
}

return uClassPointer;
};

};

// Class Engine.InterpData
// 0x0070 (0x00E0 - 0x0150)
```

```
class UInterpData : public USequenceVariable
{
public:
float                          InterpLength;                    // 0x00E0 (0x0004)
[0x0000000000000000]
float                          PathBuildTime;                   // 0x00E4 (0x0004)
[0x0000000000000000]
TArray<class UInterpGroup*>             InterpGroups;                    // 0x00E8 (0x0010)
[0x0000000000400008] (CPF_ExportObject | CPF_NeedCtorLink)
class UInterpCurveEdSetup*              CurveEdSetup;                    // 0x00F8 (0x0008)
[0x0000000000000008] (CPF_ExportObject)
TArray<class UInterpFilter*>            InterpFilters;                   // 0x0100 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
class UInterpFilter*                SelectedFilter;                  // 0x0110 (0x0008)
[0x0000000800000000]
TArray<class UInterpFilter*>            DefaultFilters;                  // 0x0118 (0x0010)
[0x0000000800402000] (CPF_Transient | CPF_NeedCtorLink)
float                          EdSectionStart;                  // 0x0128 (0x0004)
[0x0000000000000000]
float                          EdSectionEnd;                    // 0x012C (0x0004)
[0x0000000000000000]
unsigned long                      bShouldBakeAndPrune : 1;           // 0x0130 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                      bHasSetDefaultTangentWeights : 1;      // 0x0130
(0x0004) [0x0000000000000000] [0x00000002]
TArray<struct FAnimSetBakeAndPruneStatus>       BakeAndPruneStatus;              //
0x0138 (0x0010) [0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
class UInterpGroupDirector*             CachedDirectorGroup;             // 0x0148
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpData");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Bool
// 0x0004 (0x00E0 - 0x00E4)
class USeqVar_Bool : public USequenceVariable
{
public:
int32_t                         bValue;                         // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Bool");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_External
// 0x0018 (0x00E0 - 0x00F8)
class USeqVar_External : public USequenceVariable
{
public:
class UClass*                          ExpectedType;                    // 0x00E0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class FString                          VariableLabel;                   // 0x00E8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_External");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Float
// 0x0004 (0x00E0 - 0x00E4)
class USeqVar_Float : public USequenceVariable
{
public:
float                                  FloatValue;                      // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Float");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_RandomFloat
// 0x000C (0x00E4 - 0x00F0)
class USeqVar_RandomFloat : public USeqVar_Float
{
public:
float                                   Min;                                    // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   Max;                                    // 0x00EC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_RandomFloat");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Int
// 0x0004 (0x00E0 - 0x00E4)
class USeqVar_Int : public USequenceVariable
{
public:
int32_t                                 IntValue;                               // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Int");
}

return uClassPointer;
};
```

```
};

// Class Engine.SeqVar_RandomInt
// 0x000C (0x00E4 - 0x00F0)
class USeqVar_RandomInt : public USeqVar_Int
{
public:
int32_t                                 Min;                                    // 0x00E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                 Max;                                    // 0x00EC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_RandomInt");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Named
// 0x0014 (0x00E0 - 0x00F4)
class USeqVar_Named : public USequenceVariable
{
public:
class UClass*                           ExpectedType;                           // 0x00E0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                            FindVarName;                            // 0x00E8 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bStatusIsOk : 1;                        // 0x00F0 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Named");
}

return uClassPointer;
};

};
```

```
// Class Engine.SeqVar_Object
// 0x0028 (0x00E0 - 0x0108)
class USeqVar_Object : public USequenceVariable
{
public:
class UObject*                         ObjValue;                         // 0x00E0 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                         ActorLocation;                    // 0x00E8 (0x000C)
[0x0000000000002000] (CPF_Transient)
TArray<class UClass*>                  SupportedClasses;                 // 0x00F8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Object");
}

return uClassPointer;
};

void SetObjectValue(class UObject* NewValue);
class UObject* GetObjectValueW();
};

// Class Engine.SeqVar_Character
// 0x0008 (0x0108 - 0x0110)
class USeqVar_Character : public USeqVar_Object
{
public:
class UClass*                          PawnClass;                        // 0x0108 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Character");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Group
// 0x0020 (0x0108 - 0x0128)
```

```
class USeqVar_Group : public USeqVar_Object
{
public:
struct FName                              GroupName;                              // 0x0108 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                             bCachedList : 1;                        // 0x0110 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
TArray<class UObject*>                         Actors;                            // 0x0118 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Group");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_ObjectList
// 0x0010 (0x0108 - 0x0118)
class USeqVar_ObjectList : public USeqVar_Object
{
public:
TArray<class UObject*>                         ObjList;                           // 0x0108 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_ObjectList");
}

return uClassPointer;
};

void SetObjectValue(class UObject* NewValue);
class UObject* GetObjectValueW();
};

// Class Engine.SeqVar_ObjectVolume
// 0x002C (0x0108 - 0x0134)
class USeqVar_ObjectVolume : public USeqVar_Object
{
```

```
public:
float                              LastUpdateTime;                      // 0x0108 (0x0004)
[0x0000000000000000]
TArray<class UObject*>                       ContainedObjects;                       // 0x0110 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class UClass*>                        ExcludeClassList;                      // 0x0120 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                          bCollidingOnly : 1;                     // 0x0130 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_ObjectVolume");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Player
// 0x0018 (0x0108 - 0x0120)
class USeqVar_Player : public USeqVar_Object
{
public:
TArray<class UObject*>                       Players;                          // 0x0108 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
unsigned long                          bAllPlayers : 1;                     // 0x0118 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                             PlayerIdx;                        // 0x011C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Player");
}

return uClassPointer;
};

class UObject* GetObjectValueW();
void UpdatePlayersList();
};
```

```
// Class Engine.SeqVar_String
// 0x0010 (0x00E0 - 0x00F0)
class USeqVar_String : public USequenceVariable
{
public:
class FString                              StrValue;                                  // 0x00E0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_String");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Vector
// 0x000C (0x00E0 - 0x00EC)
class USeqVar_Vector : public USequenceVariable
{
public:
struct FVector                              VectValue;                                  // 0x00E0 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Vector");
}

return uClassPointer;
};

};

// Class Engine.AmbientSound
// 0x0010 (0x0270 - 0x0280)
class AAmbientSound : public AKeypoint
{
public:
unsigned long                               bAutoPlay : 1;                              // 0x0270 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                               bIsPlaying : 1;                             // 0x0270 (0x0004)
```

```
[0x0000000000000000] [0x00000002]
class UAudioComponent*                    AudioComponent;                     // 0x0278
(0x0008) [0x000000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSound");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundMovable
// 0x0000 (0x0280 - 0x0280)
class AAmbientSoundMovable : public AAmbientSound
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundMovable");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundSimple
// 0x0018 (0x0280 - 0x0298)
class AAmbientSoundSimple : public AAmbientSound
{
public:
class USoundNodeAmbient*                  AmbientProperties;                  // 0x0280
(0x0008) [0x0000000004020001] (CPF_Edit | CPF_EditConst | CPF_EditInline)
class USoundCue*                          SoundCueInstance;                   // 0x0288 (0x0008)
[0x000000000440000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
class USoundNodeAmbient*                  SoundNodeInstance;                  // 0x0290
(0x0008) [0x000000000440000A] (CPF_Const | CPF_ExportObject | CPF_NeedCtorLink |
CPF_EditInline)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundSimple");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundNonLoop
// 0x0000 (0x0298 - 0x0298)
class AAmbientSoundNonLoop : public AAmbientSoundSimple
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundNonLoop");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundSimpleToggleable
// 0x0014 (0x0298 - 0x02AC)
class AAmbientSoundSimpleToggleable : public AAmbientSoundSimple
{
public:
unsigned long                    bCurrentlyPlaying : 1;             // 0x0298 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)
unsigned long                    bFadeOnToggle : 1;                 // 0x0298 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bIgnoreAutoPlay : 1;               // 0x0298 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
float                            FadeInDuration;                    // 0x029C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            FadeInVolumeLevel;                 // 0x02A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            FadeOutDuration;                   // 0x02A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            FadeOutVolumeLevel;                // 0x02A8 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundSimpleToggleable");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct AAmbientSoundSimpleToggleable_FCheckpointRecord&
Record);
void CreateCheckpointRecord(struct AAmbientSoundSimpleToggleable_FCheckpointRecord&
Record);
void OnToggle(class USeqAct_Toggle* Action);
void StopPlaying();
void StartPlaying();
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.AmbientSoundNonLoopingToggleable
// 0x0004 (0x02AC - 0x02B0)
class AAmbientSoundNonLoopingToggleable : public AAmbientSoundSimpleToggleable
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundNonLoopingToggleable");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundSpline
// 0x001C (0x0280 - 0x029C)
class AAmbientSoundSpline : public AAmbientSound
{
public:
float                                DistanceBetweenPoints;                    // 0x0280 (0x0004)
[0x0000000800000001] (CPF_Edit)
```

```
class USplineComponent*                    SplineComponent;                    // 0x0288 (0x0008)
[0x0000000804080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FVector                     TestPoint;                          // 0x0290 (0x000C)
[0x0000000800000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundSpline");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundSimpleSpline
// 0x0008 (0x029C - 0x02A4)
class AAmbientSoundSimpleSpline : public AAmbientSoundSpline
{
public:
int32_t                            EditedSlot;                         // 0x02A0 (0x0004)
[0x0000000800000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundSimpleSpline");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundSplineMultiCue
// 0x0008 (0x029C - 0x02A4)
class AAmbientSoundSplineMultiCue : public AAmbientSoundSpline
{
public:
int32_t                            EditedSlot;                         // 0x02A0 (0x0004)
[0x0000000800000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundSplineMultiCue");
}

return uClassPointer;
};

};

// Class Engine.DistributionFloatSoundParameter
// 0x0007 (0x00A1 - 0x00A8)
class UDistributionFloatSoundParameter : public UDistributionFloatParameterBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DistributionFloatSoundParameter");
}

return uClassPointer;
};

};

// Class Engine.SoundNode
// 0x0018 (0x0060 - 0x0078)
class USoundNode : public UObject
{
public:
int32_t                                   NodeUpdateHint;                          // 0x0060 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<class USoundNode*>                 ChildNodes;                              // 0x0068 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNode");
}

return uClassPointer;
```

```
};

};

// Class Engine.ForcedLoopSoundNode
// 0x0000 (0x0078 - 0x0078)
class UForcedLoopSoundNode : public USoundNode
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ForcedLoopSoundNode");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeAmbient
// 0x0040 (0x0078 - 0x00B8)
class USoundNodeAmbient : public USoundNode
{
public:
unsigned long                           bAttenuate : 1;                         // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bSpatialize : 1;                        // 0x0078 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                           bAttenuateWithLPF : 1;                  // 0x0078 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
float                           dBAttenuationAtMax;                     // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                          DistanceModel;                         // 0x0080 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                           RadiusMin;                              // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           RadiusMax;                              // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           LPFRadiusMin;                           // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           LPFRadiusMax;                           // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           PitchMin;                               // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           PitchMax;                               // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           VolumeMin;                              // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                                   VolumeMax;                              // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FAmbientSoundSlot>                SoundSlots;                             // 0x00A8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeAmbient");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeAmbientNonLoop
// 0x0030 (0x00B8 - 0x00E8)
class USoundNodeAmbientNonLoop : public USoundNodeAmbient
{
public:
float                                   DelayMin;                               // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   DelayMax;                               // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                    DelayTime;                              // 0x00C0 (0x0028)
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeAmbientNonLoop");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeAmbientNonLoopToggle
// 0x0000 (0x00E8 - 0x00E8)
class USoundNodeAmbientNonLoopToggle : public USoundNodeAmbientNonLoop
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeAmbientNonLoopToggle");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeAttenuation
// 0x0020 (0x0078 - 0x0098)
class USoundNodeAttenuation : public USoundNode
{
public:
unsigned long                          bAttenuate : 1;                          // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bSpatialize : 1;                         // 0x0078 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          bAttenuateWithLPF : 1;                   // 0x0078 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
float                                  dBAttenuationAtMax;                      // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  OmniRadius;                              // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                DistanceAlgorithm;                       // 0x0084 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                                DistanceType;                            // 0x0085 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                  RadiusMin;                               // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  RadiusMax;                               // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  LPFRadiusMin;                            // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  LPFRadiusMax;                            // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeAttenuation");
}

return uClassPointer;
};
```

```
};

// Class Engine.SoundNodeAttenuationAndGain
// 0x0030 (0x0078 - 0x00A8)
class USoundNodeAttenuationAndGain : public USoundNode
{
public:
unsigned long                    bAttenuate : 1;                  // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bSpatialize : 1;                 // 0x0078 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bAttenuateWithLPF : 1;               // 0x0078 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
float                            dBAttenuationAtMax;              // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            OmniRadius;                      // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                          GainDistanceAlgorithm;           // 0x0084 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          AttenuateDistanceAlgorithm;          // 0x0085 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          DistanceType;                    // 0x0086 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                            MinimalVolume;                   // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RadiusMin;                       // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RadiusPeak;                      // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RadiusMax;                       // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            LPFMinimal;                      // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            LPFRadiusMin;                    // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            LPFRadiusPeak;                   // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            LPFRadiusMax;                    // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeAttenuationAndGain");
}

return uClassPointer;
};
```

```
};

// Class Engine.SoundNodeConcatenator
// 0x0010 (0x0078 - 0x0088)
class USoundNodeConcatenator : public USoundNode
{
public:
TArray<float>                    InputVolume;                    // 0x0078 (0x0010)
[0x0000000000400049] (CPF_Edit | CPF_ExportObject | CPF_EditConstArray |
CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeConcatenator");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeConcatenatorRadio
// 0x0000 (0x0078 - 0x0078)
class USoundNodeConcatenatorRadio : public USoundNode
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeConcatenatorRadio");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeDelay
// 0x0030 (0x0078 - 0x00A8)
class USoundNodeDelay : public USoundNode
{
public:
float                            DelayMin;                       // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
float                                  DelayMax;                              // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                  DelayDuration;                          // 0x0080 (0x0028)
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeDelay");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeDistanceCrossFade
// 0x0010 (0x0078 - 0x0088)
class USoundNodeDistanceCrossFade : public USoundNode
{
public:
TArray<struct FDistanceDatum>                  CrossFadeInput;                         // 0x0078
(0x0010) [0x0000000000480049] (CPF_Edit | CPF_ExportObject | CPF_EditConstArray |
CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeDistanceCrossFade");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeDoppler
// 0x0004 (0x0078 - 0x007C)
class USoundNodeDoppler : public USoundNode
{
public:
float                                  DopplerIntensity;                        // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeDoppler");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeEnveloper
// 0x0028 (0x0078 - 0x00A0)
class USoundNodeEnveloper : public USoundNode
{
public:
float                              LoopStart;                         // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              LoopEnd;                           // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              DurationAfterLoop;                     // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             LoopCount;                         // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bLoopIndefinitely : 1;                   // 0x0088 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bLoop : 1;                         // 0x0088 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
class UDistributionFloatConstantCurve*          VolumeInterpCurve;                      // 0x0090
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class UDistributionFloatConstantCurve*          PitchInterpCurve;                      // 0x0098
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeEnveloper");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeLooping
// 0x0038 (0x0078 - 0x00B0)
```

```
class USoundNodeLooping : public USoundNode
{
public:
unsigned long                    bLoopIndefinitely : 1;            // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                        LoopCountMin;                  // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        LoopCountMax;                  // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat            LoopCount;                    // 0x0088 (0x0028)
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeLooping");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeMature
// 0x0000 (0x0078 - 0x0078)
class USoundNodeMature : public USoundNode
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeMature");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeMixer
// 0x0010 (0x0078 - 0x0088)
class USoundNodeMixer : public USoundNode
{
public:
TArray<float>                      InputVolume;                   // 0x0078 (0x0010)
[0x0000000000400049] (CPF_Edit | CPF_ExportObject | CPF_EditConstArray |
CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeMixer");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeModulator
// 0x0060 (0x0078 - 0x00D8)
class USoundNodeModulator : public USoundNode
{
public:
float                                  PitchMin;                             // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  PitchMax;                             // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  VolumeMin;                            // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  VolumeMax;                            // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                   PitchModulation;                        // 0x0088 (0x0028)
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)
struct FRawDistributionFloat                   VolumeModulation;                       // 0x00B0
(0x0028) [0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeModulator");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeModulatorContinuous
// 0x0050 (0x0078 - 0x00C8)
class USoundNodeModulatorContinuous : public USoundNode
```

```
{
public:
struct FRawDistributionFloat                    PitchModulation;                        // 0x0078 (0x0028)
[0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FRawDistributionFloat                    VolumeModulation;                       // 0x00A0
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeModulatorContinuous");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeOscillator
// 0x00C8 (0x0078 - 0x0140)
class USoundNodeOscillator : public USoundNode
{
public:
unsigned long                           bModulateVolume : 1;                    // 0x0078 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bModulatePitch : 1;                     // 0x0078 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                                   AmplitudeMin;                           // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   AmplitudeMax;                           // 0x0080 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FrequencyMin;                           // 0x0084 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FrequencyMax;                           // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   OffsetMin;                              // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   OffsetMax;                              // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   CenterMin;                              // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   CenterMax;                              // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                    Amplitude;                              // 0x00A0 (0x0028)
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)
struct FRawDistributionFloat                    Frequency;                              // 0x00C8 (0x0028)
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)
struct FRawDistributionFloat                    Offset;                                 // 0x00F0 (0x0028)
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)
struct FRawDistributionFloat                    Center;                                 // 0x0118 (0x0028)
```

```
[0x0000000020480000] (CPF_Component | CPF_NeedCtorLink | CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeOscillator");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeRandom
// 0x002C (0x0078 - 0x00A4)
class USoundNodeRandom : public USoundNode
{
public:
TArray<float>                    Weights;                         // 0x0078 (0x0010)
[0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
int32_t                          PreselectAtLevelLoad;            // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                    bRandomizeWithoutReplacement : 1;          // 0x008C
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<unsigned long>            HasBeenUsed;                     // 0x0090 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                          NumRandomUsed;                   // 0x00A0 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeRandom");
}

return uClassPointer;
};

};

// Class Engine.SoundNodeWave
// 0x0440 (0x0078 - 0x04B8)
class USoundNodeWave : public USoundNode
{
public:
int32_t                          CompressionQuality;              // 0x0078 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
unsigned long                    bForceRealTimeDecompression : 1;          // 0x007C
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bLoopingSound : 1;                    // 0x007C (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bDynamicResource : 1;                 // 0x007C (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                    bUseTTS : 1;                          // 0x007C (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bProcedural : 1;                      // 0x007C (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                    bMature : 1;                          // 0x007C (0x0004)
[0x0000000000008003] [0x00000020] (CPF_Edit | CPF_Const | CPF_Localized)
unsigned long                    bManualWordWrap : 1;                  // 0x007C (0x0004)
[0x0000000000008003] [0x00000040] (CPF_Edit | CPF_Const | CPF_Localized)
unsigned long                    bSingleLine : 1;                      // 0x007C (0x0004)
[0x0000000000008003] [0x00000080] (CPF_Edit | CPF_Const | CPF_Localized)
uint8_t                          TTSSpeaker;                           // 0x0080 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          DecompressionType;                    // 0x0081 (0x0001)
[0x0000000000002002] (CPF_Const | CPF_Transient)
uint8_t                          MobileDetailMode;                     // 0x0082 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class FString                    SpokenText;                           // 0x0088 (0x0010)
[0x0000000000408003] (CPF_Edit | CPF_Const | CPF_Localized | CPF_NeedCtorLink)
float                            Volume;                               // 0x0098 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
float                            Pitch;                                // 0x009C (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
float                            Duration;                             // 0x00A0 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                          NumChannels;                          // 0x00A4 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                          SampleRate;                           // 0x00A8 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
TArray<int32_t>                  ChannelOffsets;                       // 0x00B0 (0x0010)
[0x0000000800400002] (CPF_Const | CPF_NeedCtorLink)
TArray<int32_t>                  ChannelSizes;                         // 0x00C0 (0x0010)
[0x0000000800400002] (CPF_Const | CPF_NeedCtorLink)
struct FUntypedBulkData_Mirror          RawData;                       // 0x00D0 (0x0058)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                  VorbisDecompressor;                   // 0x0128 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                  RawPCMData;                           // 0x0130 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                          RawPCMDataSize;                       // 0x0138 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FUntypedBulkData_Mirror          CompressedPCData;              // 0x0140
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror          CompressedXbox360Data;         // 0x0198
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror          CompressedDingoData;           // 0x01F0
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror          CompressedPS3Data;             // 0x0248
```

```
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror                    CompressedWiiUData;                        // 0x02A0
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror                    CompressedIPhoneData;                      // 0x02F8
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror                    CompressedFlashData;                       // 0x0350
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror                    CompressedPS4Data;                         // 0x03A8
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FUntypedBulkData_Mirror                    CompressedNNXData;                         // 0x0400
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                                           ResourceID;                        // 0x0458 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                                           ResourceSize;                      // 0x045C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FPointer                                   ResourceData;                      // 0x0460 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FSubtitleCue>                       Subtitles;                         // 0x0468 (0x0010)
[0x0000000000408003] (CPF_Edit | CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                                     Comment;                           // 0x0478 (0x0010)
[0x0000000800408003] (CPF_Edit | CPF_Const | CPF_Localized | CPF_NeedCtorLink)
TArray<struct FLocalizedSubtitle>                 LocalizedSubtitles;                        // 0x0488
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class FString                                     SourceFilePath;                    // 0x0498 (0x0010)
[0x0000000800420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)
class FString                                     SourceFileTimestamp;               // 0x04A8 (0x0010)
[0x0000000800420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeWave");
}

return uClassPointer;
};

void eventGeneratePCMData(int32_t SamplesNeeded, TArray<uint8_t>& Buffer);
};

// Class Engine.SoundNodeWaveStreaming
// 0x0018 (0x04B8 - 0x04D0)
class USoundNodeWaveStreaming : public USoundNodeWave
{
public:
TArray<uint8_t>                                   QueuedAudio;                       // 0x04B8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                                             InactiveDuration;                  // 0x04C8 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                           MaxGenerateSamples;                // 0x04CC (0x0004)
```

```
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeWaveStreaming");
}

return uClassPointer;
};

void eventGeneratePCMData(int32_t SamplesNeeded, TArray<uint8_t>& Buffer);
int32_t eventAvailableAudioBytes();
void eventResetAudio();
void eventQueueSilence(float Seconds);
void eventQueueAudio(TArray<uint8_t> Data);
};

// Class Engine.SoundNodeWaveParam
// 0x0008 (0x0078 - 0x0080)
class USoundNodeWaveParam : public USoundNode
{
public:
struct FName                          WaveParameterName;                    // 0x0078 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundNodeWaveParam");
}

return uClassPointer;
};

};

// Class Engine.LandscapeProxy
// 0x014C (0x0268 - 0x03B4)
class ALandscapeProxy : public AInfo
{
public:
struct FGuid                          LandscapeGuid;                        // 0x0268 (0x0010)
[0x0000000000000002] (CPF_Const)
int32_t                               MaxLODLevel;                          // 0x0278 (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
class UPhysicalMaterial*                    DefaultPhysMaterial;                      // 0x0280 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                StreamingDistanceMultiplier;                // 0x0288 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UMaterialInterface*                   LandscapeMaterial;                        // 0x0290 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                LODDistanceFactor;                        // 0x0298 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class ULandscapeComponent*>              LandscapeComponents;                        //
0x02A0 (0x0010) [0x000000000448000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
TArray<class ULandscapeHeightfieldCollisionComponent*>
CollisionComponents;                      // 0x02B0 (0x0010) [0x000000000448000A] (CPF_Const |
CPF_ExportObject | CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
uint8_t                              UnknownData00[0x50];                        // 0x02C0 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeProxy.MaterialInstanceConstantMap
uint8_t                              UnknownData01[0x50];                        // 0x0310 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeProxy.WeightmapUsageMap
float                                StaticLightingResolution;                  // 0x0360 (0x0004)
[0x0000000000000001] (CPF_Edit)
class ALandscape*                          LandscapeActor;                          // 0x0368 (0x0008)
[0x0000000000002001] (CPF_Edit | CPF_Transient)
unsigned long                             bIsProxy : 1;                       // 0x0370 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                             bIsSetup : 1;                       // 0x0370 (0x0004)
[0x0000000800002000] [0x00000002] (CPF_Transient)
unsigned long                             bResetup : 1;                        // 0x0370 (0x0004)
[0x0000000800002000] [0x00000004] (CPF_Transient)
unsigned long                             bIsMovingToLevel : 1;                   // 0x0370 (0x0004)
[0x0000000800002000] [0x00000008] (CPF_Transient)
struct FLightmassPrimitiveSettings              LightmassSettings;                        // 0x0374
(0x001C) [0x0000000000000001] (CPF_Edit)
int32_t                              CollisionMipLevel;                       // 0x0390 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                              MobileLODBias;                           // 0x0394 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FLandscapeLayerStruct>             LayerInfoObjs;                            // 0x0398
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
int32_t                              ComponentSizeQuads;                         // 0x03A8 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                              SubsectionSizeQuads;                      // 0x03AC (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                              NumSubsections;                          // 0x03B0 (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeProxy");
}
return uClassPointer;
};

};

// Class Engine.Landscape
// 0x0024 (0x03B4 - 0x03D8)
class ALandscape : public ALandscapeProxy
{
public:
TArray<struct FName>                    LayerNames;                     // 0x03B8 (0x0010)
[0x0000000020400000] (CPF_NeedCtorLink | CPF_Deprecated)
TArray<struct FLandscapeLayerInfo>           LayerInfos;                     // 0x03C8
(0x0010) [0x0000000020400000] (CPF_NeedCtorLink | CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Landscape");
}

return uClassPointer;
};

};

// Class Engine.Terrain
// 0x0144 (0x0268 - 0x03AC)
class ATerrain : public AInfo
{
public:
TArray<struct FTerrainHeight>              Heights;                        // 0x0268 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FTerrainInfoData>             InfoData;                       // 0x0278 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FTerrainLayer>               Layers;                         // 0x0288 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
int32_t                           NormalMapLayer;                  // 0x0298 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FTerrainDecoLayer>             DecoLayers;                      // 0x02A0 (0x0010)
[0x0000000000480003] (CPF_Edit | CPF_Const | CPF_Component | CPF_NeedCtorLink)
TArray<struct FAlphaMap>                  AlphaMaps;                      // 0x02B0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<class UTerrainComponent*>            TerrainComponents;                    // 0x02C0
(0x0010) [0x000000040448000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
int32_t                           NumSectionsX;                   // 0x02D0 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                           NumSectionsY;                   // 0x02D4 (0x0004)
```

```
[0x0000000000000002] (CPF_Const)
TArray<struct ATerrain_FTerrainWeightedMaterial>   WeightedMaterials;                    //
0x02D8 (0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)
TArray<class UTerrainWeightMapTexture*>            WeightedTextureMaps;                  //
0x02E8 (0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                          MaxTesselationLevel;                    // 0x02F8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          MinTessellationLevel;                   // 0x02FC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            TesselationDistanceScale;               // 0x0300 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            TessellationCheckDistance;              // 0x0304 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          CollisionTesselationLevel;              // 0x0308 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FCachedTerrainMaterialArray           CachedTerrainMaterials[0x2];             // 0x0310
(0x0020) [0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                          NumVerticesX;                           // 0x0330 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                          NumVerticesY;                           // 0x0334 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                          NumPatchesX;                            // 0x0338 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          NumPatchesY;                            // 0x033C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          MaxComponentSize;                       // 0x0340 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          StaticLightingResolution;               // 0x0344 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bIsOverridingLightResolution : 1;           // 0x0348 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bBilinearFilterLightmapGeneration : 1;       // 0x0348
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       bCastShadow : 1;                        // 0x0348 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                       bForceDirectLightMap : 1;                   // 0x0348 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                       bCastDynamicShadow : 1;                     // 0x0348 (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                       bEnableSpecular : 1;                    // 0x0348 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                       bBlockRigidBody : 1;                    // 0x0348 (0x0004)
[0x0000000000000003] [0x00000040] (CPF_Edit | CPF_Const)
unsigned long                       bAllowRigidBodyUnderneath : 1;              // 0x0348 (0x0004)
[0x0000000000000003] [0x00000080] (CPF_Edit | CPF_Const)
unsigned long                       bAcceptsDynamicLights : 1;                  // 0x0348 (0x0004)
[0x0000000000000003] [0x00000100] (CPF_Edit | CPF_Const)
unsigned long                       bMorphingEnabled : 1;                     // 0x0348 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                       bMorphingGradientsEnabled : 1;              // 0x0348 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                       bLocked : 1;                        // 0x0348 (0x0004)
[0x0000000000000000] [0x00000800]
unsigned long                       bHeightmapLocked : 1;                     // 0x0348 (0x0004)
```

```
[0x0000000000000000] [0x00001000]
unsigned long                    bShowingCollision : 1;                  // 0x0348 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                    bUseWorldOriginTextureUVs : 1;              // 0x0348 (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                    bShowWireframe : 1;                   // 0x0348 (0x0004)
[0x0000000000000001] [0x00008000] (CPF_Edit)
class UPhysicalMaterial*                 TerrainPhysMaterialOverride;               // 0x0350
(0x0008) [0x0000000000000003] (CPF_Edit | CPF_Const)
struct FLightingChannelContainer              LightingChannels;                   // 0x0358
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
struct FLightmassPrimitiveSettings             LightmassSettings;                   // 0x035C
(0x001C) [0x0000000000000001] (CPF_Edit)
struct FPointer                     ReleaseResourcesFence;                 // 0x0378 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                         EditorTessellationLevel;               // 0x0380 (0x0004)
[0x0000000000002001] (CPF_Edit | CPF_Transient)
TArray<struct FSelectedTerrainVertex>           SelectedVertices;                    // 0x0388
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FColor                      WireframeColor;                     // 0x0398 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FGuid                       LightingGuid;                      // 0x039C (0x0010)
[0x0000000800000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Terrain");
}

return uClassPointer;
};

void eventPostBeginPlay();
void CalcLayerBounds();
};

// Class Engine.LandscapeGizmoActor
// 0x0020 (0x0268 - 0x0288)
class ALandscapeGizmoActor : public AActor
{
public:
float                          Width;                          // 0x0268 (0x0004)
[0x0000000800000001] (CPF_Edit)
float                          Height;                         // 0x026C (0x0004)
[0x0000000800000001] (CPF_Edit)
float                          LengthZ;                         // 0x0270 (0x0004)
[0x0000000800000001] (CPF_Edit)
float                          MarginZ;                         // 0x0274 (0x0004)
[0x0000000800000001] (CPF_Edit)
```

```
float                              MinRelativeZ;                        // 0x0278 (0x0004)
[0x0000000800000001] (CPF_Edit)
float                              RelativeScaleZ;                      // 0x027C (0x0004)
[0x0000000800000001] (CPF_Edit)
class ULandscapeInfo*                         TargetLandscapeInfo;                    // 0x0280 (0x0008)
[0x0000000800022001] (CPF_Edit | CPF_Transient | CPF_EditConst)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeGizmoActor");
}

return uClassPointer;
};

};

// Class Engine.LandscapeGizmoActiveActor
// 0x0130 (0x0288 - 0x03B8)
class ALandscapeGizmoActiveActor : public ALandscapeGizmoActor
{
public:
uint8_t                            DataType;                            // 0x0288 (0x0001)
[0x0000000800002000] (CPF_Transient)
uint8_t                           UnknownData00[0x50];                    // 0x0290 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeGizmoActiveActor.SelectedData
class UTexture2D*                       GizmoTexture;                          // 0x02E0 (0x0008)
[0x0000000800000000]
struct FVector2D                       TextureScale;                          // 0x02E8 (0x0008)
[0x0000000800000000]
TArray<struct FVector>                    SampledHeight;                          // 0x02F0 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
TArray<struct FVector>                    SampledNormal;                          // 0x0300 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
int32_t                            SampleSizeX;                         // 0x0310 (0x0004)
[0x0000000800000000]
int32_t                            SampleSizeY;                         // 0x0314 (0x0004)
[0x0000000800000000]
float                              CachedWidth;                         // 0x0318 (0x0004)
[0x0000000800000000]
float                              CachedHeight;                        // 0x031C (0x0004)
[0x0000000800000000]
float                              CachedScaleXY;                        // 0x0320 (0x0004)
[0x0000000800000000]
struct FVector                        FrustumVerts[0x8];                     // 0x0324 (0x0060)
[0x0000000800002000] (CPF_Transient)
class UMaterial*                       GizmoMaterial;                        // 0x0388 (0x0008)
[0x0000000800000000]
class UMaterialInstance*                    GizmoDataMaterial;                     // 0x0390 (0x0008)
```

```
[0x0000000800000000]
class UMaterial*                    GizmoMeshMaterial;                  // 0x0398 (0x0008)
[0x0000000800000000]
class UMaterial*                    GizmoMeshMaterial2;                 // 0x03A0 (0x0008)
[0x0000000800000000]
TArray<struct FName>                    LayerNames;                     // 0x03A8 (0x0010)
[0x0000000800420001] (CPF_Edit | CPF_EditConst | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeGizmoActiveActor");
}

return uClassPointer;
};

};

// Class Engine.LandscapeComponent
// 0x0128 (0x0258 - 0x0380)
class ULandscapeComponent : public UPrimitiveComponent
{
public:
int32_t                         SectionBaseX;                   // 0x0258 (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                         SectionBaseY;                   // 0x025C (0x0004)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
int32_t                         ComponentSizeQuads;                 // 0x0260 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         SubsectionSizeQuads;                // 0x0264 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         NumSubsections;                 // 0x0268 (0x0004)
[0x0000000000000002] (CPF_Const)
class UMaterialInterface*               OverrideMaterial;                   // 0x0270 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInstanceConstant*            MaterialInstance;                   // 0x0278
(0x0008) [0x0000000000000000]
TArray<struct FWeightmapLayerAllocationInfo>      WeightmapLayerAllocations;              //
0x0280 (0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class UTexture2D*>                 WeightmapTextures;                  // 0x0290
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FVector4                     WeightmapScaleBias;                 // 0x02A0 (0x0010)
[0x0000000000000000]
float                           WeightmapSubsectionOffset;              // 0x02B0 (0x0004)
[0x0000000000000000]
uint8_t                         UnknownData00[0xC];                 // 0x02B4 (0x000C) MISSED
OFFSET
struct FVector4                     HeightmapScaleBias;                 // 0x02C0 (0x0010)
[0x0000000000000000]
```

```
class UTexture2D*                      HeightmapTexture;                        // 0x02D0 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FBoxSphereBounds                   CachedBoxSphereBounds;                   // 0x02D8
(0x001C) [0x0000000000000002] (CPF_Const)
struct FBox                            CachedLocalBox;                          // 0x02F4 (0x001C)
[0x0000000000000002] (CPF_Const)
struct FGuid                           LightingGuid;                            // 0x0310 (0x0010)
[0x0000000800000002] (CPF_Const)
TArray<class UShadowMap2D*>                 ShadowMaps;                              // 0x0320
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FGuid>                   IrrelevantLights;                        // 0x0330 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FLightMapRef                    LightMap;                                // 0x0340 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                        EditToolRenderData;                      // 0x0348 (0x0008)
[0x0000000000001000] (CPF_Native)
int32_t                                CollisionMipLevel;                       // 0x0350 (0x0004)
[0x0000000000000000]
struct FPointer                        PlatformData;                            // 0x0358 (0x0008)
[0x0000000000001000] (CPF_Native)
int32_t                                PlatformDataSize;                        // 0x0360 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
unsigned long                          bNeedPostUndo : 1;                       // 0x0364 (0x0004)
[0x0000000800002000] [0x00000001] (CPF_Transient)
int32_t                                ForcedLOD;                               // 0x0368 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                                NeighborLOD[0x8];                        // 0x036C (0x0008)
[0x0000000000000000]
uint8_t                                NeighborLODBias[0x8];                    // 0x0374 (0x0008)
[0x0000000000000000]
int32_t                                LODBias;                                 // 0x037C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeComponent");
}

return uClassPointer;
};

};

// Class Engine.LandscapeGizmoRenderComponent
// 0x0000 (0x0258 - 0x0258)
class ULandscapeGizmoRenderComponent : public UPrimitiveComponent
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeGizmoRenderComponent");
}

return uClassPointer;
};

};

// Class Engine.LandscapeHeightfieldCollisionComponent
// 0x0118 (0x0258 - 0x0370)
class ULandscapeHeightfieldCollisionComponent : public UPrimitiveComponent
{
public:
struct FUntypedBulkData_Mirror                CollisionHeightData;                       // 0x0258
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FName>                          ComponentLayers;                           // 0x02B0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FUntypedBulkData_Mirror                DominantLayerData;                         // 0x02C0
(0x0058) [0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                                       SectionBaseX;                              // 0x0318 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                       SectionBaseY;                              // 0x031C (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                       CollisionSizeQuads;                        // 0x0320 (0x0004)
[0x0000000000000000]
float                                         CollisionScale;                            // 0x0324 (0x0004)
[0x0000000000000000]
TArray<uint8_t>                               CollisionQuadFlags;                        // 0x0328 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class UPhysicalMaterial*>              PhysicalMaterials;                         // 0x0338
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FPointer                               RBHeightfield;                             // 0x0348 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FBoxSphereBounds                       CachedBoxSphereBounds;                     // 0x0350
(0x001C) [0x0000000000000002] (CPF_Const)
unsigned long                                 bIncludeHoles : 1;                         // 0x036C (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                                 bHeightFieldDataHasHole : 1;               // 0x036C (0x0004)
[0x0000000800002000] [0x00000002] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.LandscapeHeightfieldCollisionComponent");
}

return uClassPointer;
};

};

// Class Engine.TerrainComponent
// 0x009C (0x0258 - 0x02F4)
class UTerrainComponent : public UPrimitiveComponent
{
public:
TArray<class UShadowMap2D*>                ShadowMaps;                        // 0x0258
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FGuid>                       IrrelevantLights;                  // 0x0268 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FPointer                            TerrainObject;                     // 0x0278 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                                    SectionBaseX;                      // 0x0280 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                    SectionBaseY;                      // 0x0284 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                    SectionSizeX;                      // 0x0288 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                    SectionSizeY;                      // 0x028C (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                    TrueSectionSizeX;                  // 0x0290 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                                    TrueSectionSizeY;                  // 0x0294 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FLightMapRef                        LightMap;                          // 0x0298 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FTerrainPatchBounds>         PatchBounds;                       // 0x02A0
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FTerrainMaterialMask>        BatchMaterials;                    // 0x02B0
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                                    FullBatch;                         // 0x02C0 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FTerrainBVTree                      BVTree;                            // 0x02C8 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FVector>                     CollisionVertices;                 // 0x02D8 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FPointer                            RBHeightfield;                     // 0x02E8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
unsigned long                              bDisplayCollisionLevel : 1;        // 0x02F0 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.TerrainComponent");
}

return uClassPointer;
};

};

// Class Engine.LandscapeInfo
// 0x0304 (0x0060 - 0x0364)
class ULandscapeInfo : public UObject
{
public:
struct FGuid                        LandscapeGuid;                      // 0x0060 (0x0010)
[0x0000000000000002] (CPF_Const)
uint8_t                         UnknownData00[0x50];                    // 0x0070 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeInfo.LayerInfoMap
struct FPointer                      DataInterface;                     // 0x00C0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                         UnknownData01[0x50];                    // 0x00C8 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeInfo.XYtoComponentMap
uint8_t                         UnknownData02[0x50];                    // 0x0118 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeInfo.XYtoCollisionComponentMap
class ALandscapeProxy*                    LandscapeProxy;                      // 0x0168 (0x0008)
[0x0000000000000002] (CPF_Const)
uint8_t                         UnknownData03[0x50];                    // 0x0170 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeInfo.XYtoAddCollisionMap
struct FSet_Mirror                    Proxies;                        // 0x01C0 (0x0050)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FSet_Mirror                    SelectedComponents;                   // 0x0210 (0x0050)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FSet_Mirror                    SelectedCollisionComponents;               // 0x0260
(0x0050) [0x0000000000001002] (CPF_Const | CPF_Native)
struct FSet_Mirror                    SelectedRegionComponents;                // 0x02B0
(0x0050) [0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                         UnknownData04[0x50];                    // 0x0300 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.LandscapeInfo.SelectedRegion
class FString                       HeightmapFilePath;                   // 0x0350 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
unsigned long                       bIsValid : 1;                     // 0x0360 (0x0004)
[0x0000000800002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeInfo");
}

return uClassPointer;
```

```
};

};

// Class Engine.LandscapeLayerInfoObject
// 0x0018 (0x0060 - 0x0078)
class ULandscapeLayerInfoObject : public UObject
{
public:
struct FName                          LayerName;                          // 0x0060 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UPhysicalMaterial*                   PhysMaterial;                          // 0x0068 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                              Hardness;                          // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bNoWeightBlend : 1;                    // 0x0074 (0x0004)
[0x0000000800000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LandscapeLayerInfoObject");
}

return uClassPointer;
};

};

// Class Engine.TerrainWeightMapTexture
// 0x0018 (0x0280 - 0x0298)
class UTerrainWeightMapTexture : public UTexture2D
{
public:
class ATerrain*                        ParentTerrain;                         // 0x0280 (0x0008)
[0x0000000000000002] (CPF_Const)
TArray<struct FPointer>                    WeightedMaterials;                     // 0x0288 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TerrainWeightMapTexture");
}

return uClassPointer;
```

```
};

};

// Class Engine.TerrainLayerSetup
// 0x0010 (0x0060 - 0x0070)
class UTerrainLayerSetup : public UObject
{
public:
TArray<struct FTerrainFilteredMaterial>          Materials;                              // 0x0060 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TerrainLayerSetup");
}

return uClassPointer;
};

void PostBeginPlay();
};

// Class Engine.TerrainMaterial
// 0x006C (0x0060 - 0x00CC)
class UTerrainMaterial : public UObject
{
public:
struct FMatrix                              LocalToMapping;                          // 0x0060 (0x0040)
[0x0000000000000000]
uint8_t                                 MappingType;                             // 0x00A0 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                   MappingScale;                            // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MappingRotation;                         // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MappingPanU;                             // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MappingPanV;                             // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*                   Material;                                // 0x00B8 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                           DisplacementMap;                         // 0x00C0 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                                   DisplacementScale;                       // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TerrainMaterial");
}

return uClassPointer;
};

};

// Class Engine.DataStoreClient
// 0x0050 (0x0070 - 0x00C0)
class UDataStoreClient : public UUIRoot
{
public:
TArray<class FString>                    GlobalDataStoreClasses;                    // 0x0070 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<class UUIDataStore*>                  GlobalDataStores;                          // 0x0080 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class FString>                    PlayerDataStoreClassNames;                  // 0x0090
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<class UClass*>                     PlayerDataStoreClasses;                    // 0x00A0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FPlayerDataStoreGroup>            PlayerDataStores;                            // 0x00B0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DataStoreClient");
}

return uClassPointer;
};

void DebugDumpDataStoreInfo(unsigned long bVerbose);
void eventNotifyGameSessionEnded();
class UClass* FindDataStoreClass(class UClass* RequiredMetaClass);
void GetPlayerDataStoreClasses(TArray<class UClass*>& out_DataStoreClasses);
int32_t FindPlayerDataStoreIndex(class ULocalPlayer* PlayerOwner);
bool UnregisterDataStore(class UUIDataStore* DataStore);
bool RegisterDataStore(class UUIDataStore* DataStore, class ULocalPlayer* PlayerOwner);
class UUIDataStore* CreateDataStore(class UClass* DataStoreClass);
class UUIDataStore* FindDataStore(struct FName DataStoreTag, class ULocalPlayer*
PlayerOwner);
};
```

```
// Class Engine.Console
// 0x01E8 (0x00D0 - 0x02B8)
class UConsole : public UInteraction
{
public:
class ULocalPlayer*                        ConsoleTargetPlayer;                       // 0x00D0 (0x0008)
[0x0000000000000000]
class UTexture2D*                          DefaultTexture_Black;                      // 0x00D8 (0x0008)
[0x0000000000000000]
class UTexture2D*                          DefaultTexture_White;                      // 0x00E0 (0x0008)
[0x0000000000000000]
struct FName                               ConsoleKey;                                // 0x00E8 (0x0008)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
struct FName                               TypeKey;                                   // 0x00F0 (0x0008)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
int32_t                                    MaxScrollbackSize;                         // 0x00F8 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
TArray<class FString>                      Scrollback;                                // 0x0100 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                    SBHead;                                    // 0x0110 (0x0004)
[0x0000000000000000]
int32_t                                    SBPos;                                     // 0x0114 (0x0004)
[0x0000000000000000]
int32_t                                    HistoryTop;                                // 0x0118 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                    HistoryBot;                                // 0x011C (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                    HistoryCur;                                // 0x0120 (0x0004)
[0x0000000000004000] (CPF_Config)
class FString                              History[0x10];                             // 0x0128 (0x0100)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
unsigned long                              bNavigatingHistory : 1;                    // 0x0228 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                              bCaptureKeyInput : 1;                      // 0x0228 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                              bCtrl : 1;                                 // 0x0228 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                              bEnableUI : 1;                             // 0x0228 (0x0004)
[0x0000000000004000] [0x00000008] (CPF_Config)
unsigned long                              bAutoCompleteLocked : 1;                   // 0x0228 (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                              bRequireCtrlToNavigateAutoComplete : 1;    // 0x0228
(0x0004) [0x0000000000004000] [0x00000020] (CPF_Config)
unsigned long                              bIsRuntimeAutoCompleteUpToDate : 1;        // 0x0228
(0x0004) [0x0000000000002000] [0x00000040] (CPF_Transient)
class FString                              TypedStr;                                  // 0x0230 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                    TypedStrPos;                               // 0x0240 (0x0004)
[0x0000000000000000]
TArray<struct FAutoCompleteCommand>        ManualAutoCompleteList;                    //
0x0248 (0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<struct FAutoCompleteCommand>        AutoCompleteList;                          // 0x0258
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FAutoCompleteCommand>        NativeAutoCompleteList;                    //
```

```
0x0268 (0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
int32_t                              AutoCompleteIndex;                       // 0x0278 (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FAutoCompleteNode                    AutoCompleteTree;                           // 0x0280
(0x0028) [0x0000000000003000] (CPF_Native | CPF_Transient)
TArray<int32_t>                            AutoCompleteIndices;                    // 0x02A8 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Console");
}

return uClassPointer;
};

void eventOutputText(class FString Text);
void UpdateCompleteIndices();
void BuildRuntimeAutoCompleteList(unsigned long bForce);
};

// Class Engine.Input
// 0x00E0 (0x00D0 - 0x01B0)
class UInput : public UInteraction
{
public:
TArray<struct FKeyBind>                    Bindings;                               // 0x00D0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FName>                       PressedKeys;                          // 0x00E0 (0x0010)
[0x0000008000400000] (CPF_NeedCtorLink)
TArray<struct FName>                       ReleasedAxisKeys;                         // 0x00F0 (0x0010)
[0x0000008000400000] (CPF_NeedCtorLink)
int32_t                              CurrentControllerId;                     // 0x0100 (0x0004)
[0x0000000000000002] (CPF_Const)
uint8_t                              CurrentEvent;                            // 0x0104 (0x0001)
[0x0000000000000002] (CPF_Const)
float                               CurrentDelta;                            // 0x0108 (0x0004)
[0x0000000000000002] (CPF_Const)
float                               CurrentDeltaTime;                          // 0x010C (0x0004)
[0x0000000000000002] (CPF_Const)
uint8_t                             UnknownData00[0x50];                       // 0x0110 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.Input.NameToPtr
TArray<struct FPointer>                    AxisArray;                              // 0x0160 (0x0010)
[0x0000000000101002] (CPF_Const | CPF_Native)
TArray<class USeqEvent_Input*>                CachedInputEvents;                         // 0x0170
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class USeqEvent_AnalogInput*>            CachedAnalogInputEvents;                   //
0x0180 (0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class USeqEvent_TouchInput*>             CachedTouchInputEvents;                   // 0x0190
```

```
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FTouchTracker>                    CurrentTouches;                         // 0x01A0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Input");
}

return uClassPointer;
};

void SetBind(class FString Command, struct FName& BindName);
class FString GetBind(struct FName& Key);
void ResetInput();
};

// Class Engine.PlayerInput
// 0x0148 (0x01B0 - 0x02F8)
class UPlayerInput : public UInput
{
public:
unsigned long                           bUsingGamepad : 1;                          // 0x01B0 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                           bInvertMouse : 1;                         // 0x01B0 (0x0004)
[0x0000000000044000] [0x00000002] (CPF_Config | CPF_GlobalConfig)
unsigned long                           bInvertTurn : 1;                        // 0x01B0 (0x0004)
[0x0000000000044000] [0x00000004] (CPF_Config | CPF_GlobalConfig)
unsigned long                           bWasForward : 1;                          // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                           bWasBack : 1;                           // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                           bWasLeft : 1;                         // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                           bWasRight : 1;                          // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                           bEdgeForward : 1;                         // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                           bEdgeBack : 1;                          // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                           bEdgeLeft : 1;                        // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                           bEdgeRight : 1;                         // 0x01B0 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                           bEnableMouseSmoothing : 1;                  // 0x01B0 (0x0004)
[0x0000000000044000] [0x00000800] (CPF_Config | CPF_GlobalConfig)
unsigned long                           bEnableFOVScaling : 1;                  // 0x01B0 (0x0004)
[0x0000000000000000] [0x00001000]
unsigned long                           bLockTurnUntilRelease : 1;                  // 0x01B0 (0x0004)
```

```
[0x0000000000002000] [0x00002000] (CPF_Transient)
struct FName                         LastAxisKeyName;                         // 0x01B4 (0x0008)
[0x0000000000000002] (CPF_Const)
float                           DoubleClickTimer;                         // 0x01BC (0x0004)
[0x0000000000000000]
float                           DoubleClickTime;                          // 0x01C0 (0x0004)
[0x0000000000044000] (CPF_Config | CPF_GlobalConfig)
float                           MouseSensitivity;                         // 0x01C4 (0x0004)
[0x0000000000004000] (CPF_Config)
float                           aBaseX;                                   // 0x01C8 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aBaseY;                                   // 0x01CC (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aBaseZ;                                   // 0x01D0 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aMouseX;                                  // 0x01D4 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aMouseY;                                  // 0x01D8 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aForward;                                 // 0x01DC (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aTurn;                                    // 0x01E0 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aStrafe;                                  // 0x01E4 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aUp;                                      // 0x01E8 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aLookUp;                                  // 0x01EC (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aRightAnalogTrigger;                      // 0x01F0 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aLeftAnalogTrigger;                       // 0x01F4 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aPS3AccelX;                               // 0x01F8 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aPS3AccelY;                               // 0x01FC (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aPS3AccelZ;                               // 0x0200 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aPS3Gyro;                                 // 0x0204 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aWiiUPointerX;                            // 0x0208 (0x0004)
[0x0000000000000004] (CPF_Input)
float                           aWiiUPointerY;                            // 0x020C (0x0004)
[0x0000000000000004] (CPF_Input)
struct FVector                       aTilt;                                   // 0x0210 (0x000C)
[0x0000000000000004] (CPF_Input)
struct FVector                       aRotationRate;                           // 0x021C (0x000C)
[0x0000000000000004] (CPF_Input)
struct FVector                       aGravity;                                // 0x0228 (0x000C)
[0x0000000000000004] (CPF_Input)
struct FVector                       aAcceleration;                           // 0x0234 (0x000C)
[0x0000000000000004] (CPF_Input)
struct FVector                       aTouch[0x5];                             // 0x0240 (0x003C)
```

```
[0x0000000000000004] (CPF_Input)
struct FVector                  aBackTouch[0x5];                // 0x027C (0x003C)
[0x0000000000000004] (CPF_Input)
float                           RawJoyUp;                       // 0x02B8 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           RawJoyRight;                    // 0x02BC (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           RawJoyLookRight;                // 0x02C0 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           RawJoyLookUp;                   // 0x02C4 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                           MoveForwardSpeed;               // 0x02C8 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
float                           MoveStrafeSpeed;                // 0x02CC (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
float                           LookRightScale;                 // 0x02D0 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
float                           LookUpScale;                    // 0x02D4 (0x0004)
[0x0000000000004001] (CPF_Edit | CPF_Config)
uint8_t                         bStrafe;                        // 0x02D8 (0x0001)
[0x0000000000000004] (CPF_Input)
uint8_t                         bXAxis;                         // 0x02D9 (0x0001)
[0x0000000000000004] (CPF_Input)
uint8_t                         bYAxis;                         // 0x02DA (0x0001)
[0x0000000000000004] (CPF_Input)
float                           ZeroTime[0x2];                  // 0x02DC (0x0008)
[0x0000000000000000]
float                           SmoothedMouse[0x2];             // 0x02E4 (0x0008)
[0x0000000000000000]
int32_t                         MouseSamples;                   // 0x02EC (0x0004)
[0x0000000000000000]
float                           MouseSamplingTotal;             // 0x02F0 (0x0004)
[0x0000000000000000]
float                           AutoUnlockTurnTime;             // 0x02F4 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlayerInput");
}

return uClassPointer;
};

void PreClientTravel(class FString PendingURL, uint8_t TravelType, unsigned long
bIsSeamlessTravel);
void ClientInitInputSystem();
void InitInputSystem();
float SmoothMouse(float aMouse, float DeltaTime, int32_t Index, uint8_t& SampleCount);
```

```
void ClearSmoothing();
void SmartJump();
void Jump();
void ProcessInputMatching(float DeltaTime);
uint8_t CheckForDoubleClickMove(float DeltaTime);
void CatchDoubleClickInput();
void eventPlayerInput(float DeltaTime);
void AdjustMouseSensitivity(float FOVScale);
void PostProcessInput(float DeltaTime);
void PreProcessInput(float DeltaTime);
void DrawHUD(class AHUD* H);
void SetSensitivity(float F);
bool InvertTurn();
bool InvertMouse();
void CancelMobileInput();
};

// Class Engine.PlayerManagerInteraction
// 0x0000 (0x00D0 - 0x00D0)
class UPlayerManagerInteraction : public UInteraction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlayerManagerInteraction");
}

return uClassPointer;
};

};

// Class Engine.UISceneClient
// 0x00AC (0x0070 - 0x011C)
class UUISceneClient : public UUIRoot
{
public:
struct FPointer                    VfTable_FExec;                    // 0x0070 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
struct FPointer                    RenderViewport;                   // 0x0078 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FIntPoint                   MousePosition;                    // 0x0080 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class UDataStoreClient*                 DataStoreManager;                 // 0x0088 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FMatrix                     CanvasToScreen;                   // 0x0090 (0x0040)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FMatrix                     InvCanvasToScreen;                // 0x00D0 (0x0040)
```

```
[0x0000000000002002] (CPF_Const | CPF_Transient)
class UPostProcessChain*                    UIScenePostProcess;                     // 0x0110
(0x0008) [0x0000000000002000] (CPF_Transient)
unsigned long                               bEnablePostProcess : 1;                 // 0x0118 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UISceneClient");
}

return uClassPointer;
};

void eventInitializeSceneClient();
struct FMatrix GetInverseCanvasToScreen();
struct FMatrix GetCanvasToScreen();
bool IsUIActive(int32_t Flags);
};

// Class Engine.UISoundTheme
// 0x0010 (0x0060 - 0x0070)
class UUISoundTheme : public UObject
{
public:
TArray<struct FSoundEventMapping>           SoundEventBindings;                     // 0x0060
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UISoundTheme");
}

return uClassPointer;
};

void eventProcessSoundEvent(struct FName SoundEventName, class APlayerController*
SoundOwner);
};

// Class Engine.UIDataStoreSubscriber
// 0x0000 (0x0060 - 0x0060)
class UUIDataStoreSubscriber : public UInterface
{
```

```
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStoreSubscriber");
}

return uClassPointer;
};

void ClearBoundDataStores();
void GetBoundDataStores(TArray<class UUIDataStore*>& out_BoundDataStores);
void NotifyDataStoreValueUpdated(class UUIDataStore* SourceDataStore, unsigned long
bValuesInvalidated, struct FName PropertyTag, class UUIDataProvider* SourceProvider, int32_t
ArrayIndex);
bool RefreshSubscriberValue(int32_t BindingIndex);
class FString GetDataStoreBinding(int32_t BindingIndex);
void SetDataStoreBinding(class FString MarkupText, int32_t BindingIndex);
};

// Class Engine.UIDataStorePublisher
// 0x0000 (0x0060 - 0x0060)
class UUIDataStorePublisher : public UUIDataStoreSubscriber
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStorePublisher");
}

return uClassPointer;
};

bool SaveSubscriberValue(int32_t BindingIndex, TArray<class UUIDataStore*>&
out_BoundDataStores);
};

// Class Engine.UIDataProvider
// 0x0000 (0x0070 - 0x0070)
class UUIDataProvider : public UUIRoot
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider");
}

return uClassPointer;
};

};

// Class Engine.UIDataProvider_OnlinePlayerDataBase
// 0x0004 (0x0070 - 0x0074)
class UUIDataProvider_OnlinePlayerDataBase : public UUIDataProvider
{
public:
int32_t                              PlayerControllerId;                        // 0x0070 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_OnlinePlayerDataBase");
}

return uClassPointer;
};

void eventOnUnregister();
void eventOnRegister(class ULocalPlayer* InPlayer);
};

// Class Engine.UIDataProvider_OnlineFriendMessages
// 0x0074 (0x0074 - 0x00E8)
class UUIDataProvider_OnlineFriendMessages : public UUIDataProvider_OnlinePlayerDataBase
{
public:
TArray<struct FOnlineFriendMessage>             Messages;                                // 0x0078
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class FString                                SendingPlayerNameCol;                   // 0x0088 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                                bIsFriendInviteCol;                    // 0x0098 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                                bWasAcceptedCol;                       // 0x00A8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                                bWasDeniedCol;                         // 0x00B8 (0x0010)
```

```
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    MessageCol;                      // 0x00C8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    LastInviteFrom;                  // 0x00D8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_OnlineFriendMessages");
}

return uClassPointer;
};

void OnGameInviteReceived(uint8_t LocalUserNum, class FString InviterName);
void OnLoginChange(uint8_t LocalUserNum);
void OnFriendMessageReceived(uint8_t LocalUserNum, struct FUniqueNetId SendingPlayer,
class FString SendingNick, class FString Message);
void OnFriendInviteReceived(uint8_t LocalUserNum, struct FUniqueNetId RequestingPlayer, class
FString RequestingNick, class FString Message);
void ReadMessages();
void eventOnUnregister();
void eventOnRegister(class ULocalPlayer* InPlayer);
};

// Class Engine.UIDataProvider_OnlineFriends
// 0x00F4 (0x0074 - 0x0168)
class UUIDataProvider_OnlineFriends : public UUIDataProvider_OnlinePlayerDataBase
{
public:
TArray<struct FOnlineFriend>           FriendsList;                     // 0x0078 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    NickNameCol;                     // 0x0088 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    PresenceInfoCol;                 // 0x0098 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    FriendStateCol;                  // 0x00A8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    bIsOnlineCol;                    // 0x00B8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    bIsPlayingCol;                   // 0x00C8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    bIsPlayingThisGameCol;           // 0x00D8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    bIsJoinableCol;                  // 0x00E8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    bHasVoiceSupportCol;             // 0x00F8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    bHaveInvitedCol;                 // 0x0108 (0x0010)
```

```
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    bHasInvitedYouCol;                    // 0x0118 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    OfflineText;                    // 0x0128 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    OnlineText;                    // 0x0138 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    AwayText;                    // 0x0148 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    BusyText;                    // 0x0158 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_OnlineFriends");
}

return uClassPointer;
};

void eventRefreshFriendsList();
void OnLoginChange(uint8_t LocalUserNum);
void OnFriendsReadComplete(unsigned long bWasSuccessful);
void eventOnUnregister();
void eventOnRegister(class ULocalPlayer* InPlayer);
};

// Class Engine.UIDataProvider_OnlinePartyChatList
// 0x0094 (0x0074 - 0x0108)
class UUIDataProvider_OnlinePartyChatList : public UUIDataProvider_OnlinePlayerDataBase
{
public:
TArray<struct FOnlinePartyMember>                PartyMembersList;                    // 0x0078
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<class FString>                NatTypes;                    // 0x0088 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    NickNameCol;                    // 0x0098 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    NatTypeCol;                    // 0x00A8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    IsLocalCol;                    // 0x00B8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    IsInPartyVoiceCol;                    // 0x00C8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    IsTalkingCol;                    // 0x00D8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    IsInGameSessionCol;                    // 0x00E8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    IsPlayingThisGameCol;                    // 0x00F8 (0x0010)
```

```
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_OnlinePartyChatList");
}

return uClassPointer;
};

void eventRefreshMembersList();
void OnLoginChange(uint8_t LocalUserNum);
void eventOnUnregister();
void eventOnRegister(class ULocalPlayer* InPlayer);
};

// Class Engine.UIDataProvider_OnlinePlayerStorage
// 0x0030 (0x0074 - 0x00A4)
class UUIDataProvider_OnlinePlayerStorage : public UUIDataProvider_OnlinePlayerDataBase
{
public:
class UOnlinePlayerStorage*                    Profile;                              // 0x0078 (0x0008)
[0x0000000000000000]
struct FName                                ProviderName;                         // 0x0080 (0x0008)
[0x0000000000000002] (CPF_Const)
unsigned long                               bWasErrorLastRead : 1;                   // 0x0088 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                               bIsExternalUIOpen : 1;                   // 0x0088 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                               bNeedsDeferredRefresh : 1;                 // 0x0088 (0x0004)
[0x0000000000000000] [0x00000004]
TArray<struct FPlayerStorageArrayProvider>        PlayerStorageArrayProviders;               //
0x0090 (0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                  DeviceStorageSizeNeeded;                 // 0x00A0 (0x0004)
[0x0000000000004000] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_OnlinePlayerStorage");
}

return uClassPointer;
};
```

```
void OnExternalUIChange(unsigned long bIsOpening);
void OnStorageDeviceChange();
void OnDeviceSelectionComplete(unsigned long bWasSuccessful);
void ShowDeviceSelection();
void RefreshStorageData();
void OnLoginChange(uint8_t LocalUserNum);
void OnReadStorageComplete(uint8_t LocalUserNum, unsigned long bWasSuccessful);
void eventOnUnregister();
void eventOnRegister(class ULocalPlayer* InPlayer);
void ClearReadCompleteDelegate(class UOnlinePlayerInterface* PlayerInterface, uint8_t
LocalUserNum);
void AddReadCompleteDelegate(class UOnlinePlayerInterface* PlayerInterface, uint8_t
LocalUserNum);
bool GetData(class UOnlinePlayerInterface* PlayerInterface, uint8_t LocalUserNum);
bool WriteData(class UOnlinePlayerInterface* PlayerInterface, uint8_t LocalUserNum, int32_t
DeviceID, class UOnlinePlayerStorage* PlayerStorage);
bool ReadData(class UOnlinePlayerInterface* PlayerInterface, uint8_t LocalUserNum, int32_t
DeviceID, class UOnlinePlayerStorage* PlayerStorage);
};

// Class Engine.UIDataProvider_OnlineProfileSettings
// 0x0004 (0x00A4 - 0x00A8)
class UUIDataProvider_OnlineProfileSettings : public UUIDataProvider_OnlinePlayerStorage
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_OnlineProfileSettings");
}

return uClassPointer;
};

void OnStorageDeviceChange();
void RefreshStorageData();
void ClearReadCompleteDelegate(class UOnlinePlayerInterface* PlayerInterface, uint8_t
LocalUserNum);
void AddReadCompleteDelegate(class UOnlinePlayerInterface* PlayerInterface, uint8_t
LocalUserNum);
bool GetData(class UOnlinePlayerInterface* PlayerInterface, uint8_t LocalUserNum);
bool WriteData(class UOnlinePlayerInterface* PlayerInterface, uint8_t LocalUserNum, int32_t
DeviceID, class UOnlinePlayerStorage* PlayerStorage);
bool ReadData(class UOnlinePlayerInterface* PlayerInterface, uint8_t LocalUserNum, int32_t
DeviceID, class UOnlinePlayerStorage* PlayerStorage);
};

// Class Engine.UIDataProvider_PlayerAchievements
// 0x0014 (0x0074 - 0x0088)
```

```
class UUIDataProvider_PlayerAchievements : public UUIDataProvider_OnlinePlayerDataBase
{
public:
TArray<struct FAchievementDetails>              Achievements;                              // 0x0078
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_PlayerAchievements");
}

return uClassPointer;
};

void UpdateAchievements();
void OnLoginChange(uint8_t LocalUserNum);
void eventOnUnregister();
void eventOnRegister(class ULocalPlayer* InPlayer);
void OnPlayerAchievementUnlocked(unsigned long bWasSuccessful);
void OnPlayerAchievementsChanged(int32_t TitleId);
void GetAchievementDetails(int32_t AchievementId, struct FAchievementDetails&
OutAchievementDetails);
class FString GetAchievementIconPathName(int32_t AchievementId, unsigned long
bReturnLockedIcon);
void PopulateAchievementIcons();
int32_t GetMaxTotalGamerScore();
int32_t GetTotalGamerScore();
};

// Class Engine.UIDataProvider_OnlinePlayerStorageArray
// 0x0030 (0x0070 - 0x00A0)
class UUIDataProvider_OnlinePlayerStorageArray : public UUIDataProvider
{
public:
class UOnlinePlayerStorage*                    PlayerStorage;                         // 0x0070 (0x0008)
[0x0000000000000000]
int32_t                                  PlayerStorageId;                        // 0x0078 (0x0004)
[0x0000000000000000]
class FString                               ColumnHeaderText;                      // 0x0080 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FName>                         Values;                                // 0x0090 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_OnlinePlayerStorageArray");
}

return uClassPointer;
};

};

// Class Engine.UIDataProvider_SettingsArray
// 0x0038 (0x0070 - 0x00A8)
class UUIDataProvider_SettingsArray : public UUIDataProvider
{
public:
class USettings*                        Settings;                         // 0x0070 (0x0008)
[0x0000000000000000]
int32_t                                 SettingsId;                       // 0x0078 (0x0004)
[0x0000000000000000]
struct FName                            SettingsName;                     // 0x007C (0x0008)
[0x0000000000000000]
class FString                           ColumnHeaderText;                 // 0x0088 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FIdToStringMapping>            Values;                      // 0x0098 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_SettingsArray");
}

return uClassPointer;
};

};

// Class Engine.UIDataStore
// 0x0030 (0x0070 - 0x00A0)
class UUIDataStore : public UUIDataProvider
{
public:
struct FName                            Tag;                              // 0x0070 (0x0008)
[0x0000000000000000]
TArray<struct FScriptDelegate>             RefreshSubscriberNotifies;             // 0x0078
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                     __OnDataStoreValueUpdated__Delegate;        // 0x0088
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore");
}

return uClassPointer;
};

class UDataStoreClient* GetDataStoreClient();
void eventRefreshSubscribers(struct FName PropertyTag, unsigned long bInvalidateValues, class UUIDataProvider* SourceProvider, int32_t ArrayIndex);
bool NotifyGameSessionEnded();
void eventSubscriberDetached(class UUIDataStoreSubscriber* Subscriber);
void eventSubscriberAttached(class UUIDataStoreSubscriber* Subscriber);
void eventUnregistered(class ULocalPlayer* PlayerOwner);
void eventRegistered(class ULocalPlayer* PlayerOwner);
void OnDataStoreValueUpdated(class UUIDataStore* SourceDataStore, unsigned long bValuesInvalidated, struct FName PropertyTag, class UUIDataProvider* SourceProvider, int32_t ArrayIndex);
};

// Class Engine.UIDataStore_DynamicResource
// 0x0070 (0x00A0 - 0x0110)
class UUIDataStore_DynamicResource : public UUIDataStore
{
public:
class UUIDataProvider_OnlineProfileSettings*      ProfileProvider;                          // 0x00A0 (0x0008) [0x0000000000002000] (CPF_Transient)
class UUIDataStore_GameResource*                GameResourceDataStore;                    // 0x00A8 (0x0008) [0x0000000000002000] (CPF_Transient)
TArray<struct FDynamicResourceProviderDefinition>  ResourceProviderDefinitions;               // 0x00B0 (0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
struct FMultiMap_Mirror                         ResourceProviders;                        // 0x00C0 (0x0050) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_DynamicResource");
}

return uClassPointer;
};

void eventUnregistered(class ULocalPlayer* PlayerOwner);
void eventRegistered(class ULocalPlayer* PlayerOwner);
void OnLoginChange(uint8_t LocalUserNum);
```

```
bool GetResourceProviders(struct FName ProviderTag, TArray<class
UUIResourceCombinationProvider*>& out_Providers);
int32_t FindProviderTypeIndex(struct FName ProviderTag);
};

// Class Engine.UIDataStore_Fonts
// 0x0000 (0x00A0 - 0x00A0)
class UUIDataStore_Fonts : public UUIDataStore
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_Fonts");
}

return uClassPointer;
};

};

// Class Engine.UIDataStore_GameResource
// 0x0060 (0x00A0 - 0x0100)
class UUIDataStore_GameResource : public UUIDataStore
{
public:
TArray<struct FGameResourceDataProvider>          ElementProviderTypes;                    //
0x00A0 (0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
struct FMultiMap_Mirror                          ListElementProviders;                    // 0x00B0 (0x0050)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_GameResource");
}

return uClassPointer;
};

bool GetResourceProviders(struct FName ProviderTag, TArray<class
UUIResourceDataProvider*>& out_Providers);
int32_t FindProviderTypeIndex(struct FName ProviderTag);
};
```

```
// Class Engine.UIDataStore_MenuItems
// 0x0068 (0x0100 - 0x0168)
class UUIDataStore_MenuItems : public UUIDataStore_GameResource
{
public:
struct FName                              CurrentGameSettingsTag;                    // 0x0100 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FMultiMap_Mirror                       OptionProviders;                          // 0x0108 (0x0050)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class UUIDataProvider_MenuItem*>            DynamicProviders;                           // 0x0158
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_MenuItems");
}

return uClassPointer;
};

void eventUnregistered(class ULocalPlayer* PlayerOwner);
void eventRegistered(class ULocalPlayer* PlayerOwner);
void OnGameSettingsChanged(class UUIDataProvider* SourceProvider, struct FName PropTag);
};

// Class Engine.UIDataStore_GameState
// 0x0018 (0x00A0 - 0x00B8)
class UUIDataStore_GameState : public UUIDataStore
{
public:
struct FScriptDelegate                         __OnRefreshDataFieldValue__Delegate;          // 0x00A0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_GameState");
}

return uClassPointer;
};

bool NotifyGameSessionEnded();
void OnRefreshDataFieldValue();
};
```

```
// Class Engine.UIDataStore_Registry
// 0x0010 (0x00A0 - 0x00B0)
class UUIDataStore_Registry : public UUIDataStore
{
public:
TArray<struct FRegistryKeyValuePair>          RegistryData;                              // 0x00A0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_Registry");
}

return uClassPointer;
};

void eventSetData(class FString Key, class FString Value);
bool eventGetData(class FString Key, class FString& out_Data);
};

// Class Engine.UIDataStore_Remote
// 0x0000 (0x00A0 - 0x00A0)
class UUIDataStore_Remote : public UUIDataStore
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_Remote");
}

return uClassPointer;
};

};

// Class Engine.UIDataStore_OnlineGameSearch
// 0x0038 (0x00A0 - 0x00D8)
class UUIDataStore_OnlineGameSearch : public UUIDataStore_Remote
{
public:
struct FName                                  SearchResultsName;                         // 0x00A0 (0x0008)
[0x0000000000000002] (CPF_Const)
```

```
class UOnlineSubsystem*                    OnlineSub;                            // 0x00A8 (0x0008)
[0x0000000000000000]
class UOnlineGameInterface*                GameInterface_Object;                 // 0x00B0
(0x0008) [0x0000000000000000]
class UOnlineGameInterface*                GameInterface_Interface;              // 0x00B8
(0x0008) [0x0000000000000000]
TArray<struct FGameSearchCfg>              GameSearchCfgList;                    // 0x00C0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
int32_t                          SelectedIndex;                        // 0x00D0 (0x0004)
[0x0000000000000000]
int32_t                          ActiveSearchIndex;                    // 0x00D4 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_OnlineGameSearch");
}

return uClassPointer;
};

void ClearAllSearchResults();
void eventMoveToPrevious(unsigned long bInvalidateExistingSearchResults);
void eventMoveToNext(unsigned long bInvalidateExistingSearchResults);
void eventSetCurrentByName(struct FName SearchName, unsigned long
bInvalidateExistingSearchResults);
void eventSetCurrentByIndex(int32_t NewIndex, unsigned long
bInvalidateExistingSearchResults);
int32_t FindSearchConfigurationIndex(struct FName SearchTag);
class UOnlineGameSearch* eventGetActiveGameSearch();
class UOnlineGameSearch* eventGetCurrentGameSearch();
bool eventShowHostGamercard(uint8_t ControllerIndex, int32_t ListIndex);
bool eventGetSearchResultFromIndex(int32_t ListIndex, struct FOnlineGameSearchResult&
Result);
void OnSearchComplete(unsigned long bWasSuccessful);
bool OverrideQuerySubmission(uint8_t ControllerId, class UOnlineGameSearch* Search);
bool eventSubmitGameSearch(uint8_t ControllerIndex, unsigned long
bInvalidateExistingSearchResults);
bool InvalidateCurrentSearchResults();
void eventInit();
};

// Class Engine.UIDataStore_OnlinePlayerData
// 0x00F8 (0x00A0 - 0x0198)
class UUIDataStore_OnlinePlayerData : public UUIDataStore_Remote
{
public:
class UUIDataProvider_OnlineFriends*       FriendsProvider;                      // 0x00A0
(0x0008) [0x0000000000000000]
```

```
int32_t                              PlayerControllerId;                    // 0x00A8 (0x0004)
[0x0000000000000000]
class FString                        ProfileSettingsClassName;              // 0x00B0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        ProfileSettingsClass;                  // 0x00C0 (0x0008)
[0x0000000000000000]
class UUIDataProvider_OnlineProfileSettings*      ProfileProvider;                        // 0x00C8
(0x0008) [0x0000000000000000]
class FString                        ProfileProviderClassName;              // 0x00D0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        ProfileProviderClass;                  // 0x00E0 (0x0008)
[0x0000000000000000]
class FString                        PlayerStorageClassName;                // 0x00E8 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        PlayerStorageClass;                    // 0x00F8 (0x0008)
[0x0000000000000000]
class UUIDataProvider_OnlinePlayerStorage*        StorageProvider;                        // 0x0100
(0x0008) [0x0000000000000000]
class FString                        StorageProviderClassName;              // 0x0108 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        StorageProviderClass;                  // 0x0118 (0x0008)
[0x0000000000000000]
class UUIDataProvider_OnlineFriendMessages*       FriendMessagesProvider;                 //
0x0120 (0x0008) [0x0000000000000000]
class UUIDataProvider_PlayerAchievements*         AchievementsProvider;                   //
0x0128 (0x0008) [0x0000000000000000]
class FString                        FriendsProviderClassName;              // 0x0130 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        FriendsProviderClass;                  // 0x0140 (0x0008)
[0x0000000000000000]
class FString                        FriendMessagesProviderClassName;       // 0x0148
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        FriendMessagesProviderClass;           // 0x0158 (0x0008)
[0x0000000000000000]
class FString                        AchievementsProviderClassName;         // 0x0160
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        AchievementsProviderClass;             // 0x0170 (0x0008)
[0x0000000000000000]
class FString                        PartyChatProviderClassName;            // 0x0178 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class UClass*                        PartyChatProviderClass;                // 0x0188 (0x0008)
[0x0000000000000000]
class UUIDataProvider_OnlinePartyChatList*        PartyChatProvider;                      // 0x0190
(0x0008) [0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_OnlinePlayerData");
}
```

```
return uClassPointer;
};

class UOnlinePlayerStorage* eventGetCachedPlayerStorage(int32_t ControllerId);
class UOnlineProfileSettings* eventGetCachedPlayerProfile(int32_t ControllerId);
void ClearDelegates();
void RegisterDelegates();
void OnPlayerDataChange();
void OnLoginChange(uint8_t LocalUserNum);
void eventOnUnregister();
void eventOnRegister(class ULocalPlayer* InPlayer);
};

// Class Engine.UIDataStore_OnlineStats
// 0x0090 (0x00A0 - 0x0130)
class UUIDataStore_OnlineStats : public UUIDataStore_Remote
{
public:
TArray<class UClass*>                StatsReadClasses;                // 0x00A0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FName                         StatsReadName;                   // 0x00B0 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FPlayerNickMetaData           PlayerNickData;                  // 0x00B8 (0x0018)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FRankMetaData                 RankNameMetaData;                // 0x00D0
(0x0018) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FName                         TotalRowsName;                   // 0x00E8 (0x0008)
[0x0000000000000002] (CPF_Const)
TArray<class UOnlineStatsRead*>      StatsReadObjects;                // 0x00F0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class UOnlineStatsRead*              StatsRead;                       // 0x0100 (0x0008)
[0x0000000000000000]
uint8_t                              CurrentReadType;                 // 0x0108 (0x0001)
[0x0000000000000000]
class UOnlineStatsInterface*         StatsInterface_Object;           // 0x0110
(0x0008) [0x0000000000000000]
class UOnlineStatsInterface*         StatsInterface_Interface;        // 0x0118
(0x0008) [0x0000000000000000]
class UOnlinePlayerInterface*        PlayerInterface_Object;          // 0x0120
(0x0008) [0x0000000000000000]
class UOnlinePlayerInterface*        PlayerInterface_Interface;       // 0x0128
(0x0008) [0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_OnlineStats");
}
```

```
return uClassPointer;
};

static void SortResultsByRank(class UOnlineStatsRead* StatsToSort);
void OnReadComplete(unsigned long bWasSuccessful);
bool eventShowGamercard(uint8_t ConrollerIndex, int32_t ListIndex);
bool eventRefreshStats(uint8_t ControllerIndex);
void SetStatsReadInfo();
void eventInit();
};

// Class Engine.UIDataStore_Settings
// 0x0000 (0x00A0 - 0x00A0)
class UUIDataStore_Settings : public UUIDataStore
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_Settings");
}

return uClassPointer;
};

};

// Class Engine.UIDataStore_OnlineGameSettings
// 0x001C (0x00A0 - 0x00BC)
class UUIDataStore_OnlineGameSettings : public UUIDataStore_Settings
{
public:
TArray<struct FGameSettingsCfg>                GameSettingsCfgList;                    // 0x00A0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
class UClass*                              SettingsProviderClass;                  // 0x00B0 (0x0008)
[0x0000000000000002] (CPF_Const)
int32_t                                SelectedIndex;                          // 0x00B8 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_OnlineGameSettings");
}
```

```
return uClassPointer;
};

void eventUnregistered(class ULocalPlayer* PlayerOwner);
void eventRegistered(class ULocalPlayer* PlayerOwner);
void eventMoveToPrevious();
void eventMoveToNext();
void eventSetCurrentByName(struct FName SettingsName);
void eventSetCurrentByIndex(int32_t NewIndex);
class UUIDataProvider_Settings* eventGetCurrentProvider();
class UOnlineGameSettings* eventGetCurrentGameSettings();
bool eventCreateGame(uint8_t ControllerIndex);
};

// Class Engine.UIDataStore_StringBase
// 0x0000 (0x00A0 - 0x00A0)
class UUIDataStore_StringBase : public UUIDataStore
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_StringBase");
}

return uClassPointer;
};

};

// Class Engine.UIDataStore_InputAlias
// 0x0060 (0x00A0 - 0x0100)
class UUIDataStore_InputAlias : public UUIDataStore_StringBase
{
public:
TArray<struct FUIDataStoreInputAlias>            InputAliases;                              // 0x00A0
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
uint8_t                                   UnknownData00[0x50];                   // 0x00B0 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.UIDataStore_InputAlias.InputAliasLookupMap

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_InputAlias");
}
```

```
return uClassPointer;
};

bool HasAliasMappingForPlatform(struct FName DesiredAlias, uint8_t DesiredPlatform);
int32_t FindInputAliasIndex(struct FName DesiredAlias);
bool GetAliasInputKeyDataByIndex(int32_t AliasIndex, uint8_t OverridePlatform, struct
FRawInputKeyEventData& out_InputKeyData);
bool GetAliasInputKeyData(struct FName DesiredAlias, uint8_t OverridePlatform, struct
FRawInputKeyEventData& out_InputKeyData);
struct FName GetAliasInputKeyNameByIndex(int32_t AliasIndex, uint8_t OverridePlatform);
struct FName GetAliasInputKeyName(struct FName DesiredAlias, uint8_t OverridePlatform);
class FString GetAliasFontMarkupByIndex(int32_t AliasIndex, uint8_t OverridePlatform);
class FString GetAliasFontMarkup(struct FName DesiredAlias, uint8_t OverridePlatform);
};

// Class Engine.UIDataStore_StringAliasMap
// 0x0064 (0x00A0 - 0x0104)
class UUIDataStore_StringAliasMap : public UUIDataStore_StringBase
{
public:
TArray<struct FUIMenuInputMap>                MenuInputMapArray;                       // 0x00A0
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
struct FMap_Mirror                            MenuInputSets;                           // 0x00B0 (0x0050)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                                       PlayerIndex;                             // 0x0100 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataStore_StringAliasMap");
}

return uClassPointer;
};

int32_t GetStringWithFieldName(class FString FieldName, class FString& MappedString);
int32_t FindMappingWithFieldName(class FString FieldName, class FString SetName);
class ULocalPlayer* GetPlayerOwner();
};

// Class Engine.UIPropertyDataProvider
// 0x0028 (0x0070 - 0x0098)
class UUIPropertyDataProvider : public UUIDataProvider
{
public:
TArray<class UClass*>                         ComplexPropertyTypes;                    // 0x0070
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FScriptDelegate                        __CanSupportComplexPropertyType__Delegate;    //
```

```
0x0080 (0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIPropertyDataProvider");
}

return uClassPointer;
};

bool CanSupportComplexPropertyType(class UProperty* UnsupportedProperty);
};

// Class Engine.UIDataProvider_Settings
// 0x000C (0x0098 - 0x00A4)
class UUIDataProvider_Settings : public UUIPropertyDataProvider
{
public:
class USettings*                    Settings;                       // 0x0098 (0x0008)
[0x0000000000000000]
unsigned long                       bIsAListRow : 1;                // 0x00A0 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_Settings");
}

return uClassPointer;
};

};

// Class Engine.UIResourceDataProvider
// 0x0004 (0x0098 - 0x009C)
class UUIResourceDataProvider : public UUIPropertyDataProvider
{
public:
unsigned long                       bSkipDuringEnumeration : 1;          // 0x0098 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIResourceDataProvider");
}

return uClassPointer;
};

void eventInitializeProvider(unsigned long bIsEditor);
};

// Class Engine.UIDataProvider_MenuItem
// 0x00B4 (0x009C - 0x0150)
class UUIDataProvider_MenuItem : public UUIResourceDataProvider
{
public:
uint8_t                          OptionType;                     // 0x00A0 (0x0001)
[0x0000000000004000] (CPF_Config)
TArray<struct FName>                   OptionSet;                     // 0x00A8 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                       DataStoreMarkup;                  // 0x00B8 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                       DescriptionMarkup;                // 0x00C8 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
struct FName                         RequiredGameMode;                 // 0x00D8 (0x0008)
[0x0000000000004000] (CPF_Config)
class FString                       FriendlyName;                   // 0x00E0 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
class FString                       CustomFriendlyName;               // 0x00F0 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                       Description;                   // 0x0100 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
unsigned long                        bEditableCombo : 1;               // 0x0110 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
unsigned long                        bNumericCombo : 1;                // 0x0110 (0x0004)
[0x0000000000004000] [0x00000002] (CPF_Config)
unsigned long                        bKeyboardOrMouseOption : 1;            // 0x0110 (0x0004)
[0x0000000000004000] [0x00000004] (CPF_Config)
unsigned long                        bOnlineOnly : 1;                 // 0x0110 (0x0004)
[0x0000000000004000] [0x00000008] (CPF_Config)
unsigned long                        bOfflineOnly : 1;                // 0x0110 (0x0004)
[0x0000000000004000] [0x00000010] (CPF_Config)
unsigned long                        bSearchAllInis : 1;               // 0x0110 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                        bRemoveOn360 : 1;                // 0x0110 (0x0004)
[0x0000000000004000] [0x00000040] (CPF_Config)
unsigned long                        bRemoveOnPC : 1;                 // 0x0110 (0x0004)
[0x0000000000004000] [0x00000080] (CPF_Config)
unsigned long                        bRemoveOnPS3 : 1;                // 0x0110 (0x0004)
[0x0000000000004000] [0x00000100] (CPF_Config)
int32_t                          EditBoxMaxLength;                 // 0x0114 (0x0004)
[0x0000000000004000] (CPF_Config)
```

```
struct FUIRangeData                          RangeData;                              // 0x0118 (0x0014)
[0x0000000000004000] (CPF_Config)
TArray<struct FName>                          SchemaCellFields;                        // 0x0130 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                          IniName;                              // 0x0140 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIDataProvider_MenuItem");
}

return uClassPointer;
};

};

// Class Engine.UIResourceCombinationProvider
// 0x0010 (0x0070 - 0x0080)
class UUIResourceCombinationProvider : public UUIDataProvider
{
public:
class UUIResourceDataProvider*                    StaticDataProvider;                        // 0x0070
(0x0008) [0x0000000000002000] (CPF_Transient)
class UUIDataProvider_OnlineProfileSettings*       ProfileProvider;                           // 0x0078
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIResourceCombinationProvider");
}

return uClassPointer;
};

void ClearProviderReferences();
void eventInitializeProvider(unsigned long bIsEditor, class UUIResourceDataProvider*
InStaticResourceProvider, class UUIDataProvider_OnlineProfileSettings* InProfileProvider);
};

// Class Engine.GameUISceneClient
// 0x0094 (0x011C - 0x01B0)
class UGameUISceneClient : public UUISceneClient
{
```

```
public:
float                              LatestDeltaTime;                          // 0x0120 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FDouble                          DoubleClickStartTime;                    // 0x0128 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FIntPoint                         DoubleClickStartPosition;                 // 0x0130 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FMap_Mirror                          InitialPressedKeys;                    // 0x0138 (0x0050)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
unsigned long                           bUpdateInputProcessingStatus : 1;            // 0x0188
(0x0004) [0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                           bUpdateSceneViewportSizes : 1;             // 0x0188 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                           bEnableDebugInput : 1;                   // 0x0188 (0x0004)
[0x0000000000004000] [0x00000004] (CPF_Config)
unsigned long                           bRenderDebugInfo : 1;                    // 0x0188 (0x0004)
[0x0000000000004000] [0x00000008] (CPF_Config)
unsigned long                           bCaptureUnprocessedInput : 1;              // 0x0188 (0x0004)
[0x0000000000004002] [0x00000010] (CPF_Const | CPF_Config)
TArray<struct FName>                         NavAliases;                        // 0x0190 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<struct FName>                         AxisInputKeys;                      // 0x01A0 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameUISceneClient");
}

return uClassPointer;
};

int32_t FindLocalPlayerIndex(class UPlayer* P);
void NotifyPlayerRemoved(int32_t PlayerIndex, class ULocalPlayer* RemovedPlayer);
void NotifyPlayerAdded(int32_t PlayerIndex, class ULocalPlayer* AddedPlayer);
void NotifyGameSessionEnded();
void NotifyClientTravel(class APlayerController* TravellingPlayer, class FString TravelURL, uint8_t
TravelType, unsigned long bIsSeamlessTravel);
void eventPauseGame(unsigned long bDesiredPauseState, int32_t PlayerIndex);
bool CanUnpauseInternalUI();
void RequestInputProcessingUpdate();
static uint8_t GetCurrentNetMode();
};

// Class Engine.Scene
// 0x0000 (0x0060 - 0x0060)
class UScene : public UObject
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Scene");
}

return uClassPointer;
};

};

// Class Engine.InstancedFoliageActor
// 0x0068 (0x0268 - 0x02D0)
class AInstancedFoliageActor : public AActor
{
public:
struct FMap_Mirror                    FoliageMeshes;                    // 0x0268 (0x0050)
[0x0000000000001002] (CPF_Const | CPF_Native)
class UStaticMesh*                    SelectedMesh;                     // 0x02B8 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
TArray<class UInstancedStaticMeshComponent*>
InstancedStaticMeshComponents;          // 0x02C0 (0x0010) [0x000000000448200A]
(CPF_Const | CPF_ExportObject | CPF_Transient | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InstancedFoliageActor");
}

return uClassPointer;
};

};

// Class Engine.InteractiveFoliageActor
// 0x005C (0x0288 - 0x02E4)
class AInteractiveFoliageActor : public AStaticMeshActor
{
public:
class UCylinderComponent*                 CylinderComponent;                // 0x0288
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
struct FVector                        TouchingActorEntryPosition;           // 0x0290 (0x000C)
[0x0000000000002000] (CPF_Transient)
```

```
struct FVector                          FoliageVelocity;                          // 0x029C (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FVector                          FoliageForce;                             // 0x02A8 (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FVector                          FoliagePosition;                          // 0x02B4 (0x000C)
[0x0000000000002000] (CPF_Transient)
float                                   FoliageDamageImpulseScale;                // 0x02C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FoliageTouchImpulseScale;                 // 0x02C4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FoliageStiffness;                         // 0x02C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FoliageStiffnessQuadratic;                // 0x02CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   FoliageDamping;                           // 0x02D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MaxDamageImpulse;                         // 0x02D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MaxTouchImpulse;                          // 0x02D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MaxForce;                                 // 0x02DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   Mass;                                     // 0x02E0 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InteractiveFoliageActor");
}

return uClassPointer;
};

void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
};

// Class Engine.InteractiveFoliageComponent
// 0x0008 (0x0300 - 0x0308)
class UInteractiveFoliageComponent : public UStaticMeshComponent
{
public:
struct FPointer                         FoliageSceneProxy;                        // 0x0300 (0x0008)
[0x0000000000201002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InteractiveFoliageComponent");
}

return uClassPointer;
};

};

// Class Engine.ActorFactoryInteractiveFoliage
// 0x0004 (0x00B4 - 0x00B8)
class UActorFactoryInteractiveFoliage : public UActorFactoryStaticMesh
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactoryInteractiveFoliage");
}

return uClassPointer;
};

};

// Class Engine.InstancedFoliageSettings
// 0x0068 (0x0060 - 0x00C8)
class UInstancedFoliageSettings : public UObject
{
public:
float                               Density;                            // 0x0060 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               Radius;                             // 0x0064 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               ScaleMinX;                            // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               ScaleMinY;                            // 0x006C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               ScaleMinZ;                            // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               ScaleMaxX;                            // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               ScaleMaxY;                            // 0x0078 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               ScaleMaxZ;                            // 0x007C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                             LockScaleX : 1;                         // 0x0080 (0x0004)
```

```
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        LockScaleY : 1;                      // 0x0080 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                        LockScaleZ : 1;                      // 0x0080 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                        AlignToNormal : 1;                   // 0x0080 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                        RandomYaw : 1;                       // 0x0080 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                        UniformScale : 1;                    // 0x0080 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                        ReapplyDensity : 1;                  // 0x0080 (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                        ReapplyRadius : 1;                   // 0x0080 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                        ReapplyAlignToNormal : 1;                 // 0x0080 (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                        ReapplyRandomYaw : 1;                     // 0x0080 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                        ReapplyScaleX : 1;                   // 0x0080 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                        ReapplyScaleY : 1;                   // 0x0080 (0x0004)
[0x0000000000000000] [0x00000800]
unsigned long                        ReapplyScaleZ : 1;                   // 0x0080 (0x0004)
[0x0000000000000000] [0x00001000]
unsigned long                        ReapplyRandomPitchAngle : 1;                // 0x0080 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                        ReapplyGroundSlope : 1;                  // 0x0080 (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                        ReapplyHeight : 1;                   // 0x0080 (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                        ReapplyLandscapeLayer : 1;                // 0x0080 (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                        ReapplyZOffset : 1;                  // 0x0080 (0x0004)
[0x0000000000000000] [0x00020000]
unsigned long                        CastShadow : 1;                      // 0x0080 (0x0004)
[0x0000000000000001] [0x00040000] (CPF_Edit)
unsigned long                        bCastDynamicShadow : 1;                   // 0x0080 (0x0004)
[0x0000000000000001] [0x00080000] (CPF_Edit)
unsigned long                        bCastStaticShadow : 1;                  // 0x0080 (0x0004)
[0x0000000000000001] [0x00100000] (CPF_Edit)
unsigned long                        bSelfShadowOnly : 1;                   // 0x0080 (0x0004)
[0x0000000000000001] [0x00200000] (CPF_Edit)
unsigned long                        bNoModSelfShadow : 1;                    // 0x0080 (0x0004)
[0x0000000000000001] [0x00400000] (CPF_Edit)
unsigned long                        bAcceptsDynamicDominantLightShadows : 1;      // 0x0080
(0x0004) [0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                        bCastHiddenShadow : 1;                   // 0x0080 (0x0004)
[0x0000000000000001] [0x01000000] (CPF_Edit)
unsigned long                        bCastShadowAsTwoSided : 1;                 // 0x0080 (0x0004)
[0x0000000000000001] [0x02000000] (CPF_Edit)
unsigned long                        bAcceptsLights : 1;                  // 0x0080 (0x0004)
[0x0000000000000003] [0x04000000] (CPF_Edit | CPF_Const)
unsigned long                        bAcceptsDynamicLights : 1;                // 0x0080 (0x0004)
```

```
[0x0000000000000003] [0x08000000] (CPF_Edit | CPF_Const)
unsigned long                          bUseOnePassLightingOnTranslucency : 1;         // 0x0080
(0x0004) [0x0000000000000003] [0x10000000] (CPF_Edit | CPF_Const)
unsigned long                          bUsePrecomputedShadows : 1;                 // 0x0080
(0x0004) [0x0000000000000003] [0x20000000] (CPF_Edit | CPF_Const)
unsigned long                          bCollideActors : 1;                        // 0x0080 (0x0004)
[0x0000000000000001] [0x40000000] (CPF_Edit)
unsigned long                          bBlockActors : 1;                          // 0x0080 (0x0004)
[0x0000000000000001] [0x80000000] (CPF_Edit)
unsigned long                          bBlockNonZeroExtent : 1;                    // 0x0084 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bBlockZeroExtent : 1;                       // 0x0084 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                          IsSelected : 1;                            // 0x0084 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                          ShowNothing : 1;                           // 0x0084 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                          ShowPaintSettings : 1;                      // 0x0084 (0x0004)
[0x0000000000000000] [0x00000010]
unsigned long                          ShowInstanceSettings : 1;                   // 0x0084 (0x0004)
[0x0000000000000000] [0x00000020]
float                            AlignMaxAngle;                          // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RandomPitchAngle;                         // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            GroundSlope;                           // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            HeightMin;                           // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            HeightMax;                           // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FName                           LandscapeLayer;                         // 0x009C (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            ZOffsetMin;                          // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            ZOffsetMax;                          // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                            MaxInstancesPerCluster;                    // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            MaxClusterRadius;                       // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            ReapplyDensityAmount;                     // 0x00B4 (0x0004)
[0x0000000000000000]
int32_t                            StartCullDistance;                      // 0x00B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                            EndCullDistance;                        // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                            CullOption;                          // 0x00C0 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                            DetailMode;                           // 0x00C1 (0x0001)
[0x0000000000000001] (CPF_Edit)
int32_t                            DisplayOrder;                          // 0x00C4 (0x0004)
[0x0000000000000000]
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InstancedFoliageSettings");
}

return uClassPointer;
};

};

// Class Engine.FluidInfluenceActor
// 0x001C (0x0268 - 0x0284)
class AFluidInfluenceActor : public AActor
{
public:
class UArrowComponent*                    FlowDirection;                        // 0x0268 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class USpriteComponent*                   Sprite;                               // 0x0270 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UFluidInfluenceComponent*           InfluenceComponent;                   // 0x0278
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                             bActive : 1;                          // 0x0280 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)
unsigned long                             bToggled : 1;                         // 0x0280 (0x0004)
[0x0000000100000020] [0x00000002] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FluidInfluenceActor");
}

return uClassPointer;
};

void eventReplicatedEvent(struct FName VarName);
void OnToggle(class USeqAct_Toggle* inAction);
};

// Class Engine.FluidSurfaceActor
// 0x0010 (0x0268 - 0x0278)
class AFluidSurfaceActor : public AActor
{
public:
```

```
class UFluidSurfaceComponent*                    FluidComponent;                              // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
class UParticleSystem*                           ProjectileEntryEffect;                      // 0x0270 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FluidSurfaceActor");
}

return uClassPointer;
};

void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
};

// Class Engine.FluidSurfaceActorMovable
// 0x0000 (0x0278 - 0x0278)
class AFluidSurfaceActorMovable : public AFluidSurfaceActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FluidSurfaceActorMovable");
}

return uClassPointer;
};

};

// Class Engine.FluidInfluenceComponent
// 0x0070 (0x0258 - 0x02C8)
class UFluidInfluenceComponent : public UPrimitiveComponent
{
public:
unsigned long                                    bActive : 1;                                // 0x0258 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                    RaindropFillEntireFluid : 1;                // 0x0258 (0x0004)
[0x0000000200000001] [0x00000002] (CPF_Edit)
unsigned long                                    bIsToggleTriggered : 1;                     // 0x0258 (0x0004)
```

```
[0x0000000000002000] [0x00000004] (CPF_Transient)
class AFluidSurfaceActor*                    FluidActor;                              // 0x0260 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                                  InfluenceType;                         // 0x0268 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                    MaxDistance;                           // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    WaveStrength;                          // 0x0270 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    WaveFrequency;                         // 0x0274 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    WavePhase;                             // 0x0278 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    WaveRadius;                            // 0x027C (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    RaindropAreaRadius;                    // 0x0280 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    RaindropRadius;                        // 0x0284 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    RaindropStrength;                      // 0x0288 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    RaindropRate;                          // 0x028C (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    FlowSpeed;                             // 0x0290 (0x0004)
[0x0000000200000001] (CPF_Edit)
int32_t                                  FlowNumRipples;                        // 0x0294 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    FlowSideMotionRadius;                  // 0x0298 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    FlowWaveRadius;                        // 0x029C (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    FlowStrength;                          // 0x02A0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    FlowFrequency;                         // 0x02A4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    SphereOuterRadius;                     // 0x02A8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    SphereInnerRadius;                     // 0x02AC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    SphereStrength;                        // 0x02B0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                    CurrentAngle;                          // 0x02B4 (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
float                                    CurrentTimer;                          // 0x02B8 (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
class AFluidSurfaceActor*                    CurrentFluidActor;                       // 0x02C0 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FluidInfluenceComponent");
}

return uClassPointer;
};

};

// Class Engine.FluidSurfaceComponent
// 0x0100 (0x0258 - 0x0358)
class UFluidSurfaceComponent : public UPrimitiveComponent
{
public:
class UMaterialInterface*                    FluidMaterial;                              // 0x0258 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                                    LightMapResolution;                         // 0x0260 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FLightmassPrimitiveSettings              LightmassSettings;                          // 0x0264
(0x001C) [0x0000000000000001] (CPF_Edit)
unsigned long                              EnableSimulation : 1;                       // 0x0280 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                              EnableDetail : 1;                           // 0x0280 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                              bTiling : 1;                                // 0x0280 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                              bPause : 1;                                 // 0x0280 (0x0004)
[0x0000000000002001] [0x00000008] (CPF_Edit | CPF_Transient)
unsigned long                              bShowSimulationNormals : 1;                 // 0x0280 (0x0004)
[0x0000000000002001] [0x00000010] (CPF_Edit | CPF_Transient)
unsigned long                              bShowSimulationPosition : 1;                // 0x0280 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                              bShowDetailNormals : 1;                     // 0x0280 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                              bShowDetailPosition : 1;                    // 0x0280 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                              bShowFluidSimulation : 1;                   // 0x0280 (0x0004)
[0x0000000000002001] [0x00000100] (CPF_Edit | CPF_Transient)
unsigned long                              bShowFluidDetail : 1;                       // 0x0280 (0x0004)
[0x0000000000002001] [0x00000200] (CPF_Edit | CPF_Transient)
unsigned long                              bTestRipple : 1;                            // 0x0280 (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                              bTestRippleCenterOnDetail : 1;              // 0x0280 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
int32_t                                    SimulationQuadsX;                           // 0x0284 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                    SimulationQuadsY;                           // 0x0288 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      GridSpacing;                                // 0x028C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                      GridSpacingLowRes;                          // 0x0290 (0x0004)
[0x0000000000000001] (CPF_Edit)
class AActor*                              TargetSimulation;                           // 0x0298 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
float                          GPUTessellationFactor;              // 0x02A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FluidDamping;                       // 0x02A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FluidTravelSpeed;                   // 0x02A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FluidHeightScale;                   // 0x02AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FluidUpdateRate;                    // 0x02B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          ForceImpact;                        // 0x02B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          ForceContinuous;                    // 0x02B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          LightingContrast;                   // 0x02BC (0x0004)
[0x0000000000000001] (CPF_Edit)
class AActor*                      TargetDetail;                   // 0x02C0 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                          DeactivationDistance;               // 0x02C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         DetailResolution;                  // 0x02CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DetailSize;                         // 0x02D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DetailDamping;                      // 0x02D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DetailTravelSpeed;                  // 0x02D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DetailTransfer;                     // 0x02DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DetailHeightScale;                  // 0x02E0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          DetailUpdateRate;                   // 0x02E4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          NormalLength;                       // 0x02E8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          TestRippleSpeed;                    // 0x02EC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          TestRippleFrequency;                // 0x02F0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          TestRippleRadius;                   // 0x02F4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FluidWidth;                         // 0x02F8 (0x0004)
[0x0000000000000000]
float                          FluidHeight;                        // 0x02FC (0x0004)
[0x0000000000000000]
float                          TestRippleTime;                     // 0x0300 (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
float                          TestRippleAngle;                    // 0x0304 (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
float                          DeactivationTimer;                  // 0x0308 (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
float                          ViewDistance;                       // 0x030C (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
```

```
struct FVector                                  SimulationPosition;                         // 0x0310 (0x000C)
[0x0000000000003000] (CPF_Native | CPF_Transient)
struct FVector                                  DetailPosition;                             // 0x031C (0x000C)
[0x0000000000003000] (CPF_Native | CPF_Transient)
TArray<uint8_t>                                 ClampMap;                                   // 0x0328 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<class UShadowMap2D*>                     ShadowMaps;                                 // 0x0338
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FLightMapRef                             LightMap;                                   // 0x0348 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                                 FluidSimulation;                            // 0x0350 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FluidSurfaceComponent");
}

return uClassPointer;
};

void SetSimulationPosition(struct FVector WorldPos);
void SetDetailPosition(struct FVector WorldPos);
void ApplyForce(struct FVector WorldPos, float Strength, float Radius, unsigned long bImpulse);
};

// Class Engine.SpeedTreeActor
// 0x0008 (0x0268 - 0x0270)
class ASpeedTreeActor : public AActor
{
public:
class USpeedTreeComponent*                      SpeedTreeComponent;                         // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpeedTreeActor");
}

return uClassPointer;
};

};
```

```
// Class Engine.SpeedTreeComponent
// 0x00F4 (0x0258 - 0x034C)
class USpeedTreeComponent : public UPrimitiveComponent
{
public:
class USpeedTree*                          SpeedTree;                          // 0x0258 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                         bUseLeafCards : 1;                   // 0x0260 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bUseLeafMeshes : 1;                   // 0x0260 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                         bUseBranches : 1;                    // 0x0260 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                         bUseFronds : 1;                     // 0x0260 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                         bUseBillboards : 1;                  // 0x0260 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
float                              Lod3DStart;                         // 0x0264 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              Lod3DEnd;                           // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              LodBillboardStart;                    // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              LodBillboardEnd;                     // 0x0270 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              LodLevelOverride;                    // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              Branch1Material;                       // 0x0278 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              Branch2Material;                       // 0x0280 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              FrondMaterial;                        // 0x0288 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              LeafCardMaterial;                      // 0x0290 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              LeafMeshMaterial;                      // 0x0298 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*              BillboardMaterial;                     // 0x02A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture2D*                    SpeedTreeIcon;                       // 0x02A8 (0x0008)
[0x0000000800000000]
TArray<struct FSpeedTreeStaticLight>            StaticLights;                       // 0x02B0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FLightMapRef                      BranchLightMap;                      // 0x02C0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FLightMapRef                      FrondLightMap;                       // 0x02C8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FLightMapRef                      LeafMeshLightMap;                     // 0x02D0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FLightMapRef                      LeafCardLightMap;                     // 0x02D8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FLightMapRef                      BillboardLightMap;                    // 0x02E0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
```

```
uint8_t                              UnknownData00[0x8];                          // 0x02E8 (0x0008) MISSED
OFFSET
struct FMatrix                           RotationOnlyMatrix;                        // 0x02F0 (0x0040)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FLightmassPrimitiveSettings                LightmassSettings;                          // 0x0330
(0x001C) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpeedTreeComponent");
}

return uClassPointer;
};

void SetMaterial(uint8_t MeshType, class UMaterialInterface* Material);
class UMaterialInterface* GetMaterial(uint8_t MeshType);
};

// Class Engine.SpeedTreeActorFactory
// 0x000C (0x009C - 0x00A8)
class USpeedTreeActorFactory : public UActorFactory
{
public:
class USpeedTree*                              SpeedTree;                                // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpeedTreeActorFactory");
}

return uClassPointer;
};

};

// Class Engine.SpeedTreeComponentFactory
// 0x000C (0x0064 - 0x0070)
class USpeedTreeComponentFactory : public UPrimitiveComponentFactory
{
public:
class USpeedTreeComponent*                      SpeedTreeComponent;                        // 0x0068
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpeedTreeComponentFactory");
}

return uClassPointer;
};

};

// Class Engine.SpeedTree
// 0x0068 (0x0060 - 0x00C8)
class USpeedTree : public UObject
{
public:
unsigned long                    bLegacySpeedTree : 1;               // 0x0060 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
struct FPointer                  SRH;                                // 0x0068 (0x0008)
[0x0000000000201002] (CPF_Const | CPF_Native)
float                            LeafStaticShadowOpacity;            // 0x0070 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*        Branch1Material;                    // 0x0078 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*        Branch2Material;                    // 0x0080 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*        FrondMaterial;                      // 0x0088 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*        LeafCardMaterial;                   // 0x0090 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*        LeafMeshMaterial;                   // 0x0098 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*        BillboardMaterial;                  // 0x00A0 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            WindStrength;                       // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                   WindDirection;                      // 0x00AC (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FGuid                     LightingGuid;                       // 0x00B8 (0x0010)
[0x0000000800000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.SpeedTree");
}

return uClassPointer;
};

};

// Class Engine.LensFlareSource
// 0x000C (0x0268 - 0x0274)
class ALensFlareSource : public AActor
{
public:
class ULensFlareComponent*                    LensFlareComp;                              // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                                 bCurrentlyActive : 1;                       // 0x0270 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LensFlareSource");
}

return uClassPointer;
};

void SetActorParameter(struct FName ParameterName, class AActor* Param);
void SetExtColorParameter(struct FName ParameterName, float Red, float Green, float Blue, float
Alpha);
void SetColorParameter(struct FName ParameterName, struct FLinearColor Param);
void SetVectorParameter(struct FName ParameterName, struct FVector Param);
void SetFloatParameter(struct FName ParameterName, float Param);
void eventReplicatedEvent(struct FName VarName);
void OnToggle(class USeqAct_Toggle* Action);
void eventPostBeginPlay();
void SetTemplate(class ULensFlare* NewTemplate);
};

// Class Engine.LensFlareComponent
// 0x0064 (0x0258 - 0x02BC)
class ULensFlareComponent : public UPrimitiveComponent
{
public:
class ULensFlare*                             Template;                                   // 0x0258 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UDrawLightConeComponent*                PreviewInnerCone;                           // 0x0260
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
```

```
class UDrawLightConeComponent*                PreviewOuterCone;                        // 0x0268
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class UDrawLightRadiusComponent*               PreviewRadius;                          // 0x0270
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
unsigned long                          bAutoActivate : 1;                      // 0x0278 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bIsActive : 1;                     // 0x0278 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                          bHasTranslucency : 1;                   // 0x0278 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                          bHasUnlitTranslucency : 1;                 // 0x0278 (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
unsigned long                          bHasUnlitDistortion : 1;                  // 0x0278 (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                          bUsesSceneColor : 1;                    // 0x0278 (0x0004)
[0x0000000000002000] [0x00000020] (CPF_Transient)
unsigned long                          bHasSeparateTranslucency : 1;                // 0x0278 (0x0004)
[0x0000000000002000] [0x00000040] (CPF_Transient)
unsigned long                          bUseTrueConeCalculation : 1;                // 0x0278 (0x0004)
[0x0000000000002000] [0x00000080] (CPF_Transient)
unsigned long                          bVisibleForMobile : 1;                   // 0x0278 (0x0004)
[0x0000000000000000] [0x00000100]
float                             OuterCone;                         // 0x027C (0x0004)
[0x0000000000002000] (CPF_Transient)
float                             InnerCone;                         // 0x0280 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                             ConeFudgeFactor;                     // 0x0284 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                             Radius;                          // 0x0288 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                             MinStrength;                        // 0x028C (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FLinearColor                        SourceColor;                        // 0x0290 (0x0010)
[0x0000000000000001] (CPF_Edit)
TArray<struct FLensFlareElementMaterials>        Materials;                          // 0x02A0
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FPointer                         ReleaseResourcesFence;                   // 0x02B0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                             NextTraceTime;                       // 0x02B8 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LensFlareComponent");
}

return uClassPointer;
};

class UMaterialInstanceConstant* CreateAndSetMaterialInstanceConstant(int32_t
ElementIndex);
void SetMaterial(int32_t ElementIndex, class UMaterialInterface* Material);
class UMaterialInterface* GetMaterial(int32_t ElementIndex);
void SetIsActive(unsigned long bInIsActive);
void SetSourceColor(struct FLinearColor InSourceColor);
void SetTemplate(class ULensFlare* NewTemplate, unsigned long bForceSet);
};

// Class Engine.LensFlare
// 0x0270 (0x0060 - 0x02D0)
class ULensFlare : public UObject
{
public:
struct FLensFlareElement                    SourceElement;                        // 0x0060 (0x01C8)
[0x0000000004480008] (CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)
class UStaticMesh*                          SourceMesh;                           // 0x0228 (0x0008)
[0x0000000000000001] (CPF_Edit)
uint8_t                                     SourceDPG;                            // 0x0230 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                                     ReflectionsDPG;                       // 0x0231 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
TArray<struct FLensFlareElement>            Reflections;                          // 0x0238 (0x0010)
[0x0000000004480008] (CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)
float                                       OuterCone;                            // 0x0248 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                       InnerCone;                            // 0x024C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                       ConeFudgeFactor;                      // 0x0250 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                       Radius;                               // 0x0254 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                               bUseTrueConeCalculation : 1;          // 0x0258 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                               bUseFixedRelativeBoundingBox : 1;     // 0x0258
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                               bRenderDebugLines : 1;                // 0x0258 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                               ThumbnailImageOutOfDate : 1;          // 0x0258 (0x0004)
[0x0000000000000000] [0x00000008]
float                                       MinStrength;                          // 0x025C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FRawDistributionFloat                ScreenPercentageMap;                  // 0x0260
(0x0028) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FBox                                 FixedRelativeBoundingBox;             // 0x0288 (0x001C)
[0x0000000000000001] (CPF_Edit)
class UInterpCurveEdSetup*                  CurveEdSetup;                         // 0x02A8 (0x0008)
[0x0000000000000008] (CPF_ExportObject)
int32_t                                     ReflectionCount;                      // 0x02B0 (0x0004)
[0x0000000000002000] (CPF_Transient)
```

```
struct FRotator                         ThumbnailAngle;                         // 0x02B4 (0x000C)
[0x0000000000000000]
float                              ThumbnailDistance;                      // 0x02C0 (0x0004)
[0x0000000000000000]
class UTexture2D*                          ThumbnailImage;                        // 0x02C8 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LensFlare");
}

return uClassPointer;
};

};

// Class Engine.TextureFlipBook
// 0x0048 (0x0280 - 0x02C8)
class UTextureFlipBook : public UTexture2D
{
public:
struct FPointer                         VfTable_FTickableObject;                  // 0x0280 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
float                              TimeIntoMovie;                         // 0x0288 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                              TimeSinceLastFrame;                      // 0x028C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                              HorizontalScale;                       // 0x0290 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                              VerticalScale;                        // 0x0294 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                           bPaused : 1;                          // 0x0298 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                           bStopped : 1;                         // 0x0298 (0x0004)
[0x0000000000000002] [0x00000002] (CPF_Const)
unsigned long                           bLooping : 1;                         // 0x0298 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           bAutoPlay : 1;                        // 0x0298 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
int32_t                              HorizontalImages;                      // 0x029C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                              VerticalImages;                       // 0x02A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                              FBMethod;                            // 0x02A4 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                              FrameRate;                           // 0x02A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              FrameTime;                           // 0x02AC (0x0004)
```

```
[0x0000000000000000]
int32_t                              CurrentRow;                              // 0x02B0 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                              CurrentColumn;                            // 0x02B4 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                RenderOffsetU;                           // 0x02B8 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                                RenderOffsetV;                           // 0x02BC (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FPointer                          ReleaseResourcesFence;                  // 0x02C0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TextureFlipBook");
}

return uClassPointer;
};

void SetCurrentFrame(int32_t Row, int32_t Col);
void Stop();
void Pause();
void Play();
};

// Class Engine.Texture2DComposite
// 0x001C (0x0150 - 0x016C)
class UTexture2DComposite : public UTexture
{
public:
TArray<struct FSourceTexture2DRegion>           SourceRegions;                        // 0x0150
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
int32_t                              MaxTextureSize;                         // 0x0160 (0x0004)
[0x0000000000000000]
int32_t                              DestSizeX;                             // 0x0164 (0x0004)
[0x0000000000000000]
int32_t                              DestSizeY;                             // 0x0168 (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Texture2DComposite");
}
return uClassPointer;
};

void ResetSourceRegions();
void UpdateCompositeTexture(int32_t NumMipsToGenerate);
bool SourceTexturesFullyStreamedIn();
};

// Class Engine.Texture2DDynamic
// 0x0014 (0x0150 - 0x0164)
class UTexture2DDynamic : public UTexture
{
public:
int32_t                             SizeX;                          // 0x0150 (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
int32_t                             SizeY;                          // 0x0154 (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
uint8_t                             Format;                          // 0x0158 (0x0001)
[0x0000000000003000] (CPF_Native | CPF_Transient)
int32_t                             NumMips;                          // 0x015C (0x0004)
[0x0000000000003000] (CPF_Native | CPF_Transient)
unsigned long                            bIsResolveTarget : 1;                 // 0x0160 (0x0004)
[0x0000000000003000] [0x00000001] (CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Texture2DDynamic");
}

return uClassPointer;
};

static class UTexture2DDynamic* Create(int32_t InSizeX, int32_t InSizeY, uint8_t InFormat,
unsigned long InIsResolveTarget);
void UpdateMipFromPNG(int32_t MipIdx, TArray<uint8_t>& MipData);
void UpdateMipFromImageData(int32_t MipIdx, struct FImageLayout& ImageData);
void UpdateMipFromJPEG(int32_t MipIdx, TArray<uint8_t>& MipData);
void UpdateMip(int32_t MipIdx, TArray<uint8_t>& MipData);
void Init(int32_t InSizeX, int32_t InSizeY, uint8_t InFormat, unsigned long InIsResolveTarget);
};

// Class Engine.TextureCube
// 0x0048 (0x0150 - 0x0198)
class UTextureCube : public UTexture
{
public:
int32_t                             SizeX;                          // 0x0150 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
```

```
int32_t                             SizeY;                          // 0x0154 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
uint8_t                             Format;                          // 0x0158 (0x0001)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                             NumMips;                           // 0x015C (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                           bIsCubemapValid : 1;                    // 0x0160 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
class UTexture2D*                        FacePosX;                           // 0x0168 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UTexture2D*                        FaceNegX;                           // 0x0170 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UTexture2D*                        FacePosY;                           // 0x0178 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UTexture2D*                        FaceNegY;                           // 0x0180 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UTexture2D*                        FacePosZ;                           // 0x0188 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UTexture2D*                        FaceNegZ;                           // 0x0190 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TextureCube");
}

return uClassPointer;
};

};

// Class Engine.TextureMovie
// 0x0088 (0x0150 - 0x01D8)
class UTextureMovie : public UTexture
{
public:
int32_t                             SizeX;                          // 0x0150 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                             SizeY;                          // 0x0154 (0x0004)
[0x0000000000000002] (CPF_Const)
uint8_t                             Format;                          // 0x0158 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                             AddressX;                          // 0x0159 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                             AddressY;                          // 0x015A (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                             MovieStreamSource;                     // 0x015B (0x0001)
[0x0000000000000001] (CPF_Edit)
class UClass*                           DecoderClass;                      // 0x0160 (0x0008)
```

```
[0x0000000000000002] (CPF_Const)
class UCodecMovie*                    Decoder;                         // 0x0168 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                         Paused : 1;                      // 0x0170 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                         Stopped : 1;                     // 0x0170 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                         Looping : 1;                     // 0x0170 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                         ResetOnLastFrame : 1;            // 0x0170 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                         AutoPlay : 1;                    // 0x0170 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
struct FUntypedBulkData_Mirror               Data;                     // 0x0178 (0x0058)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                       ReleaseCodecFence;               // 0x01D0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TextureMovie");
}

return uClassPointer;
};

void Stop();
void Pause();
void Play();
};

// Class Engine.TextureRenderTarget
// 0x0008 (0x0150 - 0x0158)
class UTextureRenderTarget : public UTexture
{
public:
unsigned long                         bUpdateImmediate : 1;            // 0x0150 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                         bNeedsTwoCopies : 1;             // 0x0150 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                         bRenderOnce : 1;                 // 0x0150 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
float                            TargetGamma;                          // 0x0154 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TextureRenderTarget");
}

return uClassPointer;
};

};

// Class Engine.TextureRenderTarget2D
// 0x0040 (0x0158 - 0x0198)
class UTextureRenderTarget2D : public UTextureRenderTarget
{
public:
int32_t                          SizeX;                           // 0x0158 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                          SizeY;                           // 0x015C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                          Format;                           // 0x0160 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                          AddressX;                          // 0x0161 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                          AddressY;                          // 0x0162 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor                   ClearColor;                         // 0x0164 (0x0010)
[0x0000000000000002] (CPF_Const)
unsigned long                      bForceLinearGamma : 1;                  // 0x0174 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
int32_t                          ExtraTexCreateFlags;                    // 0x0178 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FScriptDelegate                  __EventResourceUpdated__Delegate;           // 0x0180
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TextureRenderTarget2D");
}

return uClassPointer;
};

static class UTextureRenderTarget2D* Create(int32_t InSizeX, int32_t InSizeY, uint8_t InFormat,
struct FLinearColor InClearColor, unsigned long bOnlyRenderOnce);
void EventResourceUpdated(class UTextureRenderTarget2D* RenderTarget);
};

// Class Engine.ScriptedTexture
// 0x0020 (0x0198 - 0x01B8)
class UScriptedTexture : public UTextureRenderTarget2D
{
public:
unsigned long                      bNeedsUpdate : 1;                    // 0x0198 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                      bSkipNextClear : 1;                  // 0x0198 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
struct FScriptDelegate                  __Render__Delegate;                    // 0x01A0 (0x0018)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ScriptedTexture");
}

return uClassPointer;
};

void Render(class UCanvas* C);
};

// Class Engine.TextureRenderTargetCube
// 0x0005 (0x0158 - 0x015D)
class UTextureRenderTargetCube : public UTextureRenderTarget
{
public:
int32_t                            SizeX;                               // 0x0158 (0x0004)
[0x0000000000000001] (CPF_Edit)
uint8_t                            Format;                              // 0x015C (0x0001)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TextureRenderTargetCube");
}

return uClassPointer;
};

};

// Class Engine.AudioDevice
// 0x033C (0x0068 - 0x03A4)
```

```
class UAudioDevice : public USubsystem
{
public:
int32_t                              MaxChannels;                          // 0x0068 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
int32_t                              CommonAudioPoolSize;                   // 0x006C (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
float                                LowPassFilterResonance;                // 0x0070 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
float                                MinCompressedDurationEditor;            // 0x0074 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
float                                MinCompressedDurationGame;              // 0x0078 (0x0004)
[0x0000000000004002] (CPF_Const | CPF_Config)
class FString                        ChirpInSoundNodeWaveName;               // 0x0080 (0x0010)
[0x0000000000404002] (CPF_Const | CPF_Config | CPF_NeedCtorLink)
class USoundNodeWave*                        ChirpInSoundNodeWave;                   // 0x0090
(0x0008) [0x0000000000000002] (CPF_Const)
class FString                        ChirpOutSoundNodeWaveName;               // 0x0098
(0x0010) [0x0000000000404002] (CPF_Const | CPF_Config | CPF_NeedCtorLink)
class USoundNodeWave*                        ChirpOutSoundNodeWave;                  // 0x00A8
(0x0008) [0x0000000000000002] (CPF_Const)
struct FPointer                      CommonAudioPool;                       // 0x00B0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                              CommonAudioPoolFreeBytes;               // 0x00B8 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<class UAudioComponent*>               AudioComponents;                        // 0x00C0
(0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)
TArray<struct FPointer>                  Sources;                              // 0x00D0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FPointer>                  FreeSources;                            // 0x00E0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                              UnknownData00[0x50];                 // 0x00F0 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AudioDevice.WaveInstanceSourceMap
unsigned long                        bGameWasTicking : 1;                   // 0x0140 (0x0004)
[0x0000000000001002] [0x00000001] (CPF_Const | CPF_Native)
unsigned long                        bSoundSpawningEnabled : 1;              // 0x0140 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
TArray<struct FListener>                 Listeners;                            // 0x0148 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint64_t                             CurrentTick;                          // 0x0158 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                              UnknownData01[0x50];                 // 0x0160 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AudioDevice.SoundClasses
uint8_t                              UnknownData02[0x50];                 // 0x01B0 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AudioDevice.SourceSoundClasses
uint8_t                              UnknownData03[0x50];                 // 0x0200 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AudioDevice.CurrentSoundClasses
uint8_t                              UnknownData04[0x50];                 // 0x0250 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AudioDevice.DestinationSoundClasses
uint8_t                              UnknownData05[0x50];                 // 0x02A0 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.AudioDevice.SoundModes
struct FPointer                      Effects;                              // 0x02F0 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
```

```
struct FName                          BaseSoundModeName;                        // 0x02F8 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
class USoundMode*                          CurrentMode;                          // 0x0300 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        SoundModeStartTime;                      // 0x0308 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        SoundModeFadeInStartTime;                  // 0x0310 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        SoundModeFadeInEndTime;                   // 0x0318 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        SoundModeEndTime;                        // 0x0320 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                           ListenerVolumeIndex;                     // 0x0328 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FInteriorSettings                   ListenerInteriorSettings;                 // 0x032C (0x0024)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        InteriorStartTime;                     // 0x0350 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        InteriorEndTime;                       // 0x0358 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        ExteriorEndTime;                       // 0x0360 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        InteriorLPFEndTime;                     // 0x0368 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FDouble                        ExteriorLPFEndTime;                     // 0x0370 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                            InteriorVolumeInterp;                   // 0x0378 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                            InteriorLPFInterp;                    // 0x037C (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                            ExteriorVolumeInterp;                   // 0x0380 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                            ExteriorLPFInterp;                    // 0x0384 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
class UAudioComponent*                       TestAudioComponent;                   // 0x0388
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
struct FPointer                       TextToSpeech;                        // 0x0390 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                           DebugState;                        // 0x0398 (0x0001)
[0x0000000000001002] (CPF_Const | CPF_Native)
float                            TransientMasterVolume;                   // 0x039C (0x0004)
[0x0000000000002000] (CPF_Transient)
float                            LastUpdateTime;                      // 0x03A0 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AudioDevice");
}

return uClassPointer;
};

class USoundClass* FindSoundClass(struct FName SoundClassName);
bool SetSoundMode(struct FName NewModeGroup, struct FName NewModeValue);
};

// Class Engine.SoundClass
// 0x0088 (0x0060 - 0x00E8)
class USoundClass : public UObject
{
public:
struct FSoundClassProperties                  Properties;                              // 0x0060 (0x0020)
[0x0000000000000001] (CPF_Edit)
TArray<struct FName>                          ChildClassNames;                         // 0x0080 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
unsigned long                                 bIsChild : 1;                            // 0x0090 (0x0004)
[0x0000000000000000] [0x00000001]
int32_t                                       MenuID;                                  // 0x0094 (0x0004)
[0x0000000800000000]
uint8_t                                       UnknownData00[0x50];                     // 0x0098 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.SoundClass.EditorData

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundClass");
}

return uClassPointer;
};

};

// Class Engine.SoundMode
// 0x0050 (0x0060 - 0x00B0)
class USoundMode : public UObject
{
public:
unsigned long                                 bApplyEQ : 1;                            // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FAudioEQEffect                         EQSettings;                              // 0x0068 (0x0028)
[0x0000000000000001] (CPF_Edit)
TArray<struct FSoundClassAdjuster>            SoundClassEffects;                       // 0x0090
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                         InitialDelay;                            // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                         FadeInTime;                              // 0x00A4 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                          Duration;                     // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FadeOutTime;                     // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SoundMode");
}

return uClassPointer;
};

};

// Class Engine.Pawn
// 0x02AC (0x0268 - 0x0514)
class APawn : public AActor
{
public:
struct FPointer                          VfTable_IInterface_Speaker;               // 0x0268 (0x0008)
[0x0000000000801002] (CPF_Const | CPF_Native | CPF_NoExport)
class UObjectProvider*                    ObjectProvider;                     // 0x0270 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UGroupComponent_ORS*                   RegistryGroup;                         // 0x0278
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                          MaxStepHeight;                     // 0x0280 (0x0004)
[0x0000000000000002] (CPF_Const)
float                          MaxJumpHeight;                      // 0x0284 (0x0004)
[0x0000000000000002] (CPF_Const)
float                          WalkableFloorZ;                     // 0x0288 (0x0004)
[0x0000000000000002] (CPF_Const)
float                          LedgeCheckThreshold;                  // 0x028C (0x0004)
[0x0000000000000002] (CPF_Const)
struct FVector                    PartialLedgeMoveDir;                   // 0x0290 (0x000C)
[0x0000000000000002] (CPF_Const)
class AController*                   Controller;                       // 0x02A0 (0x0008)
[0x0000000104000020] (CPF_Net | CPF_EditInline)
class APawn*                      NextPawn;                          // 0x02A8 (0x0008)
[0x0000000000000002] (CPF_Const)
float                          NetRelevancyTime;                    // 0x02B0 (0x0004)
[0x0000000000000000]
class APlayerController*                LastRealViewer;                        // 0x02B8 (0x0008)
[0x0000000000000000]
class AActor*                    LastViewer;                        // 0x02C0 (0x0008)
[0x0000000000000000]
unsigned long                     bScriptTickSpecial : 1;                 // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000001]
```

```
unsigned long                          bUpAndOut : 1;                          // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                          bIsWalking : 1;                        // 0x02C8 (0x0004)
[0x0000000000000020] [0x00000004] (CPF_Net)
unsigned long                          bWantsToCrouch : 1;                     // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                          bIsCrouched : 1;                        // 0x02C8 (0x0004)
[0x0000000000000022] [0x00000010] (CPF_Const | CPF_Net)
unsigned long                          bTryToUncrouch : 1;                     // 0x02C8 (0x0004)
[0x0000000000000002] [0x00000020] (CPF_Const)
unsigned long                          bCanCrouch : 1;                         // 0x02C8 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                          bCrawler : 1;                          // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                          bReducedSpeed : 1;                      // 0x02C8 (0x0004)
[0x0000000000000002] [0x00000100] (CPF_Const)
unsigned long                          bJumpCapable : 1;                       // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                          bCanJump : 1;                          // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                          bCanWalk : 1;                          // 0x02C8 (0x0004)
[0x0000000000000000] [0x00000800]
unsigned long                          bCanFly : 1;                           // 0x02C8 (0x0004)
[0x0000000000000000] [0x00001000]
unsigned long                          bCanStrafe : 1;                         // 0x02C8 (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                          bAvoidLedges : 1;                       // 0x02C8 (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                          bStopAtLedges : 1;                      // 0x02C8 (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                          bAllowLedgeOverhang : 1;                    // 0x02C8 (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                          bPartiallyOverLedge : 1;                   // 0x02C8 (0x0004)
[0x0000000000000002] [0x00020000] (CPF_Const)
unsigned long                          bSimulateGravity : 1;                    // 0x02C8 (0x0004)
[0x0000000000000022] [0x00040000] (CPF_Const | CPF_Net)
unsigned long                          bIgnoreForces : 1;                      // 0x02C8 (0x0004)
[0x0000000000000000] [0x00080000]
unsigned long                          bCanWalkOffLedges : 1;                    // 0x02C8 (0x0004)
[0x0000000000000000] [0x00100000]
unsigned long                          bCanBeBaseForPawns : 1;                    // 0x02C8 (0x0004)
[0x0000000000000000] [0x00200000]
unsigned long                          bSimGravityDisabled : 1;                   // 0x02C8 (0x0004)
[0x0000000000000002] [0x00400000] (CPF_Const)
unsigned long                          bDirectHitWall : 1;                      // 0x02C8 (0x0004)
[0x0000000000000000] [0x00800000]
unsigned long                          bPushesRigidBodies : 1;                   // 0x02C8 (0x0004)
[0x0000000000000002] [0x01000000] (CPF_Const)
unsigned long                          bForceFloorCheck : 1;                    // 0x02C8 (0x0004)
[0x0000000000000000] [0x02000000]
unsigned long                          bForceKeepAnchor : 1;                    // 0x02C8 (0x0004)
[0x0000000000000000] [0x04000000]
unsigned long                          bCanMantle : 1;                         // 0x02C8 (0x0004)
[0x0000000000000000] [0x08000000]
```

```
unsigned long                          bCanClimbUp : 1;                         // 0x02C8 (0x0004)
[0x0000000000000000] [0x10000000]
unsigned long                          bCanClimbCeilings : 1;                   // 0x02C8 (0x0004)
[0x0000000000000000] [0x20000000]
unsigned long                          bCanSwatTurn : 1;                        // 0x02C8 (0x0004)
[0x0000000000000020] [0x40000000] (CPF_Net)
unsigned long                          bCanLeap : 1;                            // 0x02C8 (0x0004)
[0x0000000000000000] [0x80000000]
unsigned long                          bCanCoverSlip : 1;                       // 0x02CC (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                          bDisplayPathErrors : 1;                  // 0x02CC (0x0004)
[0x0000000000044000] [0x00000002] (CPF_Config | CPF_GlobalConfig)
unsigned long                          bAmbientCreature : 1;                    // 0x02CC (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                          bLOSHearing : 1;                         // 0x02CC (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                          bMuffledHearing : 1;                     // 0x02CC (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                          bDontPossess : 1;                        // 0x02CC (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                          bRollToDesired : 1;                      // 0x02CC (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                          bStationary : 1;                         // 0x02CC (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                          bCachedRelevant : 1;                     // 0x02CC (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                          bModifyReachSpecCost : 1;                // 0x02CC (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                          bModifyNavPointDest : 1;                 // 0x02CC (0x0004)
[0x0000000000000000] [0x00000400]
unsigned long                          bPrevBypassSimulatedClientPhysics : 1;        // 0x02CC
(0x0004) [0x0000000000000000] [0x00000800]
unsigned long                          bRunPhysicsWithNoController : 1;              // 0x02CC
(0x0004) [0x0000000000000000] [0x00001000]
unsigned long                          bForceMaxAccel : 1;                      // 0x02CC (0x0004)
[0x0000000000000000] [0x00002000]
unsigned long                          bLimitFallAccel : 1;                     // 0x02CC (0x0004)
[0x0000000000000000] [0x00004000]
unsigned long                          bForceRMVelocity : 1;                    // 0x02CC (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                          bForceRegularVelocity : 1;               // 0x02CC (0x0004)
[0x0000000000000000] [0x00010000]
unsigned long                          bDesiredRotationSet : 1;                 // 0x02CC (0x0004)
[0x0000000000000002] [0x00020000] (CPF_Const)
unsigned long                          bLockDesiredRotation : 1;                // 0x02CC (0x0004)
[0x0000000000000002] [0x00040000] (CPF_Const)
unsigned long                          bUnlockWhenReached : 1;                  // 0x02CC (0x0004)
[0x0000000000000002] [0x00080000] (CPF_Const)
unsigned long                          bNeedsBaseTickedFirst : 1;               // 0x02CC (0x0004)
[0x0000000000000000] [0x00100000]
unsigned long                          bUsedByMatinee : 1;                      // 0x02CC (0x0004)
[0x0000000100002020] [0x00200000] (CPF_Net | CPF_Transient)
unsigned long                          bRootMotionFromInterpCurve : 1;               // 0x02CC
(0x0004) [0x0000000000000020] [0x00400000] (CPF_Net)
```

```
unsigned long                    bDebugShowCameraLocation : 1;          // 0x02CC
(0x0004) [0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                    bFastAttachedMove : 1;                 // 0x02CC (0x0004)
[0x0000000000000021] [0x01000000] (CPF_Edit | CPF_Net)
uint8_t                      WalkingPhysics;                      // 0x02D0 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                      PathSearchType;                      // 0x02D1 (0x0001)
[0x0000000000000000]
uint8_t                      RemoteViewPitch;                     // 0x02D2 (0x0001)
[0x0000000000000022] (CPF_Const | CPF_Net)
float                        UncrouchTime;                        // 0x02D4 (0x0004)
[0x0000000000000002] (CPF_Const)
float                        CrouchHeight;                        // 0x02D8 (0x0004)
[0x0000000000000000]
float                        CrouchRadius;                        // 0x02DC (0x0004)
[0x0000000000000000]
int32_t                      FullHeight;                          // 0x02E0 (0x0004)
[0x0000000000000002] (CPF_Const)
class UPathConstraint*               PathConstraintList;                 // 0x02E8 (0x0008)
[0x0000000000000000]
class UPathGoalEvaluator*              PathGoalList;                     // 0x02F0 (0x0008)
[0x0000000000000000]
float                        DesiredSpeed;                        // 0x02F8 (0x0004)
[0x0000000000000000]
float                        MaxDesiredSpeed;                     // 0x02FC (0x0004)
[0x0000000000000000]
float                        HearingThreshold;                    // 0x0300 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        Alertness;                           // 0x0304 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        SightRadius;                         // 0x0308 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        PeripheralVision;                    // 0x030C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        AvgPhysicsTime;                      // 0x0310 (0x0004)
[0x0000000000000002] (CPF_Const)
float                        Mass;                                // 0x0314 (0x0004)
[0x0000000000000000]
float                        Buoyancy;                            // 0x0318 (0x0004)
[0x0000000000000000]
float                        MeleeRange;                          // 0x031C (0x0004)
[0x0000000000000000]
class ANavigationPoint*              Anchor;                             // 0x0320 (0x0008)
[0x0000000000000002] (CPF_Const)
int32_t                      AnchorItem;                          // 0x0328 (0x0004)
[0x0000000000000002] (CPF_Const)
class ANavigationPoint*              LastAnchor;                         // 0x0330 (0x0008)
[0x0000000000000002] (CPF_Const)
float                        FindAnchorFailedTime;                // 0x0338 (0x0004)
[0x0000000000000000]
float                        LastValidAnchorTime;                 // 0x033C (0x0004)
[0x0000000000000000]
float                        DestinationOffset;                   // 0x0340 (0x0004)
[0x0000000000000000]
```

```
float                          NextPathRadius;                  // 0x0344 (0x0004)
[0x0000000000000000]
struct FVector                      SerpentineDir;                   // 0x0348 (0x000C)
[0x0000000000000000]
float                          SerpentineDist;                  // 0x0354 (0x0004)
[0x0000000000000000]
float                          SerpentineTime;                  // 0x0358 (0x0004)
[0x0000000000000000]
int32_t                         MaxPitchLimit;                  // 0x035C (0x0004)
[0x0000000000000000]
float                          GroundSpeed;                     // 0x0360 (0x0004)
[0x0000000000000020] (CPF_Net)
float                          AirSpeed;                      // 0x0364 (0x0004)
[0x0000000000000020] (CPF_Net)
float                          AccelRate;                     // 0x0368 (0x0004)
[0x0000000000000020] (CPF_Net)
float                          JumpZ;                         // 0x036C (0x0004)
[0x0000000000000020] (CPF_Net)
float                          OutofWaterZ;                     // 0x0370 (0x0004)
[0x0000000000000000]
float                          MaxOutOfWaterStepHeight;                 // 0x0374 (0x0004)
[0x0000000000000000]
float                          AirControl;                    // 0x0378 (0x0004)
[0x0000000000000020] (CPF_Net)
float                          WalkingPct;                     // 0x037C (0x0004)
[0x0000000000000000]
float                          MovementSpeedModifier;                 // 0x0380 (0x0004)
[0x0000000000000000]
float                          CrouchedPct;                     // 0x0384 (0x0004)
[0x0000000000000000]
float                          MaxFallSpeed;                    // 0x0388 (0x0004)
[0x0000000000000000]
float                          AIMaxFallSpeedFactor;                // 0x038C (0x0004)
[0x0000000000000000]
float                          BaseEyeHeight;                   // 0x0390 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          EyeHeight;                     // 0x0394 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                      Floor;                      // 0x0398 (0x000C)
[0x0000000000000000]
struct FVector                      RMVelocity;                    // 0x03A4 (0x000C)
[0x0000000000000000]
struct FVector                      noise1spot;                    // 0x03B0 (0x000C)
[0x0000000000000002] (CPF_Const)
float                          noise1time;                     // 0x03BC (0x0004)
[0x0000000000000002] (CPF_Const)
class APawn*                         noise1other;                   // 0x03C0 (0x0008)
[0x0000000000000002] (CPF_Const)
float                          noise1loudness;                  // 0x03C8 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FVector                      noise2spot;                    // 0x03CC (0x000C)
[0x0000000000000002] (CPF_Const)
float                          noise2time;                     // 0x03D8 (0x0004)
[0x0000000000000002] (CPF_Const)
```

```
class APawn*                    noise2other;                    // 0x03E0 (0x0008)
[0x0000000000000002] (CPF_Const)
float                    noise2loudness;                    // 0x03E8 (0x0004)
[0x0000000000000002] (CPF_Const)
float                    SoundDampening;                    // 0x03EC (0x0004)
[0x0000000000000000]
float                    DamageScaling;                    // 0x03F0 (0x0004)
[0x0000000000000000]
class FString                    MenuName;                    // 0x03F8 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class UClass*                    ControllerClass;                    // 0x0408 (0x0008)
[0x0000000000000000]
class APlayerReplicationInfo*                    PlayerReplicationInfo;                    // 0x0410
(0x0008) [0x0000000104000020] (CPF_Net | CPF_EditInline)
struct FName                    LandMovementState;                    // 0x0418 (0x0008)
[0x0000000000000000]
struct FName                    WaterMovementState;                    // 0x0420 (0x0008)
[0x0000000000000000]
class APlayerStart*                    LastStartSpot;                    // 0x0428 (0x0008)
[0x0000000000000000]
float                    LastStartTime;                    // 0x0430 (0x0004)
[0x0000000000000000]
class USkeletalMeshComponent*                    Mesh;                    // 0x0438 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)
class UCylinderComponent*                    CylinderComponent;                    // 0x0440
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
float                    RBPushRadius;                    // 0x0448 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                    RBPushStrength;                    // 0x044C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                    AlwaysRelevantDistanceSquared;                    // 0x0450 (0x0004)
[0x0000000000000000]
class AController*                    LastHitBy;                    // 0x0458 (0x0008)
[0x0000000000000000]
float                    ViewPitchMin;                    // 0x0460 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                    ViewPitchMax;                    // 0x0464 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                    AllowedYawError;                    // 0x0468 (0x0004)
[0x0000000000000000]
struct FRotator                    DesiredRotation;                    // 0x046C (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UPrimitiveComponent*                    PreRagdollCollisionComponent;                    // 0x0478
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class URB_BodyInstance*                    PhysicsPushBody;                    // 0x0480 (0x0008)
[0x0000000000000000]
int32_t                    FailedLandingCount;                    // 0x0488 (0x0004)
[0x0000000000000000]
TArray<class UAnimNodeSlot*>                    SlotNodes;                    // 0x0490 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<class UInterpGroup*>                    InterpGroupList;                    // 0x04A0 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
class UAudioComponent*                    FacialAudioComp;                    // 0x04B0
(0x0008) [0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)
class UMaterialInstanceConstant*                MIC_PawnMat;                              // 0x04B8
(0x0008) [0x0000000000002000] (CPF_Transient)
class UMaterialInstanceConstant*                MIC_PawnHair;                             // 0x04C0
(0x0008) [0x0000000000002000] (CPF_Transient)
TArray<struct FScalarParameterInterpStruct>        ScalarParameterInterpArray;                   //
0x04C8 (0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
struct FRootMotionCurve                         RootMotionInterpCurve;                    // 0x04D8
(0x0028) [0x0000000000400000] (CPF_NeedCtorLink)
float                                   RootMotionInterpRate;                   // 0x0500 (0x0004)
[0x0000000000000020] (CPF_Net)
float                                   RootMotionInterpCurrentTime;                // 0x0504 (0x0004)
[0x0000000000000020] (CPF_Net)
struct FVector                              RootMotionInterpCurveLastValue;              // 0x0508
(0x000C) [0x0000000000000020] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Pawn");
}

return uClassPointer;
};

void OnSetVelocity(class USeqAct_SetVelocity* Action);
void eventSpeak(class USoundCue* Cue);
void SetScalarParameterInterp(struct FScalarParameterInterpStruct& ScalarParameterInterp);
void SetRootMotionInterpCurrentTime(float inTime, float DeltaTime, unsigned long
bUpdateSkelPose);
void SetCinematicMode(unsigned long bInCinematicMode);
void ZeroMovementVariables();
void ClearPathStep();
void DrawPathStep(class UCanvas* C);
void IncrementPathChild(int32_t Cnt, class UCanvas* C);
void IncrementPathStep(int32_t Cnt, class UCanvas* C);
class UPathGoalEvaluator* CreatePathGoalEvaluator(class UClass* GoalEvalClass);
class UPathConstraint* CreatePathConstraint(class UClass* ConstraintClass);
void AddGoalEvaluator(class UPathGoalEvaluator* Evaluator);
void AddPathConstraint(class UPathConstraint* Constraint);
void ClearConstraints();
void eventSoakPause();
void eventBecomeViewTarget(class APlayerController* PC);
void eventMessagePlayer(class FString msg);
bool HandleTeleport(TArray<class UObject*> DestList, unsigned long bUpdateRotation, unsigned
long bCheckOverlap, float TeleportDistance, TArray<class AVolume*> TeleportVolumes, int32_t
PreferredDestIndex);
void OnTeleport(class USeqAct_Teleport* Action);
void OnSetMaterial(class USeqAct_SetMaterial* Action);
float GetDamageScaling();
```

```
void DoKismetAttachment(class AActor* Attachment, class USeqAct_AttachToActor* Action);
void eventSpawnedByKismet();
bool IsStationary();
struct FVector GetCollisionExtent();
float GetCollisionHeight();
float GetCollisionRadius();
bool CheatFly();
bool CheatGhost();
bool CheatWalk();
void DrawHUD(class AHUD* H);
void PlayLanded(float ImpactVel);
bool CannotJumpNow();
void eventPlayFootStepSound(int32_t FootDown);
void SetDyingPhysics();
void TurnOff();
bool DoJump(unsigned long bUpdating);
void eventTickSpecial(float DeltaTime);
void eventLanded(struct FVector HitNormal, class AActor* FloorActor);
void eventFalling();
bool eventIsSameTeam(class APawn* Other);
class ATeamInfo* GetTeam();
uint8_t GetTeamNum();
void SetMovementPhysics();
void OnAssignController(class USeqAct_AssignController* inAction);
void eventReceivedNewEvent(class USequenceEvent* Evt);
void SpawnDefaultController();
void eventPostBeginPlay();
void eventPreBeginPlay();
void eventDestroyed();
void DetachFromController(unsigned long bDestroyController);
bool CanBeBaseForPawn(class APawn* aPawn);
void eventBaseChange();
void eventStuckOnPawn(class APawn* OtherPawn);
void JumpOffPawn();
bool eventEncroachingOn(class AActor* Other);
void FaceRotation(struct FRotator NewRotation, float DeltaTime);
void eventUpdatePawnRotation(struct FRotator NewRotation);
void ClientSetRotation(struct FRotator NewRotation);
void ClientRestart();
void Restart();
void eventStartCrouch(float HeightAdjust);
void eventEndCrouch(float HeightAdjust);
void ShouldCrouch(unsigned long bCrouch);
void UnCrouch();
void eventOutsideWorldBounds();
void eventFellOutOfWorld();
void eventClientMessage(class FString S, struct FName Type);
bool LineOfSightTo(class AActor* Other);
void SetMoveTarget(class AActor* NewTarget);
bool InGodMode();
void SetViewRotation(struct FRotator NewRotation);
bool eventInFreeCam();
struct FRotator eventGetBaseAimRotation();
struct FVector eventGetPawnViewLocation();
struct FRotator eventGetViewRotation();
void eventGetActorEyesViewPoint(struct FVector& out_Location, struct FRotator& out_Rotation);
void ProcessViewRotation(float DeltaTime, struct FRotator& out_ViewRotation, struct FRotator& out_DeltaRot);
bool IsFirstPerson();
bool IsLocalHuman();
bool IsPlayerPawn();
bool IsLocallyControlled();
bool IsHumanControlled(class AController* PawnController);
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void eventSetWalking(unsigned long bNewIsWalking);
float RangedAttackTime();
bool RecommendLongRangedAttack();
void DropToGround();
struct FName GetDefaultCameraMode(class APlayerController* RequestedBy);
void UnPossessed();
void UpdateControllerOnPossess();
void UpdateObjectProviderParent();
void PossessedBy(class AController* C);
void PlayTeleportEffect(unsigned long bOut, unsigned long bSound);
class FString GetHumanReadableName();
void Reset();
void SetBaseEyeheight();
bool eventSpecialMoveThruEdge(uint8_t EdgeType, int32_t Dir, struct FVector MoveStart, struct FVector MoveDest, class AActor* RelActor, int32_t RelItem, class UNavigationHandle* NavHandle);
bool SpecialMoveTo(class ANavigationPoint* Start, class ANavigationPoint* End, class AActor* Next);
bool TermRagdoll();
bool InitRagdoll();
void GetBoundingCylinder(float& CollisionRadius, float& CollisionHeight);
bool ReachedDesiredRotation();
void SetPushesRigidBodies(unsigned long NewPush);
void ForceCrouch();
bool ReachedPoint(struct FVector Point, class AActor* NewAnchor);
bool ReachedDestination(class AActor* Goal);
class ANavigationPoint* GetBestAnchor(class AActor* TestActor, struct FVector TestLocation, unsigned long bStartPoint, unsigned long bOnlyCheckVisible, float& out_Dist);
void SetAnchor(class ANavigationPoint* NewAnchor);
void SetRemoteViewPitch(int32_t NewRemoteViewPitch);
bool IsInvisible();
bool IsValidEnemyTargetFor(class APlayerReplicationInfo* PRI, unsigned long bNoPRIisEnemy);
float GetFallDuration();
bool SuggestJumpVelocity(struct FVector Destination, struct FVector Start, unsigned long bRequireFallLanding, struct FVector& JumpVelocity);
bool ValidAnchor();
struct FVector AdjustDestination(class AActor* GoalActor, struct FVector Dest);
void eventReplicatedEvent(struct FName VarName);
void eventSetSkelControlScale(struct FName SkelControlName, float Scale);
void eventSetMorphWeight(struct FName MorphNodeName, float MorphWeight);
class UFaceFXAsset* eventGetActorFaceFXAsset();
void FaceFXAudioFinished(class UAudioComponent* AC);
void OnPlayFaceFXAnim(class USeqAct_PlayFaceFXAnim* inAction);
bool CanActorPlayFaceFXAnim();
bool IsActorPlayingFaceFXAnim();
class UAudioComponent* eventGetFaceFXAudioComponent();
void eventStopActorFaceFXAnim();
bool eventPlayActorFaceFXAnim(class UFaceFXAnimSet* AnimSet, class FString GroupName,
class FString SeqName, class USoundCue* SoundCueToPlay, class UAkEvent* AkEventToPlay);
void eventMAT_FinishAIGroup();
void eventMAT_BeginAIGroup(struct FVector StartLoc, struct FRotator StartRot);
void FinishAIGroup();
void BeginAIGroup();
void eventInterpolationFinished(class USeqAct_Interp* InterpAction);
void eventInterpolationStarted(class USeqAct_Interp* InterpAction, class UInterpGroupInst*
GroupInst);
void MAT_SetSkelControlStrength(struct FName SkelControlName, float ControlStrength);
void MAT_SetSkelControlScale(struct FName SkelControlName, float Scale);
void MAT_SetMorphWeight(struct FName MorphNodeName, float MorphWeight);
void MAT_SetAnimWeights(TArray<struct FAnimSlotInfo> SlotInfos);
void MAT_SetAnimPosition(struct FName SlotName, int32_t ChannelIndex, struct FName
InAnimSeqName, float InPosition, unsigned long bFireNotifies, unsigned long bLooping, unsigned
long bEnableRootMotion);
void eventSetAnimPosition(struct FName SlotName, int32_t ChannelIndex, struct FName
InAnimSeqName, float InPosition, unsigned long bFireNotifies, unsigned long bLooping, unsigned
long bEnableRootMotion);
void MAT_FinishAnimControl(class UInterpGroup* InInterpGroup);
void eventFinishAnimControl(class UInterpGroup* InInterpGroup);
void MAT_BeginAnimControl(class UInterpGroup* InInterpGroup);
void eventBeginAnimControl(class UInterpGroup* InInterpGroup);
bool eventRestoreAnimSetsToDefault();
void eventAnimSetListUpdated();
void AddAnimSets(TArray<class UAnimSet*>& CustomAnimSets);
void eventBuildScriptAnimSetList();
void UpdateAnimSetList();
void ClearAnimNodes();
void eventCacheAnimNodes();
void eventPostInitAnimTree(class USkeletalMeshComponent* SkelComp);
bool IsDesiredRotationLocked();
bool IsDesiredRotationInUse();
void CheckDesiredRotation();
void ResetDesiredRotation();
void LockDesiredRotation(unsigned long Lock, unsigned long InUnlockWhenReached);
bool SetDesiredRotation(struct FRotator TargetDesiredRotation, unsigned long
InLockDesiredRotation, unsigned long InUnlockWhenReached, float InterpolationTime, unsigned
long bResetRotationRate);
bool PickWallAdjust(struct FVector WallHitNormal, class AActor* HitActor);
void eventConstruct();
};

// Class Engine.MatineePawn
// 0x000C (0x0514 - 0x0520)
class AMatineePawn : public APawn
{
public:
class USkeletalMesh*                PreviewMesh;                    // 0x0518 (0x0008)
[0x0000000800000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MatineePawn");
}

return uClassPointer;
};

};

// Class Engine.Scout
// 0x009C (0x0514 - 0x05B0)
class AScout : public APawn
{
public:
TArray<struct FPathSizeInfo>            PathSizes;                          // 0x0518 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                          TestJumpZ;                          // 0x0528 (0x0004)
[0x0000000000000000]
float                          TestGroundSpeed;                    // 0x052C (0x0004)
[0x0000000000000000]
float                          TestMaxFallSpeed;                   // 0x0530 (0x0004)
[0x0000000000000000]
float                          TestFallSpeed;                      // 0x0534 (0x0004)
[0x0000000000000000]
float                          MaxLandingVelocity;                 // 0x0538 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         MinNumPlayerStarts;                 // 0x053C (0x0004)
[0x0000000000000000]
class UClass*                       DefaultReachSpecClass;              // 0x0540 (0x0008)
[0x0000000000000000]
TArray<struct FColor>                  EdgePathColors;                     // 0x0548 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                          NavMeshGen_StepSize;                // 0x0558 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_EntityHalfHeight;        // 0x055C (0x0004)
[0x0000000000000000]
float                          NavMeshGen_StartingHeightOffset;    // 0x0560 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_MaxDropHeight;           // 0x0564 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_MaxStepHeight;           // 0x0568 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_VertZDeltaSnapThresh;    // 0x056C (0x0004)
[0x0000000000000000]
float                          NavMeshGen_MinPolyArea;             // 0x0570 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_BorderBackfill_CheckDist;  // 0x0574 (0x0004)
[0x0000000000000000]
```

```
float                          NavMeshGen_MinMergeDotAreaThreshold;       // 0x0578
(0x0004) [0x0000000000000000]
float                          NavMeshGen_MinMergeDotSmallArea;           // 0x057C
(0x0004) [0x0000000000000000]
float                          NavMeshGen_MinMergeDotLargeArea;           // 0x0580 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_MaxPolyHeight;                  // 0x0584 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_HeightMergeThreshold;           // 0x0588 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_EdgeMaxDelta;                   // 0x058C (0x0004)
[0x0000000000000000]
float                          NavMeshGen_MaxGroundCheckSize;             // 0x0590 (0x0004)
[0x0000000000000000]
float                          NavMeshGen_MinEdgeLength;                  // 0x0594 (0x0004)
[0x0000000000000000]
unsigned long                       NavMeshGen_ExpansionDoObstacleMeshSimplification : 1;//
0x0598 (0x0004) [0x0000000000000000] [0x00000001]
unsigned long                       bHightlightOneWayReachSpecs : 1;          // 0x0598
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
float                          MinMantleFallDist;                         // 0x059C (0x0004)
[0x0000000000000000]
float                          MaxMantleFallDist;                         // 0x05A0 (0x0004)
[0x0000000000000000]
float                          MinMantleLateralDist;                      // 0x05A4 (0x0004)
[0x0000000000000000]
float                          MaxMantleLateralDist;                      // 0x05A8 (0x0004)
[0x0000000000000000]
float                          MaxMantleFallTime;                         // 0x05AC (0x0004)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Scout");
}

return uClassPointer;
};

void eventPreBeginPlay();
};

// Class Engine.Light
// 0x000C (0x0268 - 0x0274)
class ALight : public AActor
{
public:
class ULightComponent*              LightComponent;                       // 0x0268 (0x0008)
[0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                              bEnabled : 1;                              // 0x0270 (0x0004)
[0x0000000100000020] [0x00000001] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Light");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
};

// Class Engine.DirectionalLight
// 0x0004 (0x0274 - 0x0278)
class ADirectionalLight : public ALight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DirectionalLight");
}

return uClassPointer;
};

};

// Class Engine.DirectionalLightToggleable
// 0x0000 (0x0278 - 0x0278)
class ADirectionalLightToggleable : public ADirectionalLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.DirectionalLightToggleable");
}

return uClassPointer;
};

};

// Class Engine.DominantDirectionalLight
// 0x0000 (0x0278 - 0x0278)
class ADominantDirectionalLight : public ADirectionalLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DominantDirectionalLight");
}

return uClassPointer;
};

};

// Class Engine.DominantDirectionalLightMovable
// 0x0000 (0x0278 - 0x0278)
class ADominantDirectionalLightMovable : public ADominantDirectionalLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DominantDirectionalLightMovable");
}

return uClassPointer;
};

};

// Class Engine.PointLight
// 0x0004 (0x0274 - 0x0278)
class APointLight : public ALight
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PointLight");
}

return uClassPointer;
};

};

// Class Engine.DominantPointLight
// 0x0000 (0x0278 - 0x0278)
class ADominantPointLight : public APointLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DominantPointLight");
}

return uClassPointer;
};

};

// Class Engine.PointLightMovable
// 0x0000 (0x0278 - 0x0278)
class APointLightMovable : public APointLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PointLightMovable");
}
return uClassPointer;
};

};

// Class Engine.PointLightToggleable
// 0x0000 (0x0278 - 0x0278)
class APointLightToggleable : public APointLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PointLightToggleable");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct APointLightToggleable_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct APointLightToggleable_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
};

// Class Engine.SkyLight
// 0x0004 (0x0274 - 0x0278)
class ASkyLight : public ALight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkyLight");
}

return uClassPointer;
};

};

// Class Engine.SkyLightToggleable
// 0x0000 (0x0278 - 0x0278)
class ASkyLightToggleable : public ASkyLight
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkyLightToggleable");
}

return uClassPointer;
};

};

// Class Engine.SpotLight
// 0x0004 (0x0274 - 0x0278)
class ASpotLight : public ALight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpotLight");
}

return uClassPointer;
};

};

// Class Engine.DominantSpotLight
// 0x0000 (0x0278 - 0x0278)
class ADominantSpotLight : public ASpotLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DominantSpotLight");
}
return uClassPointer;
};

};

// Class Engine.GeneratedMeshAreaLight
// 0x0000 (0x0278 - 0x0278)
class AGeneratedMeshAreaLight : public ASpotLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GeneratedMeshAreaLight");
}

return uClassPointer;
};

};

// Class Engine.SpotLightMovable
// 0x0000 (0x0278 - 0x0278)
class ASpotLightMovable : public ASpotLight
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpotLightMovable");
}

return uClassPointer;
};

};

// Class Engine.SpotLightToggleable
// 0x0000 (0x0278 - 0x0278)
class ASpotLightToggleable : public ASpotLight
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpotLightToggleable");
}

return uClassPointer;
};

void ApplyCheckpointRecord(struct ASpotLightToggleable_FCheckpointRecord& Record);
void CreateCheckpointRecord(struct ASpotLightToggleable_FCheckpointRecord& Record);
bool ShouldSaveForCheckpoint();
};

// Class Engine.StaticLightCollectionActor
// 0x0018 (0x0274 - 0x028C)
class AStaticLightCollectionActor : public ALight
{
public:
TArray<class ULightComponent*>                LightComponents;                         // 0x0278
(0x0010) [0x000000000448000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
int32_t                                       MaxLightComponents;                      // 0x0288 (0x0004)
[0x0000000000004000] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticLightCollectionActor");
}

return uClassPointer;
};

};

// Class Engine.LightComponent
// 0x0127 (0x009D - 0x01C4)
class ULightComponent : public UActorComponent
{
public:
struct FPointer                               SceneInfo;                               // 0x00A0 (0x0008)
[0x0000000001003002] (CPF_Const | CPF_Native | CPF_Transient)
uint8_t                                       UnknownData00[0x8];                      // 0x00A8 (0x0008) MISSED
OFFSET
struct FMatrix                                WorldToLight;                            // 0x00B0 (0x0040)
```

```
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FMatrix                          LightToWorld;                       // 0x00F0 (0x0040)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FGuid                            LightGuid;                          // 0x0130 (0x0010)
[0x0000000000200002] (CPF_Const)
struct FGuid                            LightmapGuid;                       // 0x0140 (0x0010)
[0x0000000000200002] (CPF_Const)
float                                   Brightness;                         // 0x0150 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
struct FColor                           LightColor;                         // 0x0154 (0x0004)
[0x0000000200000003] (CPF_Edit | CPF_Const)
class ULightFunction*                   Function;                           // 0x0158 (0x0008)
[0x000000000440000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_NeedCtorLink |
CPF_EditInline)
unsigned long                           bEnabled : 1;                       // 0x0160 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                           CastShadows : 1;                    // 0x0160 (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                           CastStaticShadows : 1;              // 0x0160 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                           CastDynamicShadows : 1;             // 0x0160 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           bCastCompositeShadow : 1;           // 0x0160 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                           bAffectCompositeShadowDirection : 1;        // 0x0160
(0x0004) [0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                           bNonModulatedSelfShadowing : 1;             // 0x0160
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                           bSelfShadowOnly : 1;                // 0x0160 (0x0004)
[0x0000000200000001] [0x00000080] (CPF_Edit)
unsigned long                           bAllowPreShadow : 1;                // 0x0160 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                           bForceDynamicLight : 1;             // 0x0160 (0x0004)
[0x0000000000000002] [0x00000200] (CPF_Const)
unsigned long                           UseDirectLightMap : 1;              // 0x0160 (0x0004)
[0x0000000000000002] [0x00000400] (CPF_Const)
unsigned long                           bHasLightEverBeenBuiltIntoLightMap : 1;     // 0x0160
(0x0004) [0x0000000000000002] [0x00000800] (CPF_Const)
unsigned long                           bCanAffectDynamicPrimitivesOutsideDynamicChannel : 1;//
0x0160 (0x0004) [0x0000000000000002] [0x00001000] (CPF_Const)
unsigned long                           bRenderLightShafts : 1;             // 0x0160 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                           bUseImageReflectionSpecular : 1;            // 0x0160 (0x0004)
[0x0000000000000001] [0x00004000] (CPF_Edit)
unsigned long                           bPrecomputedLightingIsValid : 1;            // 0x0160 (0x0004)
[0x0000000000000002] [0x00008000] (CPF_Const)
unsigned long                           bExplicitlyAssignedLight : 1;               // 0x0160 (0x0004)
[0x0000000000000002] [0x00010000] (CPF_Const)
unsigned long                           bAllowCompositingIntoDLE : 1;               // 0x0160 (0x0004)
[0x0000000000000000] [0x00020000]
class ULightEnvironmentComponent*       LightEnvironment;                           // 0x0168
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
struct FLightingChannelContainer        LightingChannels;                           // 0x0170
```

```
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                         LightAffectsClassification;            // 0x0174 (0x0001)
[0x0000000000020003] (CPF_Edit | CPF_Const | CPF_EditConst)
uint8_t                         LightShadowMode;                       // 0x0175 (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor                   ModShadowColor;                  // 0x0178 (0x0010)
[0x0000000000000001] (CPF_Edit)
float                          ModShadowFadeoutTime;                   // 0x0188 (0x0004)
[0x0000000000000000]
float                          ModShadowFadeoutExponent;               // 0x018C (0x0004)
[0x0000000000000000]
int32_t                         LightListIndex;                        // 0x0190 (0x0004)
[0x0000000000201002] (CPF_Const | CPF_Native)
uint8_t                         ShadowProjectionTechnique;             // 0x0194 (0x0001)
[0x0000000000000000]
uint8_t                         ShadowFilterQuality;                   // 0x0195 (0x0001)
[0x0000000000000001] (CPF_Edit)
int32_t                         MinShadowResolution;                   // 0x0198 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         MaxShadowResolution;                   // 0x019C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                         ShadowFadeResolution;                  // 0x01A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          OcclusionDepthRange;                    // 0x01A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BloomScale;                             // 0x01A8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          BloomThreshold;                         // 0x01AC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BloomScreenBlendThreshold;              // 0x01B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                      BloomTint;                          // 0x01B4 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          RadialBlurPercent;                      // 0x01B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          OcclusionMaskDarkness;                  // 0x01BC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                          ReflectionSpecularBrightness;           // 0x01C0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightComponent");
}

return uClassPointer;
};

void OnUpdatePropertyLightColor();
void OnUpdatePropertyBrightness();
void OnUpdatePropertyOcclusionMaskDarkness();
void OnUpdatePropertyBloomTint();
void OnUpdatePropertyBloomScale();
void UpdateLightShaftParameters();
void UpdateColorAndBrightness();
struct FVector GetDirection();
struct FVector GetOrigin();
void SetLightProperties(float NewBrightness, struct FColor NewLightColor, class ULightFunction*
NewLightFunction);
void SetEnabled(unsigned long bSetEnabled);
};

// Class Engine.DirectionalLightComponent
// 0x0024 (0x01C4 - 0x01E8)
class UDirectionalLightComponent : public ULightComponent
{
public:
float                                    TraceDistance;                          // 0x01C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    WholeSceneDynamicShadowRadius;              // 0x01CC (0x0004)
[0x0000000200000001] (CPF_Edit)
int32_t                                   NumWholeSceneDynamicShadowCascades;          // 0x01D0
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
float                                    CascadeDistributionExponent;              // 0x01D4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FLightmassDirectionalLightSettings        LightmassSettings;                       // 0x01D8
(0x0010) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DirectionalLightComponent");
}

return uClassPointer;
};

void OnUpdatePropertyBrightness();
void OnUpdatePropertyLightColor();
};

// Class Engine.DominantDirectionalLightComponent
// 0x00C8 (0x01E8 - 0x02B0)
class UDominantDirectionalLightComponent : public UDirectionalLightComponent
{
public:
uint8_t                                   UnknownData00[0x8];                      // 0x01E8 (0x0008) MISSED
OFFSET
struct FDominantShadowInfo                     DominantLightShadowInfo;                 // 0x01F0
```

```
(0x00B0) [0x0000000000000002] (CPF_Const)
struct FArray_Mirror                    DominantLightShadowMap;                 // 0x02A0
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DominantDirectionalLightComponent");
}

return uClassPointer;
};

};

// Class Engine.PointLightComponent
// 0x00A0 (0x01C4 - 0x0264)
class UPointLightComponent : public ULightComponent
{
public:
float                              ShadowRadiusMultiplier;                // 0x01C8 (0x0004)
[0x0000000000000000]
float                              Radius;                                // 0x01CC (0x0004)
[0x0000000200000001] (CPF_Edit)
float                              FalloffExponent;                       // 0x01D0 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                              ShadowFalloffExponent;                 // 0x01D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              MinShadowFalloffRadius;                // 0x01D8 (0x0004)
[0x0000000000000000]
struct FMatrix                          CachedParentToWorld;                   // 0x01E0 (0x0040)
[0x0000000000000002] (CPF_Const)
struct FVector                          Translation;                           // 0x0220 (0x000C)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FPlane                           ShadowPlane;                           // 0x0230 (0x0010)
[0x0000000000000002] (CPF_Const)
class UDrawLightRadiusComponent*             PreviewLightRadius;                    // 0x0240
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
struct FLightmassPointLightSettings          LightmassSettings;                     // 0x0248
(0x0010) [0x0000000000000001] (CPF_Edit)
class UDrawLightRadiusComponent*             PreviewLightSourceRadius;              // 0x0258
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
float                              MaxShadowDistanceToCastInLightDirection;     // 0x0260
(0x0004) [0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PointLightComponent");
}

return uClassPointer;
};

void OnUpdatePropertyBrightness();
void OnUpdatePropertyLightColor();
void SetTranslation(struct FVector NewTranslation);
};

// Class Engine.DominantPointLightComponent
// 0x0004 (0x0264 - 0x0268)
class UDominantPointLightComponent : public UPointLightComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DominantPointLightComponent");
}

return uClassPointer;
};

};

// Class Engine.SpotLightComponent
// 0x0030 (0x0264 - 0x0294)
class USpotLightComponent : public UPointLightComponent
{
public:
float                                   InnerConeAngle;                         // 0x0268 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   OuterConeAngle;                         // 0x026C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   LightShaftConeAngle;                    // 0x0270 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UDrawLightConeComponent*          PreviewInnerCone;                       // 0x0278
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class UDrawLightConeComponent*          PreviewOuterCone;                       // 0x0280
(0x0008) [0x000000000408000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
struct FRotator                         Rotation;                               // 0x0288 (0x000C)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SpotLightComponent");
}

return uClassPointer;
};

void SetRotation(struct FRotator NewRotation);
};

// Class Engine.DominantSpotLightComponent
// 0x00CC (0x0294 - 0x0360)
class UDominantSpotLightComponent : public USpotLightComponent
{
public:
uint8_t                          UnknownData00[0xC];                 // 0x0294 (0x000C) MISSED
OFFSET
struct FDominantShadowInfo              DominantLightShadowInfo;              // 0x02A0
(0x00B0) [0x0000000000000002] (CPF_Const)
struct FArray_Mirror                   DominantLightShadowMap;               // 0x0350
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DominantSpotLightComponent");
}

return uClassPointer;
};

};

// Class Engine.SkyLightComponent
// 0x000C (0x01C4 - 0x01D0)
class USkyLightComponent : public ULightComponent
{
public:
float                            LowerBrightness;                   // 0x01C8 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FColor                          LowerColor;                        // 0x01CC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkyLightComponent");
}

return uClassPointer;
};

};

// Class Engine.SphericalHarmonicLightComponent
// 0x00A0 (0x01C4 - 0x0264)
class USphericalHarmonicLightComponent : public ULightComponent
{
public:
uint8_t                               UnknownData00[0xC];                   // 0x01C4 (0x000C) MISSED
OFFSET
struct FSHVectorRGB                         WorldSpaceIncidentLighting;               // 0x01D0
(0x0090) [0x0000000000000001] (CPF_Edit)
unsigned long                         bRenderBeforeModShadows : 1;               // 0x0260
(0x0004) [0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SphericalHarmonicLightComponent");
}

return uClassPointer;
};

};

// Class Engine.LightEnvironmentComponent
// 0x0023 (0x009D - 0x00C0)
class ULightEnvironmentComponent : public UActorComponent
{
public:
unsigned long                         bEnabled : 1;                         // 0x00A0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                         bForceNonCompositeDynamicLights : 1;         // 0x00A0
(0x0004) [0x0000000000000000] [0x00000002]
unsigned long                         bAllowDynamicShadowsOnTranslucency : 1;       // 0x00A0
(0x0004) [0x0000000000000000] [0x00000004]
```

```
unsigned long                                bAllowPreShadow : 1;                        // 0x00A0 (0x0004)
[0x0000000000002002] [0x00000008] (CPF_Const | CPF_Transient)
unsigned long                                bTranslucencyShadowed : 1;                  // 0x00A0 (0x0004)
[0x0000000000002002] [0x00000010] (CPF_Const | CPF_Transient)
float                                 DominantShadowFactor;                       // 0x00A4 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class ULightComponent*                        AffectingDominantLight;                     // 0x00A8
(0x0008) [0x000000000408200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_EditInline)
TArray<class UPrimitiveComponent*>                AffectedComponents;                        // 0x00B0
(0x0010) [0x000000000448200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_Component | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightEnvironmentComponent");
}

return uClassPointer;
};

bool IsEnabled();
void SetEnabled(unsigned long bNewEnabled);
};

// Class Engine.DynamicLightEnvironmentComponent
// 0x00B0 (0x00C0 - 0x0170)
class UDynamicLightEnvironmentComponent : public ULightEnvironmentComponent
{
public:
struct FPointer                              State;                                  // 0x00C0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
float                                 InvisibleUpdateTime;                        // 0x00C8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 MinTimeBetweenFullUpdates;                  // 0x00CC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 VelocityUpdateTimeScale;                    // 0x00D0 (0x0004)
[0x0000000000000000]
float                                 ShadowInterpolationSpeed;                   // 0x00D4 (0x0004)
[0x0000000000000000]
int32_t                                NumVolumeVisibilitySamples;                 // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                 LightingBoundsScale;                        // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor                             AmbientShadowColor;                         // 0x00E0 (0x0010)
[0x0000000000000000]
struct FVector                                AmbientShadowSourceDirection;               // 0x00F0
(0x000C) [0x0000000000000000]
struct FLinearColor                             AmbientGlow;                                // 0x00FC (0x0010)
```

```
[0x0000000000000000]
float                          LightDistance;                         // 0x010C (0x0004)
[0x0000000000000000]
float                          ShadowDistance;                        // 0x0110 (0x0004)
[0x0000000000000000]
unsigned long                      bCastShadows : 1;                        // 0x0114 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                      bCompositeShadowsFromDynamicLights : 1;       // 0x0114
(0x0004) [0x0000000000000000] [0x00000002]
unsigned long                      bForceCompositeAllLights : 1;              // 0x0114 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                      bAffectedBySmallDynamicLights : 1;           // 0x0114
(0x0004) [0x0000000000000000] [0x00000008]
unsigned long                      bUseBooleanEnvironmentShadowing : 1;          // 0x0114
(0x0004) [0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                      bShadowFromEnvironment : 1;                // 0x0114 (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                      bDynamic : 1;                           // 0x0114 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                      bSynthesizeDirectionalLight : 1;             // 0x0114 (0x0004)
[0x0000000000000000] [0x00000080]
unsigned long                      bSynthesizeSHLight : 1;                    // 0x0114 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                      bRequiresNonLatentUpdates : 1;              // 0x0114 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
unsigned long                      bTraceFromClosestBoundsPoint : 1;            // 0x0114
(0x0004) [0x0000000000000000] [0x00000400]
unsigned long                      bIsCharacterLightEnvironment : 1;            // 0x0114 (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                      bOverrideOwnerLightingChannels : 1;          // 0x0114
(0x0004) [0x0000000000000000] [0x00001000]
unsigned long                      bAlwaysInfluencedByDominantDirectionalLight : 1;// 0x0114
(0x0004) [0x0000000000000000] [0x00002000]
float                          ModShadowFadeoutTime;                   // 0x0118 (0x0004)
[0x0000000000000000]
float                          ModShadowFadeoutExponent;                 // 0x011C (0x0004)
[0x0000000000000000]
struct FLinearColor                     MaxModulatedShadowColor;                  // 0x0120
(0x0010) [0x0000000000000000]
float                          DominantShadowTransitionStartDistance;        // 0x0130 (0x0004)
[0x0000000000000000]
float                          DominantShadowTransitionEndDistance;          // 0x0134 (0x0004)
[0x0000000000000000]
float                          MinShadowAngle;                        // 0x0138 (0x0004)
[0x0000000000000000]
uint8_t                         BoundsMethod;                         // 0x013C (0x0001)
[0x0000000000000000]
struct FBoxSphereBounds                    OverriddenBounds;                       // 0x0140 (0x001C)
[0x0000000000000000]
struct FLightingChannelContainer                OverriddenLightingChannels;                // 0x015C
(0x0004) [0x0000000000000000]
TArray<class ULightComponent*>                OverriddenLightComponents;                // 0x0160
(0x0010) [0x000000000448000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DynamicLightEnvironmentComponent");
}

return uClassPointer;
};

struct FLinearColor GetLightIntensity();
void ResetEnvironment();
};

// Class Engine.ParticleLightEnvironmentComponent
// 0x001C (0x0170 - 0x018C)
class UParticleLightEnvironmentComponent : public UDynamicLightEnvironmentComponent
{
public:
int32_t                          ReferenceCount;                    // 0x0170 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                          NumPooledReuses;                   // 0x0174 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class AActor*                    SharedInstigator;                  // 0x0178 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
class UParticleSystem*           SharedParticleSystem;              // 0x0180 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                    bAllowDLESharing : 1;              // 0x0188 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ParticleLightEnvironmentComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawLightConeComponent
// 0x0000 (0x0268 - 0x0268)
class UDrawLightConeComponent : public UDrawConeComponent
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawLightConeComponent");
}

return uClassPointer;
};

};

// Class Engine.DrawLightRadiusComponent
// 0x0004 (0x0274 - 0x0278)
class UDrawLightRadiusComponent : public UDrawSphereComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DrawLightRadiusComponent");
}

return uClassPointer;
};

};

// Class Engine.LightFunction
// 0x0018 (0x0060 - 0x0078)
class ULightFunction : public UObject
{
public:
class UMaterialInterface*                SourceMaterial;                    // 0x0060 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FVector                     Scale;                         // 0x0068 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                              DisabledBrightness;                // 0x0074 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.LightFunction");
}

return uClassPointer;
};

};

// Class Engine.SkeletalMeshComponent
// 0x0530 (0x0280 - 0x07B0)
class USkeletalMeshComponent : public UMeshComponent
{
public:
class USkeletalMesh*                    SkeletalMesh;                       // 0x0280 (0x0008)
[0x0000000000000001] (CPF_Edit)
class USkeletalMeshComponent*               AttachedToSkelComponent;                // 0x0288
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UAnimTree*                        AnimTreeTemplate;                   // 0x0290 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UAnimNode*                        Animations;                         // 0x0298 (0x0008)
[0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
TArray<class UAnimNode*>                    AnimTickArray;                      // 0x02A0 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class UAnimNode*>                    AnimAlwaysTickArray;                // 0x02B0
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<int32_t>                         AnimTickRelevancyArray;             // 0x02C0 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<float>                           AnimTickWeightsArray;               // 0x02D0 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<class USkelControlBase*>             SkelControlTickArray;               // 0x02E0
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
class UPhysicsAsset*                    PhysicsAsset;                       // 0x02F0 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
class UPhysicsAssetInstance*                PhysicsAssetInstance;               // 0x02F8
(0x0008) [0x000000000440200A] (CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_NeedCtorLink | CPF_EditInline)
struct FPointer                         ApexClothing;                       // 0x0300 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
float                               PhysicsWeight;                      // 0x0308 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                               GlobalAnimRateScale;                // 0x030C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                               StreamingDistanceMultiplier;            // 0x0310 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FPointer                         MeshObject;                         // 0x0318 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
struct FColor                           WireframeColor;                     // 0x0320 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FBoneAtom>                    SpaceBases;                         // 0x0328 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FBoneAtom>                    LocalAtoms;                         // 0x0338 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
```

```
TArray<struct FBoneAtom>                    CachedLocalAtoms;                       // 0x0348
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FBoneAtom>                    CachedSpaceBases;                       // 0x0358
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                          LowUpdateFrameRate;                       // 0x0368 (0x0004)
[0x0000000000044002] (CPF_Const | CPF_Config | CPF_GlobalConfig)
TArray<uint8_t>                         RequiredBones;                          // 0x0370 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<uint8_t>                         ComposeOrderedRequiredBones;              // 0x0380
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
class USkeletalMeshComponent*                    ParentAnimComponent;                      // 0x0390
(0x0008) [0x000000000408000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_Component |
CPF_EditInline)
TArray<int32_t>                         ParentBoneMap;                          // 0x0398 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class UAnimSet*>                     AnimSets;                               // 0x03A8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UAnimSet*>                     TemporarySavedAnimSets;                   // 0x03B8
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<class UMorphTargetSet*>                  MorphSets;                              // 0x03C8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FActiveMorph>                  ActiveMorphs;                           // 0x03D8 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
TArray<struct FActiveMorph>                  ActiveCurveMorphs;                        // 0x03E8
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
uint8_t                          UnknownData00[0x50];                      // 0x03F8 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.SkeletalMeshComponent.MorphTargetIndexMap
TArray<struct FAttachment>                  Attachments;                            // 0x0448 (0x0010)
[0x0000000000680002] (CPF_Const | CPF_Component | CPF_NeedCtorLink)
TArray<uint8_t>                         SkelControlIndex;                       // 0x0458 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
TArray<uint8_t>                         PostPhysSkelControlIndex;               // 0x0468 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
int32_t                          ForcedLodModel;                         // 0x0478 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          MinLodModel;                          // 0x047C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          PredictedLODLevel;                      // 0x0480 (0x0004)
[0x0000000000000000]
int32_t                          OldPredictedLODLevel;                     // 0x0484 (0x0004)
[0x0000000000000000]
int32_t                          DistanceLODLevel;                       // 0x0488 (0x0004)
[0x0000000000000000]
float                           AnimationLODDistanceFactor;                 // 0x048C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          AnimationLODFrameRate;                    // 0x0490 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                           MaxDistanceFactor;                      // 0x0494 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                          ChunkIndexPreview;                       // 0x0498 (0x0004)
[0x0000000800002000] (CPF_Transient)
int32_t                          SectionIndexPreview;                      // 0x049C (0x0004)
[0x0000000800002000] (CPF_Transient)
int32_t                          bForceWireframe;                        // 0x04A0 (0x0004)
```

```
[0x0000000000000000]
int32_t                         bForceRefpose;                          // 0x04A4 (0x0004)
[0x0000000000000000]
int32_t                         bOldForceRefPose;                         // 0x04A8 (0x0004)
[0x0000000000000000]
unsigned long                       bNoSkeletonUpdate : 1;                      // 0x04AC (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
int32_t                         bDisplayBones;                          // 0x04B0 (0x0004)
[0x0000000000000000]
int32_t                         bShowPrePhysBones;                          // 0x04B4 (0x0004)
[0x0000000000000000]
int32_t                         bHideSkin;                            // 0x04B8 (0x0004)
[0x0000000000000000]
int32_t                         bForceRawOffset;                          // 0x04BC (0x0004)
[0x0000000000000000]
int32_t                         bIgnoreControllers;                         // 0x04C0 (0x0004)
[0x0000000000000000]
int32_t                         bTransformFromAnimParent;                   // 0x04C4 (0x0004)
[0x0000000000000000]
int32_t                         TickTag;                              // 0x04C8 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                         InitTag;                              // 0x04CC (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                         CachedAtomsTag;                           // 0x04D0 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
int32_t                         bUseSingleBodyPhysics;                      // 0x04D4 (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                         bRequiredBonesUpToDate;                     // 0x04D8 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                          MinDistFactorForKinematicUpdate;             // 0x04DC (0x0004)
[0x0000000000000000]
int32_t                         FramesPhysicsAsleep;                        // 0x04E0 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                         SkipRateForTickAnimNodesAndGetBoneAtoms;      // 0x04E4
(0x0004) [0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                       bSkipTickAnimNodes : 1;                 // 0x04E8 (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
unsigned long                       bSkipGetBoneAtoms : 1;                  // 0x04E8 (0x0004)
[0x0000000000002002] [0x00000002] (CPF_Const | CPF_Transient)
unsigned long                       bInterpolateBoneAtoms : 1;               // 0x04E8 (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                       bHasValidBodies : 1;                    // 0x04E8 (0x0004)
[0x0000000000002002] [0x00000008] (CPF_Const | CPF_Transient)
unsigned long                       bSkipAllUpdateWhenPhysicsAsleep : 1;          // 0x04E8
(0x0004) [0x0000000000000000] [0x00000010]
unsigned long                       bComponentUseFixedSkelBounds : 1;            // 0x04E8
(0x0004) [0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                       bUseBoundsFromParentAnimComponent : 1;        // 0x04E8
(0x0004) [0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                       bConsiderAllBodiesForBounds : 1;             // 0x04E8
(0x0004) [0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                       bUpdateSkelWhenNotRendered : 1;              // 0x04E8
(0x0004) [0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                       bIgnoreControllersWhenNotRendered : 1;        // 0x04E8
```

```
(0x0004) [0x0000000000000000] [0x00000200]
unsigned long                          bTickAnimNodesWhenNotRendered : 1;           // 0x04E8
(0x0004) [0x0000000000000000] [0x00000400]
unsigned long                          bNotUpdatingKinematicDueToDistance : 1;       // 0x04E8
(0x0004) [0x0000000000000002] [0x00000800] (CPF_Const)
unsigned long                          bForceDiscardRootMotion : 1;                 // 0x04E8 (0x0004)
[0x0000000000000001] [0x00001000] (CPF_Edit)
unsigned long                          bNotifyRootMotionProcessed : 1;               // 0x04E8 (0x0004)
[0x0000000000000001] [0x00002000] (CPF_Edit)
unsigned long                          bRootMotionModeChangeNotify : 1;             // 0x04E8
(0x0004) [0x0000000000000000] [0x00004000]
unsigned long                          bRootMotionExtractedNotify : 1;              // 0x04E8 (0x0004)
[0x0000000000000000] [0x00008000]
unsigned long                          bProcessingRootMotion : 1;                   // 0x04E8 (0x0004)
[0x0000000000002000] [0x00010000] (CPF_Transient)
unsigned long                          bDisableFaceFXMaterialInstanceCreation : 1;    // 0x04E8
(0x0004) [0x0000000000000001] [0x00020000] (CPF_Edit)
unsigned long                          bDisableFaceFX : 1;                          // 0x04E8 (0x0004)
[0x0000000000000001] [0x00040000] (CPF_Edit)
unsigned long                          bAnimTreeInitialised : 1;                    // 0x04E8 (0x0004)
[0x0000000000002002] [0x00080000] (CPF_Const | CPF_Transient)
unsigned long                          bForceMeshObjectUpdate : 1;                  // 0x04E8 (0x0004)
[0x0000000000002000] [0x00100000] (CPF_Transient)
unsigned long                          bHasPhysicsAssetInstance : 1;                // 0x04E8 (0x0004)
[0x0000000000000003] [0x00200000] (CPF_Edit | CPF_Const)
unsigned long                          bUpdateKinematicBonesFromAnimation : 1;       // 0x04E8
(0x0004) [0x0000000000000001] [0x00400000] (CPF_Edit)
unsigned long                          bUpdateJointsFromAnimation : 1;              // 0x04E8
(0x0004) [0x0000000000000001] [0x00800000] (CPF_Edit)
unsigned long                          bSkelCompFixed : 1;                          // 0x04E8 (0x0004)
[0x0000000000000002] [0x01000000] (CPF_Const)
unsigned long                          bHasHadPhysicsBlendedIn : 1;                 // 0x04E8 (0x0004)
[0x0000000000000002] [0x02000000] (CPF_Const)
unsigned long                          bForceUpdateAttachmentsInTick : 1;           // 0x04E8
(0x0004) [0x0000000000000001] [0x04000000] (CPF_Edit)
unsigned long                          bEnableFullAnimWeightBodies : 1;             // 0x04E8
(0x0004) [0x0000000000002000] [0x08000000] (CPF_Transient)
unsigned long                          bPerBoneVolumeEffects : 1;                   // 0x04E8 (0x0004)
[0x0000000000000001] [0x10000000] (CPF_Edit)
unsigned long                          bPerBoneMotionBlur : 1;                      // 0x04E8 (0x0004)
[0x0000000000000001] [0x20000000] (CPF_Edit)
unsigned long                          bSyncActorLocationToRootRigidBody : 1;        // 0x04E8
(0x0004) [0x0000000000000001] [0x40000000] (CPF_Edit)
unsigned long                          bUseRawData : 1;                             // 0x04E8 (0x0004)
[0x0000000000000002] [0x80000000] (CPF_Const)
unsigned long                          bDisableWarningWhenAnimNotFound : 1;          // 0x04EC
(0x0004) [0x0000000000000000] [0x00000001]
unsigned long                          bOverrideAttachmentOwnerVisibility : 1;       // 0x04EC
(0x0004) [0x0000000000000000] [0x00000002]
unsigned long                          bNeedsToDeleteHitMask : 1;                   // 0x04EC (0x0004)
[0x0000000000002002] [0x00000004] (CPF_Const | CPF_Transient)
unsigned long                          bPauseAnims : 1;                             // 0x04EC (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                          bChartDistanceFactor : 1;                    // 0x04EC (0x0004)
```

```
[0x0000000000000000] [0x00000010]
unsigned long                    bEnableLineCheckWithBounds : 1;          // 0x04EC
(0x0004) [0x0000000000000000] [0x00000020]
unsigned long                    bCanHighlightSelectedSections : 1;       // 0x04EC
(0x0004) [0x0000000000002000] [0x00000040] (CPF_Transient)
unsigned long                    bUpdateMorphWhenParentAnimComponentExists : 1; //
0x04EC (0x0004) [0x0000000000000001] [0x00000080] (CPF_Edit)
struct FVector                   LineCheckBoundsScale;                // 0x04F0 (0x000C)
[0x0000000000000000]
unsigned long                    bEnableClothSimulation : 1;            // 0x04FC (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                    bDisableClothCollision : 1;           // 0x04FC (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                    bClothFrozen : 1;                  // 0x04FC (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                    bAutoFreezeClothWhenNotRendered : 1;       // 0x04FC
(0x0004) [0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                    bClothAwakeOnStartup : 1;              // 0x04FC (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                    bClothBaseVelClamp : 1;               // 0x04FC (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                    bClothBaseVelInterp : 1;              // 0x04FC (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)
unsigned long                    bAttachClothVertsToBaseBody : 1;          // 0x04FC
(0x0004) [0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                    bIsClothOnStaticObject : 1;            // 0x04FC (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                    bUpdatedFixedClothVerts : 1;            // 0x04FC (0x0004)
[0x0000000000000000] [0x00000200]
unsigned long                    bClothPositionalDampening : 1;           // 0x04FC (0x0004)
[0x0000000000000001] [0x00000400] (CPF_Edit)
unsigned long                    bClothWindRelativeToOwner : 1;           // 0x04FC (0x0004)
[0x0000000000000001] [0x00000800] (CPF_Edit)
unsigned long                    bRecentlyRendered : 1;               // 0x04FC (0x0004)
[0x0000000000002000] [0x00001000] (CPF_Transient)
unsigned long                    bCacheAnimSequenceNodes : 1;            // 0x04FC
(0x0004) [0x0000000000000000] [0x00002000]
unsigned long                    bNeedsInstanceWeightUpdate : 1;          // 0x04FC
(0x0004) [0x0000000000002002] [0x00004000] (CPF_Const | CPF_Transient)
unsigned long                    bAlwaysUseInstanceWeights : 1;          // 0x04FC (0x0004)
[0x0000000000002002] [0x00008000] (CPF_Const | CPF_Transient)
unsigned long                    bUpdateComposeSkeletonPasses : 1;         // 0x04FC
(0x0004) [0x0000000000002002] [0x00010000] (CPF_Const | CPF_Transient)
unsigned long                    bValidTemporarySavedAnimSets : 1;         // 0x04FC
(0x0004) [0x0000000000003002] [0x00020000] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FBonePair>              InstanceVertexWeightBones;          // 0x0500
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FSkelMeshComponentLODInfo>       LODInfo;                       // 0x0510
(0x0010) [0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
struct FVector                   FrozenLocalToWorldPos;               // 0x0520 (0x000C)
[0x0000000000000002] (CPF_Const)
struct FRotator                  FrozenLocalToWorldRot;               // 0x052C (0x000C)
[0x0000000000000002] (CPF_Const)
struct FVector                   ClothExternalForce;               // 0x0538 (0x000C)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FVector                    ClothWind;                        // 0x0544 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                    ClothBaseVelClampRange;                    // 0x0550 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                        ClothBlendWeight;                      // 0x055C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        ClothDynamicBlendWeight;                   // 0x0560 (0x0004)
[0x0000000000000000]
float                        ClothBlendMinDistanceFactor;                // 0x0564 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                        ClothBlendMaxDistanceFactor;                // 0x0568 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                    MinPosDampRange;                        // 0x056C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                    MaxPosDampRange;                        // 0x0578 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                    MinPosDampScale;                        // 0x0584 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                    MaxPosDampScale;                        // 0x0590 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FPointer                   ClothSim;                          // 0x05A0 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                        SceneIndex;                         // 0x05A8 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FVector>                    ClothMeshPosData;                    // 0x05B0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FVector>                    ClothMeshNormalData;                    // 0x05C0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<int32_t>                        ClothMeshIndexData;                  // 0x05D0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
int32_t                        NumClothMeshVerts;                     // 0x05E0 (0x0004)
[0x0000000000000000]
int32_t                        NumClothMeshIndices;                    // 0x05E4 (0x0004)
[0x0000000000000000]
TArray<int32_t>                        ClothMeshParentData;                   // 0x05E8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
int32_t                        NumClothMeshParentIndices;                // 0x05F8 (0x0004)
[0x0000000000000000]
TArray<struct FVector>                    ClothMeshWeldedPosData;                  // 0x0600
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FVector>                    ClothMeshWeldedNormalData;                // 0x0610
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<int32_t>                        ClothMeshWeldedIndexData;               // 0x0620 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                        ClothDirtyBufferFlag;                   // 0x0630 (0x0004)
[0x0000000000000000]
uint8_t                        ClothRBChannel;                       // 0x0634 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FRBCollisionChannelContainer            ClothRBCollideWithChannels;                // 0x0638
(0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
float                        ClothForceScale;                      // 0x063C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                        ClothImpulseScale;                     // 0x0640 (0x0004)
```

```
[0x0000000000000001] (CPF_Edit)
float                          ClothAttachmentTearFactor;                 // 0x0644 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bClothUseCompartment : 1;                  // 0x0648 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
float                          MinDistanceForClothReset;                 // 0x064C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FVector                      LastClothLocation;                        // 0x0650 (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
uint8_t                        ApexClothingRBChannel;                    // 0x065C (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FRBCollisionChannelContainer            ApexClothingRBCollideWithChannels;           //
0x0660 (0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
uint8_t                        ApexClothingCollisionRBChannel;             // 0x0664 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bAutoFreezeApexClothingWhenNotRendered : 1;    // 0x0668
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bLocalSpaceWind : 1;                      // 0x0668 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
struct FVector                      WindVelocity;                             // 0x066C (0x000C)
[0x0000000200000001] (CPF_Edit)
float                          WindVelocityBlendTime;                    // 0x0678 (0x0004)
[0x0000000200000001] (CPF_Edit)
unsigned long                       bSkipInitClothing : 1;                    // 0x067C (0x0004)
[0x0000000000002002] [0x00000001] (CPF_Const | CPF_Transient)
struct FPointer                     SoftBodySim;                              // 0x0680 (0x0008)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
int32_t                        SoftBodySceneIndex;                       // 0x0688 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
unsigned long                       bEnableSoftBodySimulation : 1;              // 0x068C (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
TArray<struct FVector>                   SoftBodyTetraPosData;                 // 0x0690 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<int32_t>                      SoftBodyTetraIndexData;               // 0x06A0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
int32_t                        NumSoftBodyTetraVerts;                    // 0x06B0 (0x0004)
[0x0000000000000000]
int32_t                        NumSoftBodyTetraIndices;                   // 0x06B4 (0x0004)
[0x0000000000000000]
float                          SoftBodyImpulseScale;                     // 0x06B8 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bSoftBodyFrozen : 1;                      // 0x06BC (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                       bAutoFreezeSoftBodyWhenNotRendered : 1;       // 0x06BC
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       bSoftBodyAwakeOnStartup : 1;                // 0x06BC (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                       bSoftBodyUseCompartment : 1;                // 0x06BC
(0x0004) [0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
uint8_t                        SoftBodyRBChannel;                        // 0x06C0 (0x0001)
[0x0000000000000003] (CPF_Edit | CPF_Const)
struct FRBCollisionChannelContainer            SoftBodyRBCollideWithChannels;              //
0x06C4 (0x0004) [0x0000000000000003] (CPF_Edit | CPF_Const)
struct FPointer                     SoftBodyASVPlane;                         // 0x06C8 (0x0008)
```

```
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
class UMaterial*                    LimitMaterial;                    // 0x06D0 (0x0008)
[0x0000000000000000]
uint8_t                        UnknownDataO1[0x8];                    // 0x06D8 (0x0008) MISSED
OFFSET
struct FBoneAtom                        RootMotionDelta;                    // 0x06E0 (0x0020)
[0x0000000000002000] (CPF_Transient)
struct FVector                        RootMotionVelocity;                    // 0x0700 (0x000C)
[0x0000000000002000] (CPF_Transient)
struct FVector                        RootBoneTranslation;                    // 0x070C (0x000C)
[0x0000000000002002] (CPF_Const | CPF_Transient)
struct FVector                        RootMotionAccelScale;                    // 0x0718 (0x000C)
[0x0000000000000000]
uint8_t                        RootMotionMode;                    // 0x0724 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                        PreviousRMM;                    // 0x0725 (0x0001)
[0x0000000000000002] (CPF_Const)
uint8_t                        PendingRMM;                    // 0x0726 (0x0001)
[0x0000000000000000]
uint8_t                        OldPendingRMM;                    // 0x0727 (0x0001)
[0x0000000000000000]
int32_t                        bRMMOneFrameDelay;                    // 0x0728 (0x0004)
[0x0000000000000002] (CPF_Const)
uint8_t                        RootMotionRotationMode;                    // 0x072C (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                        AnimRotationOnly;                    // 0x072D (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                        FaceFXBlendMode;                    // 0x072E (0x0001)
[0x0000000000000001] (CPF_Edit)
struct FPointer                        FaceFXActorInstance;                    // 0x0730 (0x0008)
[0x0000000000003000] (CPF_Native | CPF_Transient)
class UAudioComponent*                        CachedFaceFXAudioComp;                    // 0x0738
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
TArray<uint8_t>                        BoneVisibilityStates;                    // 0x0740 (0x0010)
[0x0000000000402002] (CPF_Const | CPF_Transient | CPF_NeedCtorLink)
class UAkEvent*                        CachedFaceFxAkEvent;                    // 0x0750 (0x0008)
[0x0000000000000000]
uint8_t                        UnknownData02[0x8];                    // 0x0758 (0x0008) MISSED
OFFSET
struct FBoneAtom                        LocalToWorldBoneAtom;                    // 0x0760 (0x0020)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                        ProgressiveDrawingFraction;                    // 0x0780 (0x0004)
[0x0000000000002000] (CPF_Transient)
uint8_t                        CustomSortAlternateIndexMode;                    // 0x0784 (0x0001)
[0x0000000000002000] (CPF_Transient)
TArray<struct FName>                        MorphTargetsQueried;                    // 0x0788 (0x0010)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
unsigned long                        bUseTickOptimization : 1;                    // 0x0798 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
int32_t                        TickCount;                    // 0x079C (0x0004)
[0x0000000000000002] (CPF_Const)
int32_t                        LastDropRate;                    // 0x07A0 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                        LastDropRateChange;                    // 0x07A4 (0x0004)
```

```
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          AccumulatedDroppedDeltaTime;           // 0x07A8 (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)
float                          ComponentDroppedDeltaTime;             // 0x07AC (0x0004)
[0x0000000000002002] (CPF_Const | CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshComponent");
}

return uClassPointer;
};

bool GetSocketOrBoneWorldLocationAndRotation(struct FName SocketOrBoneName, struct
FVector& OutLoc, struct FRotator& OutRot);
void Warmup(float WarmupTime);
void SetMaterial(int32_t ElementIndex, class UMaterialInterface* Material);
struct FRotator GetRotation();
struct FVector GetPosition();
void BreakConstraint(struct FVector Impulse, struct FVector HitLocation, struct FName
InBoneName, unsigned long bVelChange);
void SkelMeshCompOnParticleSystemFinished(class UParticleSystemComponent* PSC);
bool eventPlayParticleEffect(class UAnimNotify_PlayParticleEffect* AnimNotifyData, class
UParticleSystemComponent*& OutParticles);
bool eventCreateForceField(class UAnimNotify_ForceField* AnimNotifyData);
void StopAnim();
void PlayAnim(struct FName AnimName, float Duration, unsigned long bLoop, unsigned long
bRestartIfAlreadyPlaying, float StartTime, unsigned long bPlayBackwards);
void CreateAnimTree();
void ShowMaterialSection(int32_t MaterialID, unsigned long bShow, int32_t LODIndex);
void UpdateMeshForBrokenConstraints();
void UnHideBoneByName(struct FName BoneName);
void HideBoneByName(struct FName BoneName, uint8_t PhysBodyOption);
bool IsBoneHidden(int32_t BoneIndex);
void UnHideBone(int32_t BoneIndex);
void HideBone(int32_t BoneIndex, uint8_t PhysBodyOption);
void SetFaceFXRegisterEx(class FString RegName, uint8_t RegOp, float FirstValue, float
FirstInterpDuration, float NextValue, float NextInterpDuration);
void SetFaceFXRegister(class FString RegName, float RegVal, uint8_t RegOp, float
InterpDuration);
float GetFaceFXRegister(class FString RegName);
void DeclareFaceFXRegister(class FString RegName);
bool IsPlayingFaceFXAnim();
void StopFaceFXAnim();
bool PlayFaceFXAnim(class UFaceFXAnimSet* FaceFXAnimSetRef, class FString AnimName,
class FString GroupName, class USoundCue* SoundCueToPlay, class UAkEvent* AkEventToPlay);
void ToggleInstanceVertexWeights(unsigned long bEnable, int32_t LODIdx);
void UpdateInstanceVertexWeightBones(TArray<struct FBonePair> BonePairs);
```

```
int32_t FindInstanceVertexweightBonePair(struct FBonePair Bones);
void RemoveInstanceVertexWeightBoneParented(struct FName BoneName);
void AddInstanceVertexWeightBoneParented(struct FName BoneName, unsigned long
bPairWithParent);
bool GetBonesWithinRadius(struct FVector Origin, float Radius, int32_t TraceFlags, TArray<struct
FName>& out_Bones);
void UpdateAnimations();
void ForceSkelUpdate();
void UpdateRBBonesFromSpaceBases(unsigned long bMoveUnfixedBodies, unsigned long
bTeleport);
void SetHasPhysicsAssetInstance(unsigned long bHasInstance, unsigned long
bUseCurrentPosition);
class URB_BodyInstance* FindBodyInstanceNamed(struct FName BoneName);
struct FName FindConstraintBoneName(int32_t ConstraintIndex);
int32_t FindConstraintIndex(struct FName ConstraintName);
void InitMorphTargets();
void InitSkelControls();
void UpdateParentBoneMap();
void SetParentAnimComponent(class USkeletalMeshComponent* NewParentAnimComp);
void SetAnimTreeTemplate(class UAnimTree* NewTemplate);
struct FVector GetClosestCollidingBoneLocation(struct FVector TestLocation, unsigned long
bCheckZeroExtent, unsigned long bCheckNonZeroExtent);
struct FName FindClosestBone(struct FVector TestLocation, float IgnoreScale, struct FVector&
BoneLocation);
void TransformFromBoneSpace(struct FName BoneName, struct FVector InPosition, struct
FRotator InRotation, struct FVector& OutPosition, struct FRotator& OutRotation);
void TransformToBoneSpace(struct FName BoneName, struct FVector InPosition, struct
FRotator InRotation, struct FVector& OutPosition, struct FRotator& OutRotation);
struct FVector GetBoneAxis(struct FName BoneName, uint8_t Axis);
struct FVector GetComposedRefPosePosition(struct FName BoneName);
struct FVector GetRefPosePosition(int32_t BoneIndex);
bool BoneIsChildOf(struct FName BoneName, struct FName ParentBoneName);
void GetBoneNames(TArray<struct FName>& BoneNames);
struct FName GetParentBone(struct FName BoneName);
struct FMatrix GetBoneMatrix(int32_t BoneIndex);
struct FName GetBoneName(int32_t BoneIndex);
int32_t MatchRefBone(struct FName BoneName);
struct FVector GetBoneLocation(struct FName BoneName, int32_t Space);
struct FQuat GetBoneQuaternion(struct FName BoneName, int32_t Space);
class UMorphNodeBase* FindMorphNode(struct FName InNodeName);
class USkelControlBase* FindSkelControl(struct FName InControlName);
void AllAnimNodes(class UClass* BaseClass, class UAnimNode*& Node);
class UAnimNode* FindAnimNode(struct FName InNodeName);
class UMorphTarget* FindMorphTarget(struct FName MorphTargetName);
float GetAnimLength(struct FName AnimSeqName);
float GetAnimRateByDuration(struct FName AnimSeqName, float Duration);
void RestoreSavedAnimSets();
void SaveAnimSets();
class UAnimSequence* FindAnimSequence(struct FName AnimSeqName);
void WakeSoftBody();
void SetSoftBodyFrozen(unsigned long bNewFrozen);
void UpdateSoftBodyParams();
void SetClothValidBounds(struct FVector ClothValidBoundsMin, struct FVector
ClothValidBoundsMax);
```

```
void EnableClothValidBounds(unsigned long IfEnableClothValidBounds);
void AttachClothToCollidingShapes(unsigned long AttatchTwoWay, unsigned long
AttachTearable);
void SetClothVelocity(struct FVector VelocityOffSet);
void SetClothPosition(struct FVector ClothOffSet);
void SetClothSleep(unsigned long IfClothSleep);
void SetClothThickness(float ClothThickness);
void SetClothTearFactor(float ClothTearFactor);
void SetClothStretchingStiffness(float ClothStretchingStiffness);
void SetClothSolverIterations(int32_t ClothSolverIterations);
void SetClothSleepLinearVelocity(float ClothSleepLinearVelocity);
void SetClothPressure(float ClothPressure);
void SetClothFriction(float ClothFriction);
void SetClothFlags(int32_t ClothFlags);
void SetClothDampingCoefficient(float ClothDampingCoefficient);
void SetClothCollisionResponseCoefficient(float ClothCollisionResponseCoefficient);
void SetClothBendingStiffness(float ClothBendingStiffness);
void SetClothAttachmentTearFactor(float ClothAttachTearFactor);
void SetClothAttachmentResponseCoefficient(float ClothAttachmentResponseCoefficient);
float GetClothThickness();
float GetClothTearFactor();
float GetClothStretchingStiffness();
int32_t GetClothSolverIterations();
float GetClothSleepLinearVelocity();
float GetClothPressure();
float GetClothFriction();
int32_t GetClothFlags();
float GetClothDampingCoefficient();
float GetClothCollisionResponseCoefficient();
float GetClothBendingStiffness();
float GetClothAttachmentTearFactor();
float GetClothAttachmentResponseCoefficient();
void ForceApexClothingTeleport();
void ForceApexClothingTeleportAndReset();
void ResetClothVertsToRefPose();
void SetAttachClothVertsToBaseBody(unsigned long bAttachVerts);
void SetClothExternalForce(struct FVector InForce);
void UpdateClothParams();
void SetEnableClothingSimulation(unsigned long bInEnable);
void SetClothFrozen(unsigned long bNewFrozen);
void SetEnableClothSimulation(unsigned long bInEnable);
void SetForceRefPose(unsigned long bNewForceRefPose);
void SetPhysicsAsset(class UPhysicsAsset* NewPhysicsAsset, unsigned long bForceReInit);
void SetSkeletalMesh(class USkeletalMesh* NewMesh, unsigned long bKeepSpaceBases);
struct FMatrix GetTransformMatrix();
void AttachedComponents(class UClass* BaseClass, class UActorComponent*&
OutComponent);
bool IsComponentAttached(class UActorComponent* Component, struct FName BoneName);
class UActorComponent* FindComponentAttachedToBone(struct FName InBoneName);
struct FName GetSocketBoneName(struct FName InSocketName);
class USkeletalMeshSocket* GetSocketByName(struct FName InSocketName);
bool GetSocketWorldLocationAndRotation(struct FName InSocketName, int32_t Space, struct
FVector& OutLocation, struct FRotator& OutRotation);
void AttachComponentToSocket(class UActorComponent* Component, struct FName
SocketName);
void DetachComponent(class UActorComponent* Component);
void AttachComponent(class UActorComponent* Component, struct FName BoneName, struct
FVector RelativeLocation, struct FRotator RelativeRotation, struct FVector RelativeScale);
};

// Class Engine.SkeletalMesh
// 0x0494 (0x0060 - 0x04F4)
class USkeletalMesh : public UObject
{
public:
struct FBoxSphereBounds                  Bounds;                         // 0x0060 (0x001C)
[0x0000000000001003] (CPF_Edit | CPF_Const | CPF_Native)
TArray<class UMaterialInterface*>          Materials;                       // 0x0080 (0x0010)
[0x0000000000001003] (CPF_Edit | CPF_Const | CPF_Native)
TArray<class UApexClothingAsset*>          ClothingAssets;                      // 0x0090
(0x0010) [0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
TArray<struct FApexClothingAssetInfo>        ClothingLodMap;                       // 0x00A0
(0x0010) [0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
struct FVector                    Origin;                       // 0x00B0 (0x000C)
[0x0000000000001003] (CPF_Edit | CPF_Const | CPF_Native)
struct FRotator                   RotOrigin;                      // 0x00BC (0x000C)
[0x0000000000001003] (CPF_Edit | CPF_Const | CPF_Native)
TArray<int32_t>                    RefSkeleton;                      // 0x00C8 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
int32_t                        SkeletalDepth;                    // 0x00D8 (0x0004)
[0x0000000000001002] (CPF_Const | CPF_Native)
uint8_t                       UnknownData00[0x50];                 // 0x00E0 (0x0050)
UNKNOWN PROPERTY: MapProperty Engine.SkeletalMesh.NameIndexMap
struct FIndirectArray_Mirror              LODModels;                       // 0x0130 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
struct FPointer                    SourceData;                      // 0x0140 (0x0008)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FBoneAtom>                RefBasesInvMatrix;                   // 0x0148
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)
TArray<struct FBoneMirrorInfo>             SkelMirrorTable;                   // 0x0158 (0x0010)
[0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
uint8_t                        SkelMirrorAxis;                    // 0x0168 (0x0001)
[0x0000000000000001] (CPF_Edit)
uint8_t                        SkelMirrorFlipAxis;                  // 0x0169 (0x0001)
[0x0000000000000001] (CPF_Edit)
TArray<class USkeletalMeshSocket*>          Sockets;                         // 0x0170 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class FString>                  BoneBreakNames;                   // 0x0180 (0x0010)
[0x0000000000021003] (CPF_Edit | CPF_Const | CPF_Native | CPF_EditConst)
TArray<uint8_t>                    BoneBreakOptions;                  // 0x0190 (0x0010)
[0x0000000000001003] (CPF_Edit | CPF_Const | CPF_Native)
TArray<struct FSkeletalMeshLODInfo>         LODInfo;                         // 0x01A0 (0x0010)
[0x0000000000400041] (CPF_Edit | CPF_EditConstArray | CPF_NeedCtorLink)
TArray<struct FSkeletalMeshLODDistanceInfo>     LODDistanceInfo;                      // 0x01B0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FSkeletalMeshOptimizationSettings>  OptimizationSettings;                    //
0x01C0 (0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FName>                   PerPolyCollisionBones;                // 0x01D0 (0x0010)
```

```
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FName>                AddToParentPerPolyCollisionBone;            // 0x01E0
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<int32_t>                     PerPolyBoneKDOPs;                        // 0x01F0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
unsigned long                       bPerPolyUseSoftWeighting : 1;             // 0x0200 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                       bUseSimpleLineCollision : 1;              // 0x0200 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                       bUseSimpleBoxCollision : 1;               // 0x0200 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                       bForceCPUSkinning : 1;                   // 0x0200 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                       bUseFullPrecisionUVs : 1;                // 0x0200 (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
unsigned long                       bHasBeenSimplified : 1;                  // 0x0200 (0x0004)
[0x0000000000000000] [0x00000020]
class UFaceFXAsset*                 FaceFXAsset;                             // 0x0208 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UPhysicsAsset*                BoundsPreviewAsset;                        // 0x0210 (0x0008)
[0x0000000800000001] (CPF_Edit)
TArray<class UMorphTargetSet*>           PreviewMorphSets;                        // 0x0218
(0x0010) [0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                             LODBiasPC;                               // 0x0228 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             LODBiasPS3;                              // 0x022C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             LODBiasXbox360;                            // 0x0230 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             LODBiasPS4;                              // 0x0234 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                             LODBiasNNX;                              // 0x0238 (0x0004)
[0x0000000000000001] (CPF_Edit)
class FString                       SourceFilePath;                           // 0x0240 (0x0010)
[0x0000000800420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)
class FString                       SourceFileTimestamp;                        // 0x0250 (0x0010)
[0x0000000800420003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_NeedCtorLink)
TArray<struct FPointer>             ClothMesh;                               // 0x0260 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<float>                       ClothMeshScale;                          // 0x0270 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<int32_t>                     ClothToGraphicsVertMap;                    // 0x0280 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<float>                       ClothMovementScale;                        // 0x0290 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
uint8_t                             ClothMovementScaleGenMode;                  // 0x02A0 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                               ClothToAnimMeshMaxDist;                    // 0x02A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bLimitClothToAnimMesh : 1;                  // 0x02A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<int32_t>                     ClothWeldingMap;                          // 0x02B0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
int32_t                             ClothWeldingDomain;                        // 0x02C0 (0x0004)
```

```
[0x0000000000000002] (CPF_Const)
TArray<int32_t>                    ClothWeldedIndices;                    // 0x02C8 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
unsigned long                      bForceNoWelding : 1;                   // 0x02D8 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
int32_t                        NumFreeClothVerts;                     // 0x02DC (0x0004)
[0x0000000000000002] (CPF_Const)
TArray<int32_t>                    ClothIndexBuffer;                      // 0x02E0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FName>                   ClothBones;                        // 0x02F0 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
int32_t                        ClothHierarchyLevels;                  // 0x0300 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                      bEnableClothBendConstraints : 1;           // 0x0304 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                      bEnableClothDamping : 1;                 // 0x0304 (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                      bUseClothCOMDamping : 1;                 // 0x0304 (0x0004)
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
float                          ClothStretchStiffness;                 // 0x0308 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothBendStiffness;                    // 0x030C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothDensity;                        // 0x0310 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothThickness;                       // 0x0314 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothDamping;                        // 0x0318 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                        ClothIterations;                      // 0x031C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                        ClothHierarchicalIterations;               // 0x0320 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothFriction;                       // 0x0324 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothRelativeGridSpacing;                // 0x0328 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothPressure;                       // 0x032C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothCollisionResponseCoefficient;          // 0x0330 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothAttachmentResponseCoefficient;          // 0x0334 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothAttachmentTearFactor;                // 0x0338 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          ClothSleepLinearVelocity;                // 0x033C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                          HardStretchLimitFactor;                 // 0x0340 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                      bHardStretchLimit : 1;                  // 0x0344 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                      bEnableClothOrthoBendConstraints : 1;        // 0x0344
(0x0004) [0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                      bEnableClothSelfCollision : 1;             // 0x0344 (0x0004)
```

```
[0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
unsigned long                    bEnableClothPressure : 1;              // 0x0344 (0x0004)
[0x0000000000000003] [0x00000008] (CPF_Edit | CPF_Const)
unsigned long                    bEnableClothTwoWayCollision : 1;            // 0x0344 (0x0004)
[0x0000000000000003] [0x00000010] (CPF_Edit | CPF_Const)
TArray<struct FClothSpecialBoneInfo>          ClothSpecialBones;                   // 0x0348
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
unsigned long                    bEnableClothLineChecks : 1;               // 0x0358 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                    bClothMetal : 1;                 // 0x0358 (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
float                         ClothMetalImpulseThreshold;               // 0x035C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                         ClothMetalPenetrationDepth;               // 0x0360 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                         ClothMetalMaxDeformationDistance;           // 0x0364 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                    bEnableClothTearing : 1;               // 0x0368 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
float                         ClothTearFactor;                  // 0x036C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                        ClothTearReserve;                   // 0x0370 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                    bEnableValidBounds : 1;                // 0x0374 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FVector                   ValidBoundsMin;                      // 0x0378 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                   ValidBoundsMax;                      // 0x0384 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FMap_Mirror                  ClothTornTriMap;                     // 0x0390 (0x0050)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<int32_t>                   SoftBodySurfaceToGraphicsVertMap;           // 0x03E0
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<int32_t>                   SoftBodySurfaceIndices;               // 0x03F0 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FVector>                SoftBodyTetraVertsUnscaled;              // 0x0400
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<int32_t>                   SoftBodyTetraIndices;                 // 0x0410 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FSoftBodyTetraLink>          SoftBodyTetraLinks;                 // 0x0420
(0x0010) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
TArray<struct FPointer>                CachedSoftBodyMeshes;                // 0x0430
(0x0010) [0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<float>                    CachedSoftBodyMeshScales;              // 0x0440 (0x0010)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
TArray<struct FName>                 SoftBodyBones;                    // 0x0450 (0x0010)
[0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
TArray<struct FSoftBodySpecialBoneInfo>        SoftBodySpecialBones;                 // 0x0460
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
float                         SoftBodyVolumeStiffness;                // 0x0470 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                         SoftBodyStretchingStiffness;               // 0x0474 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                         SoftBodyDensity;                   // 0x0478 (0x0004)
```

```
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                               SoftBodyParticleRadius;                    // 0x047C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                               SoftBodyDamping;                           // 0x0480 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                             SoftBodySolverIterations;                  // 0x0484 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                               SoftBodyFriction;                          // 0x0488 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                               SoftBodyRelativeGridSpacing;               // 0x048C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                               SoftBodySleepLinearVelocity;               // 0x0490 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bEnableSoftBodySelfCollision : 1;          // 0x0494 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
float                               SoftBodyAttachmentResponse;                // 0x0498 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                               SoftBodyCollisionResponse;                 // 0x049C (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
float                               SoftBodyDetailLevel;                       // 0x04A0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                             SoftBodySubdivisionLevel;                  // 0x04A4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bSoftBodyIsoSurface : 1;                   // 0x04A8 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                       bEnableSoftBodyDamping : 1;                // 0x04A8 (0x0004)
[0x0000000000000003] [0x00000002] (CPF_Edit | CPF_Const)
unsigned long                       bUseSoftBodyCOMDamping : 1;                // 0x04A8
(0x0004) [0x0000000000000003] [0x00000004] (CPF_Edit | CPF_Const)
float                               SoftBodyAttachmentThreshold;               // 0x04AC (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bEnableSoftBodyTwoWayCollision : 1;        // 0x04B0
(0x0004) [0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
float                               SoftBodyAttachmentTearFactor;              // 0x04B4 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
unsigned long                       bEnableSoftBodyLineChecks : 1;             // 0x04B8 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)
unsigned long                       bHasVertexColors : 1;                      // 0x04B8 (0x0004)
[0x0000000000000000] [0x00000002]
TArray<unsigned long>               GraphicsIndexIsCloth;                      // 0x04C0 (0x0010)
[0x0000000000001002] (CPF_Const | CPF_Native)
TArray<float>                       CachedStreamingTextureFactors;             // 0x04D0
(0x0010) [0x0000000000001002] (CPF_Const | CPF_Native)
float                               StreamingDistanceMultiplier;               // 0x04E0 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)
int32_t                             ReleaseResourcesFence;                     // 0x04E4 (0x0004)
[0x0000000000003002] (CPF_Const | CPF_Native | CPF_Transient)
uint64_t                            SkelMeshRUID;                              // 0x04E8 (0x0008)
[0x0000000000002002] (CPF_Const | CPF_Transient)
unsigned long                       bUseClothingAssetMaterial : 1;             // 0x04F0 (0x0004)
[0x0000000000000003] [0x00000001] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMesh");
}

return uClassPointer;
};

struct FVector GetSocketRelativeLocation(struct FName InSocketName);
bool HasSocket(struct FName InSocketName);
};

// Class Engine.SkeletalMeshSocket
// 0x0058 (0x0060 - 0x00B8)
class USkeletalMeshSocket : public UObject
{
public:
struct FName                             SocketName;                              // 0x0060 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FName                             BoneName;                                // 0x0068 (0x0008)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FVector                           RelativeLocation;                        // 0x0070 (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FRotator                          RelativeRotation;                        // 0x007C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector                           RelativeScale;                           // 0x0088 (0x000C)
[0x0000000000000001] (CPF_Edit)
class USkeletalMesh*                     PreviewSkelMesh;                         // 0x0098 (0x0008)
[0x0000000800000001] (CPF_Edit)
class USkeletalMeshComponent*            PreviewSkelComp;                         // 0x00A0
(0x0008) [0x00000000040A200B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_Transient |
CPF_EditConst | CPF_Component | CPF_EditInline)
class UStaticMesh*                       PreviewStaticMesh;                       // 0x00A8 (0x0008)
[0x0000000800000001] (CPF_Edit)
class UParticleSystem*                   PreviewParticleSystem;                   // 0x00B0 (0x0008)
[0x0000000800000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshSocket");
}

return uClassPointer;
};

};
```

```
// Class Engine.SplineActor
// 0x0070 (0x0268 - 0x02D8)
class ASplineActor : public AActor
{
public:
TArray<struct FSplineConnection>                Connections;                              // 0x0268 (0x0010)
[0x0000000000480000] (CPF_Component | CPF_NeedCtorLink)
struct FVector                                  SplineActorTangent;                       // 0x0278 (0x000C)
[0x0000000200000001] (CPF_Edit)
struct FColor                                   SplineColor;                              // 0x0284 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                                   bDisableDestination : 1;                  // 0x0288 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                   bAlreadyVisited : 1;                      // 0x0288 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
TArray<class ASplineActor*>                     LinksFrom;                                // 0x0290 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class ASplineActor*                             nextOrdered;                              // 0x02A0 (0x0008)
[0x0000000000002000] (CPF_Transient)
class ASplineActor*                             prevOrdered;                              // 0x02A8 (0x0008)
[0x0000000000002000] (CPF_Transient)
class ASplineActor*                             previousPath;                             // 0x02B0 (0x0008)
[0x0000000000002000] (CPF_Transient)
int32_t                                         bestPathWeight;                           // 0x02B8 (0x0004)
[0x0000000000002000] (CPF_Transient)
int32_t                                         visitedWeight;                            // 0x02BC (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FInterpCurveFloat                        SplineVelocityOverTime;                   // 0x02C0 (0x0018)
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SplineActor");
}

return uClassPointer;
};

void OnToggleHidden(class USeqAct_ToggleHidden* Action);
void OnToggle(class USeqAct_Toggle* inAction);
void GetAllConnectedSplineActors(TArray<class ASplineActor*>& OutSet);
bool FindSplinePathTo(class ASplineActor* Goal, TArray<class ASplineActor*>& OutRoute);
class ASplineActor* GetBestConnectionInDirection(struct FVector DesiredDir, unsigned long
bUseLinksFrom);
class ASplineActor* GetRandomConnection(unsigned long bUseLinksFrom);
void BreakAllConnectionsFrom();
void BreakAllConnections();
void BreakConnectionTo(class ASplineActor* NextActor);
```

```
class ASplineActor* FindTargetForComponent(class USplineComponent* SplineComp);
class USplineComponent* FindSplineComponentTo(class ASplineActor* NextActor);
bool IsConnectedTo(class ASplineActor* NextActor, unsigned long
bCheckForDisableDestination);
void AddConnectionTo(class ASplineActor* NextActor);
void UpdateConnectedSplineComponents(unsigned long bFinish);
struct FVector GetWorldSpaceTangent();
};

// Class Engine.SplineLoftActor
// 0x005C (0x02D8 - 0x0334)
class ASplineLoftActor : public ASplineActor
{
public:
float                                   ScaleX;                                 // 0x02D8 (0x0004)
[0x0000000200000001] (CPF_Edit)
float                                   ScaleY;                                 // 0x02DC (0x0004)
[0x0000000200000001] (CPF_Edit)
TArray<class USplineMeshComponent*>             SplineMeshComps;                        // 0x02E0
(0x0010) [0x0000000004480008] (CPF_ExportObject | CPF_Component | CPF_NeedCtorLink |
CPF_EditInline)
class UStaticMesh*                              DeformMesh;                             // 0x02F0 (0x0008)
[0x0000000000000003] (CPF_Edit | CPF_Const)
TArray<class UMaterialInterface*>               DeformMeshMaterials;                    // 0x02F8
(0x0010) [0x0000000000400003] (CPF_Edit | CPF_Const | CPF_NeedCtorLink)
float                                   Roll;                                   // 0x0308 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FVector                                  WorldXDir;                              // 0x030C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FVector2D                                Offset;                                 // 0x0318 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                                   bSmoothInterpRollAndScale : 1;          // 0x0320 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                   bAcceptsLights : 1;                     // 0x0320 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
class UDynamicLightEnvironmentComponent*        MeshLightEnvironment;                   //
0x0328 (0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject |
CPF_EditConst | CPF_Component | CPF_EditInline)
float                                   MeshMaxDrawDistance;                    // 0x0330 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SplineLoftActor");
}

return uClassPointer;
};
```

```
void UpdateSplineParams();
void ClearLoftMesh();
};

// Class Engine.SplineLoftActorMovable
// 0x0004 (0x0334 - 0x0338)
class ASplineLoftActorMovable : public ASplineLoftActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SplineLoftActorMovable");
}

return uClassPointer;
};

};

// Class Engine.SplineComponent
// 0x0048 (0x0258 - 0x02A0)
class USplineComponent : public UPrimitiveComponent
{
public:
struct FInterpCurveVector                    SplineInfo;                              // 0x0258 (0x0018)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                        SplineCurviness;                         // 0x0270 (0x0004)
[0x0000000000020001] (CPF_Edit | CPF_EditConst)
struct FColor                                SplineColor;                             // 0x0274 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                        SplineDrawRes;                           // 0x0278 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                        SplineArrowSize;                         // 0x027C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                                bSplineDisabled : 1;                     // 0x0280 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FInterpCurveFloat                     SplineReparamTable;                      // 0x0288 (0x0018)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SplineComponent");
}
return uClassPointer;
};

float GetDistanceAlongSpline(struct FVector Location, unsigned long bClamp);
struct FVector GetTangentAtDistanceAlongSpline(float Distance);
struct FVector GetLocationAtDistanceAlongSpline(float Distance);
float GetSplineLength();
void UpdateSplineReparamTable();
void UpdateSplineCurviness();
};

// Class Engine.ProcBuilding
// 0x0128 (0x02A4 - 0x03CC)
class AProcBuilding : public AVolume
{
public:
class UProcBuildingRuleset*                    Ruleset;                              // 0x02A8 (0x0008)
[0x0000000800000001] (CPF_Edit)
TArray<struct FPBMeshCompInfo>                 BuildingMeshCompInfos;                 // 0x02B0
(0x0010) [0x00000000004A0003] (CPF_Edit | CPF_Const | CPF_EditConst | CPF_Component |
CPF_NeedCtorLink)
TArray<struct FPBFracMeshCompInfo>             BuildingFracMeshCompInfos;              //
0x02C0 (0x0010) [0x00000000004A0003] (CPF_Edit | CPF_Const | CPF_EditConst |
CPF_Component | CPF_NeedCtorLink)
class UStaticMeshComponent*                    SimpleMeshComp;                        // 0x02D0
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                                  bGenerateRoofMesh : 1;                 // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                  bGenerateFloorMesh : 1;                // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                                  bApplyRulesToRoof : 1;                 // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                                  bApplyRulesToFloor : 1;                // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                                  bSplitWallsAtRoofLevels : 1;           // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                                  bSplitWallsAtWallEdges : 1;            // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                                  bQuickEdited : 1;                      // 0x02D8 (0x0004)
[0x0000000000002000] [0x00000040] (CPF_Transient)
unsigned long                                  bBuildingBrushCollision : 1;           // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000080] (CPF_Edit)
unsigned long                                  bDebugDrawEdgeInfo : 1;                // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000100] (CPF_Edit)
unsigned long                                  bDebugDrawScopes : 1;                  // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000200] (CPF_Edit)
TArray<class UStaticMeshComponent*>            LODMeshComps;                          // 0x02E0
(0x0010) [0x000000000448000A] (CPF_Const | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
TArray<struct FPBFaceUVInfo>                   LODMeshUVInfos;                        // 0x02F0
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<struct FPBScope2D>                      TopLevelScopes;                        // 0x0300
```

```
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
int32_t                          NumMeshedTopLevelScopes;                // 0x0310 (0x0004)
[0x0000000000000000]
TArray<struct FPBFaceUVInfo>                 TopLevelScopeUVInfos;                   // 0x0318
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<struct FPBScopeProcessInfo>               TopLevelScopeInfos;                    // 0x0328
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
TArray<struct FPBEdgeInfo>                 EdgeInfos;                             // 0x0338 (0x0010)
[0x0000000800400000] (CPF_NeedCtorLink)
float                            MaxFacadeZ;                            // 0x0348 (0x0004)
[0x0000000000000000]
float                            MinFacadeZ;                            // 0x034C (0x0004)
[0x0000000000000000]
TArray<class AProcBuilding*>                 OverlappingBuildings;                  // 0x0350
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                            SimpleMeshMassiveLODDistance;              // 0x0360 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RenderToTexturePullBackAmount;             // 0x0364 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          RoofLightmapRes;                       // 0x0368 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          NonRectWallLightmapRes;                 // 0x036C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            LODRenderToTextureScale;                // 0x0370 (0x0004)
[0x0000000800000001] (CPF_Edit)
struct FName                        ParamSwatchName;                       // 0x0374 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<struct FPBMaterialParam>                BuildingMaterialParams;                 // 0x0380
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UMaterialInstanceConstant*>        BuildingMatParamMICs;                   //
0x0390 (0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
class AStaticMeshActor*                  LowLODPersistentActor;                 // 0x03A0
(0x0008) [0x0000100000220003] (CPF_Edit | CPF_Const | CPF_EditConst)
class UStaticMeshComponent*                  CurrentSimpleMeshComp;                  // 0x03A8
(0x0008) [0x0000000004082008] (CPF_ExportObject | CPF_Transient | CPF_Component |
CPF_EditInline)
class AActor*                        CurrentSimpleMeshActor;                 // 0x03B0 (0x0008)
[0x0000000000002000] (CPF_Transient)
TArray<class AProcBuilding*>                 AttachedBuildings;                     // 0x03B8
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)
int32_t                          BuildingInstanceVersion;                // 0x03C8 (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ProcBuilding");
}

return uClassPointer;
};

int32_t FindEdgeForTopLevelScope(int32_t TopLevelScopeIndex, uint8_t Edge);
void BreakFractureComponent(class UFracturedStaticMeshComponent* Comp, struct FVector
BoxMin, struct FVector BoxMax);
void GetAllGroupedProcBuildings(TArray<class AProcBuilding*>& OutSet);
class AProcBuilding* GetBaseMostBuilding();
TArray<class UStaticMeshComponent*> FindComponentsForTopLevelScope(int32_t
TopLevelScopeIndex);
void ClearBuildingMeshes();
};

// Class Engine.ProcBuilding_SimpleLODActor
// 0x0000 (0x0288 - 0x0288)
class AProcBuilding_SimpleLODActor : public AStaticMeshActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ProcBuilding_SimpleLODActor");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeBase
// 0x0034 (0x0060 - 0x0094)
class UPBRuleNodeBase : public UObject
{
public:
TArray<struct FPBRuleLink>                    NextRules;                              // 0x0060 (0x0010)
[0x0000000000400040] (CPF_EditConstArray | CPF_NeedCtorLink)
class FString                                 Comment;                                // 0x0070 (0x0010)
[0x0000000800400001] (CPF_Edit | CPF_NeedCtorLink)
int32_t                                       RulePosX;                               // 0x0080 (0x0004)
[0x0000000800000000]
int32_t                                       RulePosY;                               // 0x0084 (0x0004)
[0x0000000800000000]
int32_t                                       InDrawY;                                // 0x0088 (0x0004)
[0x0000000800000000]
int32_t                                       DrawWidth;                              // 0x008C (0x0004)
[0x0000000800000000]
int32_t                                       DrawHeight;                             // 0x0090 (0x0004)
[0x0000000800000000]

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeBase");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeAlternate
// 0x0014 (0x0094 - 0x00A8)
class UPBRuleNodeAlternate : public UPBRuleNodeBase
{
public:
uint8_t                          RepeatAxis;                          // 0x0098 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                            ASize;                               // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            BMaxSize;                            // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                    bInvertPatternOrder : 1;             // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bEqualSizeAB : 1;                    // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeAlternate");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeComment
// 0x001C (0x0094 - 0x00B0)
class UPBRuleNodeComment : public UPBRuleNodeBase
{
public:
int32_t                          SizeX;                               // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                          SizeY;                               // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
```

```
int32_t                                  BorderWidth;                              // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FColor                            BorderColor;                              // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bFilled : 1;                              // 0x00A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
struct FColor                            FillColor;                                // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeComment");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeCorner
// 0x0034 (0x0094 - 0x00C8)
class UPBRuleNodeCorner : public UPBRuleNodeBase
{
public:
float                                    CornerSize;                               // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<struct FRBCornerAngleInfo>                  Angles;                                   // 0x00A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
float                                    FlatThreshold;                            // 0x00B0 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bNoMeshForConcaveCorners : 1;                  // 0x00B4
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                            bUseAdjacentRulesetForRightGap : 1;            // 0x00B4
(0x0004) [0x0000000000000001] [0x00000002] (CPF_Edit)
uint8_t                                  CornerType;                               // 0x00B8 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                    CornerShapeOffset;                        // 0x00BC (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                  RoundTesselation;                         // 0x00C0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                    RoundCurvature;                           // 0x00C4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeCorner");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeCycle
// 0x0014 (0x0094 - 0x00A8)
class UPBRuleNodeCycle : public UPBRuleNodeBase
{
public:
uint8_t                                   RepeatAxis;                              // 0x0098 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                    RepeatSize;                              // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                   CycleSize;                               // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                                   bFixRepeatSize : 1;                      // 0x00A4 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeCycle");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeEdgeAngle
// 0x001C (0x0094 - 0x00B0)
class UPBRuleNodeEdgeAngle : public UPBRuleNodeBase
{
public:
uint8_t                                   Edge;                                    // 0x0098 (0x0001)
[0x0000000000000001] (CPF_Edit)
TArray<struct FRBEdgeAngleInfo>                 Angles;                                  // 0x00A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeEdgeAngle");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeEdgeMesh
// 0x000C (0x0094 - 0x00A0)
class UPBRuleNodeEdgeMesh : public UPBRuleNodeBase
{
public:
float                                   FlatThreshold;                            // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   MainXPullIn;                              // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeEdgeMesh");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeExtractTopBottom
// 0x0014 (0x0094 - 0x00A8)
class UPBRuleNodeExtractTopBottom : public UPBRuleNodeBase
{
public:
float                                   ExtractTopZ;                              // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   ExtractNotTopZ;                           // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   ExtractBottomZ;                           // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                   ExtractNotBottomZ;                        // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeExtractTopBottom");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeLODQuad
// 0x0008 (0x0094 - 0x009C)
class UPBRuleNodeLODQuad : public UPBRuleNodeBase
{
public:
float                                   MassiveLODDistanceScale;                    // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeLODQuad");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeMesh
// 0x0068 (0x0094 - 0x00FC)
class UPBRuleNodeMesh : public UPBRuleNodeBase
{
public:
TArray<struct FBuildingMeshInfo>                BuildingMeshes;                             // 0x0098
(0x0010) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
struct FBuildingMeshInfo                        PartialOccludedBuildingMesh;                // 0x00A8
(0x0050) [0x0000000000480001] (CPF_Edit | CPF_Component | CPF_NeedCtorLink)
unsigned long                                   bDoOcclusionTest : 1;                       // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                   bBlockAll : 1;                              // 0x00F8 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeMesh");
```

```
}

return uClassPointer;
};

int32_t PickRandomBuildingMesh();
};

// Class Engine.PBRuleNodeOcclusion
// 0x0004 (0x0094 - 0x0098)
class UPBRuleNodeOcclusion : public UPBRuleNodeBase
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeOcclusion");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeQuad
// 0x0020 (0x0094 - 0x00B4)
class UPBRuleNodeQuad : public UPBRuleNodeBase
{
public:
class UMaterialInterface*                Material;                        // 0x0098 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            RepeatMaxSizeX;                   // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RepeatMaxSizeZ;                   // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                           QuadLightmapRes;                  // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            YOffset;                       // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                       bDisableMaterialRepeat : 1;          // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeQuad");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeRandom
// 0x0010 (0x0094 - 0x00A4)
class UPBRuleNodeRandom : public UPBRuleNodeBase
{
public:
int32_t                        NumOutputs;                   // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                        MinNumExecuted;                 // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                        MaxNumExecuted;                 // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeRandom");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeRepeat
// 0x000C (0x0094 - 0x00A0)
class UPBRuleNodeRepeat : public UPBRuleNodeBase
{
public:
uint8_t                        RepeatAxis;                   // 0x0098 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                         RepeatMaxSize;                 // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeRepeat");
}
```

```
return uClassPointer;
};

};

// Class Engine.PBRuleNodeSize
// 0x0010 (0x0094 - 0x00A4)
class UPBRuleNodeSize : public UPBRuleNodeBase
{
public:
uint8_t                              SizeAxis;                          // 0x0098 (0x0001)
[0x0000000000000001] (CPF_Edit)
float                                DecisionSize;                      // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                        bUseTopLevelScopeSize : 1;              // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeSize");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeSplit
// 0x001C (0x0094 - 0x00B0)
class UPBRuleNodeSplit : public UPBRuleNodeBase
{
public:
uint8_t                              SplitAxis;                         // 0x0098 (0x0001)
[0x0000000000000001] (CPF_Edit)
TArray<struct FRBSplitInfo>                SplitSetup;                        // 0x00A0 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeSplit");
}

return uClassPointer;
```

```
};

};

// Class Engine.PBRuleNodeSubRuleset
// 0x000C (0x0094 - 0x00A0)
class UPBRuleNodeSubRuleset : public UPBRuleNodeBase
{
public:
class UProcBuildingRuleset*                SubRuleset;                          // 0x0098 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeSubRuleset");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeTransform
// 0x001C (0x0094 - 0x00B0)
class UPBRuleNodeTransform : public UPBRuleNodeBase
{
public:
class UDistributionVector*                 Translation;                         // 0x0098 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)
class UDistributionVector*                 Rotation;                            // 0x00A0 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)
class UDistributionVector*                 Scale;                               // 0x00A8 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeTransform");
}

return uClassPointer;
};

};
```

```
// Class Engine.PBRuleNodeVariation
// 0x0008 (0x0094 - 0x009C)
class UPBRuleNodeVariation : public UPBRuleNodeBase
{
public:
unsigned long                          bVariationOfScopeOnLeft : 1;                // 0x0098 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeVariation");
}

return uClassPointer;
};

};

// Class Engine.PBRuleNodeWindowWall
// 0x002C (0x0094 - 0x00C0)
class UPBRuleNodeWindowWall : public UPBRuleNodeBase
{
public:
float                                  CellMaxSizeX;                          // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  CellMaxSizeZ;                          // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  WindowSizeX;                           // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  WindowSizeZ;                           // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  WindowPosX;                            // 0x00A8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                                  WindowPosZ;                            // 0x00AC (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                          bScaleWindowWithCell : 1;              // 0x00B0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                  YOffset;                         // 0x00B4 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*                     Material;                         // 0x00B8 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.PBRuleNodeWindowWall");
}

return uClassPointer;
};

};

// Class Engine.ProcBuildingRuleset
// 0x0088 (0x0060 - 0x00E8)
class UProcBuildingRuleset : public UObject
{
public:
class UPBRuleNodeBase*                  RootRule;                          // 0x0060 (0x0008)
[0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
unsigned long                     bBeingEdited : 1;                  // 0x0068 (0x0004)
[0x0000000800002000] [0x00000001] (CPF_Transient)
unsigned long                     bEnableInteriorTexture : 1;              // 0x0068 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                     bLODOnlyRoof : 1;                  // 0x0068 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                     bPickRandomSwatch : 1;                // 0x0068 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
class UMaterialInterface*               DefaultRoofMaterial;                  // 0x0070 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*               DefaultFloorMaterial;                 // 0x0078 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UMaterialInterface*               DefaultNonRectWallMaterial;               // 0x0080
(0x0008) [0x0000000000000001] (CPF_Edit)
float                          RoofZOffset;                     // 0x0088 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          NotRoofZOffset;                    // 0x008C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FloorZOffset;                     // 0x0090 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          NotFloorZOffset;                    // 0x0094 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          RoofPolyInset;                     // 0x0098 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          FloorPolyInset;                    // 0x009C (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          BuildingLODSpecular;                  // 0x00A0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                          RoofEdgeScopeRaise;                  // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)
class UTexture*                    LODCubemap;                       // 0x00A8 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture*                    InteriorTexture;                    // 0x00B0 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<struct FPBVariationInfo>              Variations;                       // 0x00B8 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<struct FPBParamSwatch>               ParamSwatches;                      // 0x00C8
(0x0010) [0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
TArray<class UPBRuleNodeComment*>              Comments;                          // 0x00D8
```

```
(0x0010) [0x0000000800400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ProcBuildingRuleset");
}

return uClassPointer;
};

};

// Class Engine.ReplicationInfo
// 0x0000 (0x0268 - 0x0268)
class AReplicationInfo : public AInfo
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ReplicationInfo");
}

return uClassPointer;
};

};

// Class Engine.GameReplicationInfo
// 0x0070 (0x0268 - 0x02D8)
class AGameReplicationInfo : public AReplicationInfo
{
public:
class UGroupComponent_ORS*                    RegistryGroup;                              // 0x0268
(0x0008) [0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_EditInline)
class UClass*                                 GameClass;                                  // 0x0270 (0x0008)
[0x0000000100000020] (CPF_Net)
unsigned long                                 bStopCountDown : 1;                         // 0x0278 (0x0004)
[0x0000000000000020] [0x00000001] (CPF_Net)
unsigned long                                 bMatchHasBegun : 1;                         // 0x0278 (0x0004)
[0x0000000100000020] [0x00000002] (CPF_Net)
unsigned long                                 bMatchIsOver : 1;                           // 0x0278 (0x0004)
[0x0000000100000020] [0x00000004] (CPF_Net)
```

```
int32_t                                  RemainingTime;                           // 0x027C (0x0004)
[0x0000000000000020] (CPF_Net)
int32_t                                  ElapsedTime;                            // 0x0280 (0x0004)
[0x0000000000000020] (CPF_Net)
int32_t                                  RemainingMinute;                         // 0x0284 (0x0004)
[0x0000000000000020] (CPF_Net)
int32_t                                  GoalScore;                              // 0x0288 (0x0004)
[0x0000000000000020] (CPF_Net)
int32_t                                  TimeLimit;                              // 0x028C (0x0004)
[0x0000000000000020] (CPF_Net)
TArray<class ATeamInfo*>                      Teams;                                   // 0x0290 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                                ServerName;                             // 0x02A0 (0x0010)
[0x0000000100444021] (CPF_Edit | CPF_Net | CPF_Config | CPF_GlobalConfig |
CPF_NeedCtorLink)
class AActor*                                 Winner;                                 // 0x02B0 (0x0008)
[0x0000000000000020] (CPF_Net)
TArray<class APlayerReplicationInfo*>             PRIArray;                               // 0x02B8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class APlayerReplicationInfo*>             InactivePRIArray;                          // 0x02C8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.GameReplicationInfo");
}

return uClassPointer;
};

bool eventShouldShowGore();
bool IsCoopMultiplayerGame();
bool IsMultiplayerGame();
void EndGame();
void StartMatch();
void SetTeam(int32_t Index, class ATeamInfo* TI);
void RemovePRI(class APlayerReplicationInfo* PRI);
void AddPRI(class APlayerReplicationInfo* PRI);
bool OnSameTeam(class AActor* A, class AActor* B);
void eventTimer();
void Reset();
void ReceivedGameClass();
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.PlayerReplicationInfo
// 0x01A8 (0x0268 - 0x0410)
class APlayerReplicationInfo : public AReplicationInfo
```

```
{
public:
class UObjectProvider*                    ObjectProvider;                         // 0x0268 (0x0008)
[0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UGroupComponent_ORS*                RegistryGroup;                          // 0x0270
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
int32_t                                   Score;                                  // 0x0278 (0x0004)
[0x0000000100000020] (CPF_Net)
int32_t                                   Deaths;                                 // 0x027C (0x0004)
[0x0000000000000020] (CPF_Net)
uint8_t                                   Ping;                                   // 0x0280 (0x0001)
[0x0000000000000020] (CPF_Net)
uint8_t                                   TTSSpeaker;                             // 0x0281 (0x0001)
[0x0000000000002000] (CPF_Transient)
int32_t                                   NumLives;                               // 0x0284 (0x0004)
[0x0000000000000000]
class FString                             PlayerName;                             // 0x0288 (0x0010)
[0x0000000100400020] (CPF_Net | CPF_NeedCtorLink)
class FString                             OldName;                                // 0x0298 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                   PlayerID;                               // 0x02A8 (0x0004)
[0x0000000000000020] (CPF_Net)
class ATeamInfo*                          Team;                                   // 0x02B0 (0x0008)
[0x0000000104000020] (CPF_Net | CPF_EditInline)
unsigned long                             bAdmin : 1;                             // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000001] (CPF_Net)
unsigned long                             bIsSpectator : 1;                       // 0x02B8 (0x0004)
[0x0000000100000020] [0x00000002] (CPF_Net)
unsigned long                             bOnlySpectator : 1;                     // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000004] (CPF_Net)
unsigned long                             bWaitingPlayer : 1;                     // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000008] (CPF_Net)
unsigned long                             bReadyToPlay : 1;                       // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000010] (CPF_Net)
unsigned long                             bOutOfLives : 1;                        // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000020] (CPF_Net)
unsigned long                             bBot : 1;                               // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000040] (CPF_Net)
unsigned long                             bIsInactive : 1;                        // 0x02B8 (0x0004)
[0x0000000100000020] [0x00000080] (CPF_Net)
unsigned long                             bFromPreviousLevel : 1;                 // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000100] (CPF_Net)
unsigned long                             bTimedOut : 1;                          // 0x02B8 (0x0004)
[0x0000000000000020] [0x00000200] (CPF_Net)
unsigned long                             bUnregistered : 1;                      // 0x02B8 (0x0004)
[0x0000000000002000] [0x00000400] (CPF_Transient)
int32_t                                   StartTime;                              // 0x02BC (0x0004)
[0x0000000000000000]
class FString                             StringSpectating;                       // 0x02C0 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                             StringUnknown;                          // 0x02D0 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
int32_t                                   Kills;                                  // 0x02E0 (0x0004)
[0x0000000000000000]
```

```
float                              ExactPing;                         // 0x02E4 (0x0004)
[0x0000000000000000]
class FString                      SavedNetworkAddress;                 // 0x02E8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FUniqueNetId                UniqueId;                          // 0x02F8 (0x0048)
[0x0000000100400020] (CPF_Net | CPF_NeedCtorLink)
struct FName                       SessionName;                       // 0x0340 (0x0008)
[0x0000000000000002] (CPF_Const)
struct FAutomatedTestingDatum              AutomatedTestingData;                 // 0x0348
(0x0008) [0x0000000000000000]
class UTexture2D*                  Avatar;                            // 0x0350 (0x0008)
[0x0000000000002000] (CPF_Transient)
struct FNetPacketStats             PacketStats;                       // 0x0358 (0x001C)
[0x0000000000000000]
struct FNetPacketStats             PrevPacketStats;                     // 0x0374 (0x001C)
[0x0000000000000000]
struct FClientConnectionStats              NetStats;                          // 0x0390 (0x0058)
[0x0000000000000000]
struct FPRIRemoteUserData                  RemoteUserData;                       // 0x03E8
(0x0010) [0x0000000100400020] (CPF_Net | CPF_NeedCtorLink)
struct FScriptDelegate             __EventTeamChanged__Delegate;              // 0x03F8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlayerReplicationInfo");
}

return uClassPointer;
};

void UnregisterPlayerFromSession();
void RegisterPlayerWithSession();
bool IsInvalidName();
uint8_t GetTeamNum();
void SetUniqueId(struct FUniqueNetId PlayerUniqueId);
void SeamlessTravelTo(class APlayerReplicationInfo* NewPRI);
void IncrementDeaths(int32_t Amt);
void CopyProperties(class APlayerReplicationInfo* PRI);
void OverrideWith(class APlayerReplicationInfo* PRI);
class APlayerReplicationInfo* Duplicate();
void SetWaitingPlayer(unsigned long B);
void eventSetPlayerName(class FString S);
void DisplayDebug(class AHUD* HUD, float& YL, float& YPos);
class FString GetHumanReadableName();
void Reset();
void eventDestroyed();
void Unregister();
void UpdatePing(float TimeStamp);
void eventReplicatedEvent(struct FName VarName);
void RemoteUserDataReplicated();
void SetPlayerTeam(class ATeamInfo* NewTeam);
void ClientInitialize(class AController* C);
void eventUpdateRemoteUserData(struct FPRIRemoteUserData Data);
void ServerUpdateRemoteUserData(struct FPRIRemoteUserData Data);
void eventClientFillRemoteUserData();
void eventPostBeginPlay();
void UpdateRegistryGroupParent();
void UpdateObjectProviderParent();
void eventOnOwnerChanged();
void eventConstruct();
void OnTeamChanged();
void EventTeamChanged(class APlayerReplicationInfo* PRI);
};

// Class Engine.TeamInfo
// 0x0028 (0x0268 - 0x0290)
class ATeamInfo : public AReplicationInfo
{
public:
class FString                         TeamName;                          // 0x0268 (0x0010)
[0x0000000000408022] (CPF_Const | CPF_Net | CPF_Localized | CPF_NeedCtorLink)
int32_t                             Size;                              // 0x0278 (0x0004)
[0x0000000000000000]
int32_t                             Score;                             // 0x027C (0x0004)
[0x0000000100000020] (CPF_Net)
int32_t                             TeamIndex;                         // 0x0280 (0x0004)
[0x0000000100000020] (CPF_Net)
struct FColor                         TeamColor;                         // 0x0284 (0x0004)
[0x0000000000000000]
class UGroupComponent_ORS*                 RegistryGroup;                       // 0x0288
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TeamInfo");
}

return uClassPointer;
};

uint8_t GetTeamNum();
class FString GetHumanReadableName();
void RemoveFromTeam(class AController* Other);
bool AddToTeam(class AController* Other);
void eventDestroyed();
void eventReplicatedEvent(struct FName VarName);
};
```

```
// Class Engine.Camera
// 0x0350 (0x0268 - 0x05B8)
class ACamera : public AActor
{
public:
class APlayerController*                    PCOwner;                              // 0x0268 (0x0008)
[0x0000000000000000]
struct FName                         CameraStyle;                          // 0x0270 (0x0008)
[0x0000000000000000]
float                                DefaultFOV;                           // 0x0278 (0x0004)
[0x0000000000000000]
unsigned long                        bLockedFOV : 1;                       // 0x027C (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                        bConstrainAspectRatio : 1;            // 0x027C (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                        bEnableFading : 1;                    // 0x027C (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                        bFadeAudio : 1;                       // 0x027C (0x0004)
[0x0000000000000000] [0x00000008]
unsigned long                        bForceDisableTemporalAA : 1;          // 0x027C (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
unsigned long                        bEnableColorScaling : 1;              // 0x027C (0x0004)
[0x0000000000000000] [0x00000020]
unsigned long                        bEnableColorScaleInterp : 1;          // 0x027C (0x0004)
[0x0000000000000000] [0x00000040]
unsigned long                        bUseClientSideCameraUpdates : 1;      // 0x027C
(0x0004) [0x0000000000000000] [0x00000080]
unsigned long                        bDebugClientSideCamera : 1;           // 0x027C (0x0004)
[0x0000000000000000] [0x00000100]
unsigned long                        bShouldSendClientSideCameraUpdate : 1;     // 0x027C
(0x0004) [0x0000000000000000] [0x00000200]
float                                LockedFOV;                            // 0x0280 (0x0004)
[0x0000000000000000]
float                                ConstrainedAspectRatio;               // 0x0284 (0x0004)
[0x0000000000000000]
float                                DefaultAspectRatio;                   // 0x0288 (0x0004)
[0x0000000000000000]
float                                OffAxisYawAngle;                      // 0x028C (0x0004)
[0x0000000000000000]
float                                OffAxisPitchAngle;                    // 0x0290 (0x0004)
[0x0000000000000000]
struct FColor                        FadeColor;                            // 0x0294 (0x0004)
[0x0000000000000000]
float                                FadeAmount;                           // 0x0298 (0x0004)
[0x0000000000000000]
float                                CamOverridePostProcessAlpha;          // 0x029C (0x0004)
[0x0000000000000000]
struct FPostProcessSettings                 CamPostProcessSettings;               // 0x02A0
(0x0168) [0x0000000000400000] (CPF_NeedCtorLink)
struct FRenderingPerformanceOverrides       RenderingOverrides;                   // 0x0408
(0x0004) [0x0000000000000000]
struct FVector                       ColorScale;                           // 0x040C (0x000C)
[0x0000000000000000]
```

```
struct FVector                          DesiredColorScale;                      // 0x0418 (0x000C)
[0x0000000000000000]
struct FVector                          OriginalColorScale;                     // 0x0424 (0x000C)
[0x0000000000000000]
float                              ColorScaleInterpDuration;                // 0x0430 (0x0004)
[0x0000000000000000]
float                              ColorScaleInterpStartTime;               // 0x0434 (0x0004)
[0x0000000000000000]
struct FTCameraCache                         CameraCache;                        // 0x0438 (0x0020)
[0x0000000000000000]
struct FTCameraCache                         LastFrameCameraCache;                    // 0x0458
(0x0020) [0x0000000000000000]
struct FTViewTarget                        ViewTarget;                          // 0x0478 (0x0038)
[0x0000000000000000]
struct FTViewTarget                        PendingViewTarget;                      // 0x04B0 (0x0038)
[0x0000000000000000]
float                              BlendTimeToGo;                        // 0x04E8 (0x0004)
[0x0000000000000000]
struct FViewTargetTransitionParams              BlendParams;                            // 0x04EC
(0x0010) [0x0000000000000000]
TArray<class UCameraModifier*>                ModifierList;                         // 0x0500 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
float                              FreeCamDistance;                       // 0x0510 (0x0004)
[0x0000000000000000]
struct FVector                          FreeCamOffset;                         // 0x0514 (0x000C)
[0x0000000000000000]
struct FVector2D                          FadeAlpha;                            // 0x0520 (0x0008)
[0x0000000000000000]
float                              FadeTime;                            // 0x0528 (0x0004)
[0x0000000000000000]
float                              FadeTimeRemaining;                      // 0x052C (0x0004)
[0x0000000000000000]
TArray<class AEmitterCameraLensEffectBase*>       CameraLensEffects;                      //
0x0530 (0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
class UCameraModifier_CameraShake*              CameraShakeCamMod;                      //
0x0540 (0x0008) [0x0000000004002001] (CPF_Edit | CPF_Transient | CPF_EditInline)
class UClass*                          CameraShakeCamModClass;                  // 0x0548
(0x0008) [0x0000000000000001] (CPF_Edit)
class UCameraAnimInst*                      AnimInstPool[0x8];                    // 0x0550 (0x0040)
[0x0000000000000000]
TArray<class UCameraAnimInst*>                ActiveAnims;                          // 0x0590
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<class UCameraAnimInst*>                FreeAnims;                            // 0x05A0
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
class ADynamicCameraActor*                    AnimCameraActor;                       // 0x05B0
(0x0008) [0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.Camera");
}

return uClassPointer;
};

void SetCameraFade(unsigned long bNewEnableFading, struct FColor NewFadeColor, struct
FVector2D NewFadeAlpha, float NewFadeTime, unsigned long bNewFadeAudio);
void StopCameraAnim(class UCameraAnimInst* AnimInst, unsigned long bImmediate);
void StopAllCameraAnimsByType(class UCameraAnim* Anim, unsigned long bImmediate);
void StopAllCameraAnims(unsigned long bImmediate);
class UCameraAnimInst* PlayCameraAnim(class UCameraAnim* Anim, float Rate, float Scale,
float BlendInTime, float BlendOutTime, unsigned long bLoop, unsigned long bRandomStartTime,
float Duration, unsigned long bSingleInstance);
void ClearAllCameraShakes();
static void PlayWorldCameraShake(class UCameraShake* Shake, class AActor* ShakeInstigator,
struct FVector Epicenter, float InnerRadius, float OuterRadius, float Falloff, unsigned long
bTryForceFeedback, unsigned long bOrientShakeTowardsEpicenter);
static float CalcRadialShakeScale(class ACamera* Cam, struct FVector Epicenter, float
InnerRadius, float OuterRadius, float Falloff);
void StopCameraShake(class UCameraShake* Shake);
void PlayCameraShake(class UCameraShake* Shake, float Scale, uint8_t PlaySpace, struct
FRotator UserPlaySpaceRot);
void ClearCameraLensEffects();
void RemoveCameraLensEffect(class AEmitterCameraLensEffectBase* Emitter);
void AddCameraLensEffect(class UClass* LensEffectEmitterClass);
class AEmitterCameraLensEffectBase* FindCameraLensEffect(class UClass*
LensEffectEmitterClass);
void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void ProcessViewRotation(float DeltaTime, struct FRotator& OutViewRotation, struct FRotator&
OutDeltaRot);
void SetViewTarget(class AActor* NewViewTarget, struct FViewTargetTransitionParams
TransitionParams);
void UpdateViewTarget(float DeltaTime, struct FTViewTarget& OutVT);
void CheckViewTarget(struct FTViewTarget& VT);
void FillCameraCache(struct FTPOV& NewPOV);
struct FTPOV BlendViewTargets(float Alpha, struct FTViewTarget& A, struct FTViewTarget& B);
void ApplyAudioFade();
void UpdateFade(float DeltaTime);
void DoUpdateCamera(float DeltaTime);
void eventUpdateCamera(float DeltaTime);
void SetDesiredColorScale(struct FVector NewColorScale, float InterpTime);
struct FRotator GetCameraRotation();
void GetCameraViewPoint(struct FVector& OutCamLoc, struct FRotator& OutCamRot);
void SetFOV(float NewFOV);
float GetFOVAngle();
void InitializeFor(class APlayerController* PC);
void ApplyCameraModifiers(float DeltaTime, struct FTPOV& OutPOV);
void eventDestroyed();
void PostBeginPlay();
class UCameraModifier* CreateCameraModifier(class UClass* ModifierClass);
};

// Class Engine.CameraActor
// 0x0190 (0x0268 - 0x03F8)
class ACameraActor : public AActor
{
public:
unsigned long                          bConstrainAspectRatio : 1;                // 0x0268 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                          bCamOverridePostProcess : 1;              // 0x0268 (0x0004)
[0x0000000020000000] [0x00000002]  CPF_Deprecated)
float                                  AspectRatio;                              // 0x026C (0x0004)
[0x0000000200000021] (CPF_Edit | CPF_Net)
float                                  FOVAngle;                                 // 0x0270 (0x0004)
[0x0000000200000021] (CPF_Edit | CPF_Net)
float                                  CamOverridePostProcessAlpha;              // 0x0274 (0x0004)
[0x0000000200000001] (CPF_Edit)
struct FPostProcessSettings                  CamOverridePostProcess;                  // 0x0278
(0x0168) [0x0000000200400001] (CPF_Edit | CPF_NeedCtorLink)
class UCameraModifier_CameraShake*             CameraShakeCamMod;                      //
0x03E0 (0x0008) [0x0000000004400009] (CPF_Edit | CPF_ExportObject | CPF_NeedCtorLink |
CPF_EditInline)
class UDrawFrustumComponent*                   DrawFrustum;                            // 0x03E8
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)
class UStaticMeshComponent*                    MeshComp;                               // 0x03F0
(0x0008) [0x0000000004080008] (CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CameraActor");
}

return uClassPointer;
};

void DisplayDebug(class AHUD* HUD, float& out_YL, float& out_YPos);
void GetCameraView(float DeltaTime, struct FTPOV& OutPOV);
};

// Class Engine.DynamicCameraActor
// 0x0000 (0x03F8 - 0x03F8)
class ADynamicCameraActor : public ACameraActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.DynamicCameraActor");
}

return uClassPointer;
};

};

// Class Engine.CameraAnim
// 0x01A0 (0x0060 - 0x0200)
class UCameraAnim : public UObject
{
public:
class UInterpGroupCamera*            CameraInterpGroup;              // 0x0060
(0x0008) [0x0000000000000000]
class UInterpGroup*                  PreviewInterpGroup;             // 0x0068 (0x0008)
[0x0000000800002000] (CPF_Transient)
float                                AnimLength;                     // 0x0070 (0x0004)
[0x0000000000000002] (CPF_Const)
struct FBox                          BoundingBox;                    // 0x0074 (0x001C)
[0x0000000000000002] (CPF_Const)
struct FPostProcessSettings          BasePPSettings;                 // 0x0090 (0x0168)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
float                                BasePPSettingsAlpha;            // 0x01F8 (0x0004)
[0x0000000000000002] (CPF_Const)
float                                BaseFOV;                        // 0x01FC (0x0004)
[0x0000000000000002] (CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CameraAnim");
}

return uClassPointer;
};

};

// Class Engine.CameraAnimInst
// 0x0218 (0x0060 - 0x0278)
class UCameraAnimInst : public UObject
{
public:
class UCameraAnim*                   CamAnim;                        // 0x0060 (0x0008)
[0x0000000000000000]
class UInterpGroupInst*              InterpGroupInst;                // 0x0068 (0x0008)
[0x0000000004400008] (CPF_ExportObject | CPF_NeedCtorLink | CPF_EditInline)
float                                CurTime;                        // 0x0070 (0x0004)
[0x0000000000002000] (CPF_Transient)
```

```
unsigned long                       bLooping : 1;                       // 0x0074 (0x0004)
[0x0000000000002000] [0x00000001] (CPF_Transient)
unsigned long                       bFinished : 1;                      // 0x0074 (0x0004)
[0x0000000000002000] [0x00000002] (CPF_Transient)
unsigned long                       bAutoReleaseWhenFinished : 1;               // 0x0074 (0x0004)
[0x0000000000002000] [0x00000004] (CPF_Transient)
unsigned long                       bBlendingIn : 1;                     // 0x0074 (0x0004)
[0x0000000000002000] [0x00000008] (CPF_Transient)
unsigned long                       bBlendingOut : 1;                    // 0x0074 (0x0004)
[0x0000000000002000] [0x00000010] (CPF_Transient)
float                          BlendInTime;                        // 0x0078 (0x0004)
[0x0000000000000000]
float                          BlendOutTime;                        // 0x007C (0x0004)
[0x0000000000000000]
float                          CurBlendInTime;                       // 0x0080 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                          CurBlendOutTime;                      // 0x0084 (0x0004)
[0x0000000000002000] (CPF_Transient)
float                          PlayRate;                          // 0x0088 (0x0004)
[0x0000000000000000]
float                          BasePlayScale;                       // 0x008C (0x0004)
[0x0000000000000000]
float                          TransientScaleModifier;                   // 0x0090 (0x0004)
[0x0000000000000000]
float                          CurrentBlendWeight;                    // 0x0094 (0x0004)
[0x0000000000000000]
float                          RemainingTime;                       // 0x0098 (0x0004)
[0x0000000000002000] (CPF_Transient)
class UInterpTrackMove*                  MoveTrack;                        // 0x00A0 (0x0008)
[0x0000000000002000] (CPF_Transient)
class UInterpTrackInstMove*                MoveInst;                         // 0x00A8 (0x0008)
[0x0000000000002000] (CPF_Transient)
class UAnimNodeSequence*                  SourceAnimNode;                       // 0x00B0
(0x0008) [0x0000000000002000] (CPF_Transient)
uint8_t                          PlaySpace;                         // 0x00B8 (0x0001)
[0x0000008000000000]
struct FMatrix                       UserPlaySpaceMatrix;                   // 0x00C0 (0x0040)
[0x0000000000002000] (CPF_Transient)
struct FPostProcessSettings                LastPPSettings;                     // 0x0100 (0x0168)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
float                          LastPPSettingsAlpha;                  // 0x0268 (0x0004)
[0x0000000000002000] (CPF_Transient)
struct FVector                       LastCameraLoc;                     // 0x026C (0x000C)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CameraAnimInst");
}
return uClassPointer;
};

void SetPlaySpace(uint8_t NewSpace, struct FRotator UserPlaySpace);
void ApplyTransientScaling(float Scalar);
void Stop(unsigned long bImmediate);
void AdvanceAnim(float DeltaTime, unsigned long bJump);
void Update(float NewRate, float NewScale, float NewBlendInTime, float NewBlendOutTime, float
NewDuration);
void Play(class UCameraAnim* Anim, class AActor* CamActor, float InRate, float InScale, float
InBlendInTime, float InBlendOutTime, unsigned long bInLoop, unsigned long bRandomStartTime,
float Duration);
};

// Class Engine.CameraModifier
// 0x0024 (0x0060 - 0x0084)
class UCameraModifier : public UObject
{
public:
unsigned long                  bDisabled : 1;                 // 0x0060 (0x0004)
[0x0000000000000000] [0x00000001]
unsigned long                  bPendingDisable : 1;               // 0x0060 (0x0004)
[0x0000000000000000] [0x00000002]
unsigned long                  bExclusive : 1;                // 0x0060 (0x0004)
[0x0000000000000000] [0x00000004]
unsigned long                  bDebug : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
class ACamera*                  CameraOwner;                      // 0x0068 (0x0008)
[0x0000000000000000]
uint8_t                       Priority;                    // 0x0070 (0x0001)
[0x0000000000000000]
float                        AlphaInTime;                   // 0x0074 (0x0004)
[0x0000000000000000]
float                        AlphaOutTime;                   // 0x0078 (0x0004)
[0x0000000000000000]
float                        Alpha;                       // 0x007C (0x0004)
[0x0000000000002000] (CPF_Transient)
float                        TargetAlpha;                   // 0x0080 (0x0004)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CameraModifier");
}

return uClassPointer;
};
```

```
void UpdateAlpha(class ACamera* Camera, float DeltaTime);
bool ProcessViewRotation(class AActor* ViewTarget, float DeltaTime, struct FRotator&
out_ViewRotation, struct FRotator& out_DeltaRot);
void ToggleModifier();
void EnableModifier();
void eventDisableModifier(unsigned long bImmediate);
bool RemoveCameraModifier(class ACamera* Camera);
bool AddCameraModifier(class ACamera* Camera);
bool IsDisabled();
bool ModifyCamera(class ACamera* Camera, float DeltaTime, struct FTPOV& OutPOV);
void Init();
};

// Class Engine.CameraModifier_CameraShake
// 0x0018 (0x0084 - 0x009C)
class UCameraModifier_CameraShake : public UCameraModifier
{
public:
TArray<struct FCameraShakeInstance>          ActiveShakes;                        // 0x0088
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
float                              SplitScreenShakeScale;                  // 0x0098 (0x0004)
[0x0000000000000003] (CPF_Edit | CPF_Const)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CameraModifier_CameraShake");
}

return uClassPointer;
};

bool ModifyCamera(class ACamera* Camera, float DeltaTime, struct FTPOV& OutPOV);
void UpdateCameraShake(float DeltaTime, struct FCameraShakeInstance& Shake, struct
FTPOV& OutPOV);
void RemoveAllCameraShakes();
void RemoveCameraShake(class UCameraShake* Shake);
void AddCameraShake(class UCameraShake* NewShake, float Scale, uint8_t PlaySpace, struct
FRotator UserPlaySpaceRot);
struct FCameraShakeInstance InitializeShake(class UCameraShake* NewShake, float Scale,
uint8_t PlaySpace, struct FRotator UserPlaySpaceRot);
void ReinitShake(int32_t ActiveShakeIdx, float Scale);
static float InitializeOffset(struct FFOscillator& Param);
};

// Class Engine.CameraShake
// 0x0084 (0x0060 - 0x00E4)
class UCameraShake : public UObject
{
public:
```

```
unsigned long                        bSingleInstance : 1;                    // 0x0060 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                        bRandomAnimSegment : 1;                 // 0x0060 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                            OscillationDuration;                    // 0x0064 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            OscillationBlendInTime;                 // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            OscillationBlendOutTime;                // 0x006C (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FROscillator                  RotOscillation;                     // 0x0070 (0x0024)
[0x0000000000000001] (CPF_Edit)
struct FVOscillator                  LocOscillation;                     // 0x0094 (0x0024)
[0x0000000000000001] (CPF_Edit)
struct FFOscillator                  FOVOscillation;                     // 0x00B8 (0x000C)
[0x0000000000000001] (CPF_Edit)
class UCameraAnim*                      Anim;                            // 0x00C8 (0x0008)
[0x0000000000000001] (CPF_Edit)
float                            AnimPlayRate;                           // 0x00D0 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AnimScale;                              // 0x00D4 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AnimBlendInTime;                        // 0x00D8 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            AnimBlendOutTime;                       // 0x00DC (0x0004)
[0x0000000000000001] (CPF_Edit)
float                            RandomAnimSegmentDuration;              // 0x00E0 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CameraShake");
}

return uClassPointer;
};

class APlayerController* GetAPC();
void ToggleShake(unsigned long bShake);
void eventOnPropertyChanged();
struct FFOscillator GetRandomFOscillator(float AmplitudeScale, float FrequencyScale);
struct FVOscillator GetRandomLocOscillation(float AmplitudeScale, float FrequencyScale);
struct FROscillator GetRandomROscillator(float AmplitudeScale, float FrequencyScale);
void SetRandomLocShake(float AmplitudeScale, float FrequencyScale);
void SetRandomRotShake(float AmplitudeScale, float FrequencyScale);
class UCameraShake* CreateScaled(float AmplitudeScale, float FrequencyScale, float
DurationScale);
float GetLocOscillationMagnitude();
float GetRotOscillationMagnitude();
```

```
};

// Class Engine.CloudStorageUpgradeHelper
// 0x0000 (0x0060 - 0x0060)
class UCloudStorageUpgradeHelper : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CloudStorageUpgradeHelper");
}

return uClassPointer;
};

void eventGetCloudUpgradeKeys(TArray<class FString>& CloudKeys);
void eventHandleLocalKeyValue(class FString& CloudKeyName, struct FPlatformInterfaceData&
CloudValue, int32_t& bShouldMoveToCloud, int32_t& bShouldDeleteLocalKey);
void eventHandleLocalDocument(class FString& DocName, int32_t& bShouldMoveToCloud,
int32_t& bShouldDeleteLocalFile);
};

// Class Engine.AnalyticEventsBase
// 0x0018 (0x0088 - 0x00A0)
class UAnalyticEventsBase : public UPlatformInterfaceBase
{
public:
unsigned long                        bSessionInProgress : 1;                // 0x0088 (0x0004)
[0x0000000000000002] [0x00000001] (CPF_Const)
unsigned long                        bAutoStartSession : 1;                 // 0x0088 (0x0004)
[0x0000000000004000] [0x00000002] (CPF_Config)
int32_t                              SessionPauseThresholdSec;              // 0x008C (0x0004)
[0x0000000000004000] (CPF_Config)
class FString                        UserId;                                // 0x0090 (0x0010)
[0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnalyticEventsBase");
}

return uClassPointer;
};
```

```
void eventSendCachedEvents();
void eventLogCurrencyGivenEvent(class FString GameCurrencyType, int32_t
GameCurrencyAmount);
void eventLogCurrencyPurchaseEvent(class FString GameCurrencyType, int32_t
GameCurrencyAmount, class FString RealCurrencyType, float RealMoneyCost, class FString
PaymentProvider);
void eventLogItemPurchaseEvent(class FString ItemID, class FString Currency, int32_t
PerItemCost, int32_t ItemQuantity);
void eventLogUserAttributeUpdateArray(TArray<struct FEventStringParam> AttributeArray);
void eventLogUserAttributeUpdate(class FString AttributeName, class FString AttributeValue);
void eventLogErrorMessage(class FString ErrorName, class FString ErrorMessage);
void eventEndStringEventParamArray(class FString EventName, TArray<struct
FEventStringParam> ParamArray);
void eventLogStringEventParamArray(class FString EventName, TArray<struct
FEventStringParam> ParamArray, unsigned long bTimed);
void eventEndStringEventParam(class FString EventName, class FString ParamName, class
FString ParamValue);
void eventLogStringEventParam(class FString EventName, class FString ParamName, class
FString ParamValue, unsigned long bTimed);
void eventEndStringEvent(class FString EventName);
void eventLogStringEvent(class FString EventName, unsigned long bTimed);
void eventEndSession();
void eventStartSession();
void eventSetUserId(class FString NewUserId);
void eventInit();
bool IsSessionInProgress();
};

// Class Engine.MultiProviderAnalytics
// 0x0020 (0x00A0 - 0x00C0)
class UMultiProviderAnalytics : public UAnalyticEventsBase
{
public:
TArray<class FString>                    AnalyticsProviderClassNames;             // 0x00A0
(0x0010) [0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<class UAnalyticEventsBase*>             AnalyticsProviders;                  // 0x00B0
(0x0010) [0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MultiProviderAnalytics");
}

return uClassPointer;
};

void eventSendCachedEvents();
void eventLogCurrencyGivenEvent(class FString GameCurrencyType, int32_t
GameCurrencyAmount);
void eventLogCurrencyPurchaseEvent(class FString GameCurrencyType, int32_t
GameCurrencyAmount, class FString RealCurrencyType, float RealMoneyCost, class FString
PaymentProvider);
void eventLogItemPurchaseEvent(class FString ItemID, class FString Currency, int32_t
PerItemCost, int32_t ItemQuantity);
void eventLogUserAttributeUpdateArray(TArray<struct FEventStringParam> AttributeArray);
void eventLogUserAttributeUpdate(class FString AttributeName, class FString AttributeValue);
void eventLogErrorMessage(class FString ErrorName, class FString ErrorMessage);
void eventEndStringEventParamArray(class FString EventName, TArray<struct
FEventStringParam> ParamArray);
void eventLogStringEventParamArray(class FString EventName, TArray<struct
FEventStringParam> ParamArray, unsigned long bTimed);
void eventEndStringEventParam(class FString EventName, class FString ParamName, class
FString ParamValue);
void eventLogStringEventParam(class FString EventName, class FString ParamName, class
FString ParamValue, unsigned long bTimed);
void eventEndStringEvent(class FString EventName);
void eventLogStringEvent(class FString EventName, unsigned long bTimed);
void eventEndSession();
void eventStartSession();
void eventSetUserId(class FString NewUserId);
void Init();
};

// Class Engine.AppNotificationsBase
// 0x0068 (0x0088 - 0x00F0)
class UAppNotificationsBase : public UPlatformInterfaceBase
{
public:
struct FLaunchNotificationInfo          AppLaunchNotification;              // 0x0088
(0x0038) [0x0000000000400002] (CPF_Const | CPF_NeedCtorLink)
struct FScriptDelegate          __OnReceivedLocalNotification__Delegate;      // 0x00C0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate          __OnReceivedRemoteNotification__Delegate;     // 0x00D8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AppNotificationsBase");
}

return uClassPointer;
};

void DebugLogNotification(struct FNotificationInfo& Notification);
void OnReceivedRemoteNotification(unsigned long bWasAppActive, struct FNotificationInfo&
Notification);
void OnReceivedLocalNotification(unsigned long bWasAppActive, struct FNotificationInfo&
Notification);
void CancelAllScheduledLocalNotifications();
void ScheduleLocalNotification(int32_t StartOffsetSeconds, struct FNotificationInfo&
Notification);
bool WasLaunchedViaNotification();
void eventInit();
};

// Class Engine.CloudStorageBase
// 0x0014 (0x0088 - 0x009C)
class UCloudStorageBase : public UPlatformInterfaceBase
{
public:
TArray<class FString>                    LocalCloudFiles;                    // 0x0088 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
unsigned long                            bSuppressDelegateCalls : 1;            // 0x0098 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CloudStorageBase");
}

return uClassPointer;
};

bool eventUpgradeLocalStorageToCloud(class UCloudStorageUpgradeHelper* UpgradeHelper,
unsigned long bForceSearchAgain);
bool eventResolveConflictWithVersionIndex(int32_t Index);
bool eventResolveConflictWithNewestDocument();
bool eventWaitForWritesToFinish(float MaxTimeSeconds);
bool eventIsStillWritingFiles();
bool eventSaveDocumentWithObject(int32_t Index, class UObject* ObjectData, int32_t
SaveVersion);
bool eventSaveDocumentWithBytes(int32_t Index, TArray<uint8_t> ByteData);
bool eventSaveDocumentWithString(int32_t Index, class FString StringData);
bool eventWriteCloudDocument(int32_t Index);
class UObject* eventParseDocumentAsObject(int32_t Index, class UClass* ObjectClass, int32_t
ExpectedVersion, unsigned long bIsForConflict);
void eventParseDocumentAsBytes(int32_t Index, unsigned long bIsForConflict, TArray<uint8_t>&
ByteData);
class FString eventParseDocumentAsString(int32_t Index, unsigned long bIsForConflict);
bool eventReadCloudDocument(int32_t Index, unsigned long bIsForConflict);
void eventDeleteAllCloudDocuments();
int32_t eventCreateCloudDocument(class FString Filename);
class FString eventGetCloudDocumentName(int32_t Index);
int32_t eventGetNumCloudDocuments(unsigned long bIsForConflict);
bool eventQueryForCloudDocuments();
bool eventWriteKeyValue(class FString KeyName, struct FPlatformInterfaceData& Value);
bool eventReadKeyValueFromLocalStore(class FString KeyName, uint8_t Type, struct
FPlatformInterfaceDelegateResult& Value);
bool eventReadKeyValue(class FString KeyName, uint8_t Type, struct
FPlatformInterfaceDelegateResult& Value);
bool IsUsingLocalStorage();
void eventInit();
};

// Class Engine.FacebookIntegration
// 0x0060 (0x0088 - 0x00E8)
class UFacebookIntegration : public UPlatformInterfaceBase
{
public:
class FString                    AppID;                    // 0x0088 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
TArray<class FString>                Permissions;                // 0x0098 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                    Username;                   // 0x00A8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    UserId;                   // 0x00B8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
class FString                    AccessToken;                 // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FFacebookFriend>            FriendsList;                 // 0x00D8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FacebookIntegration");
}

return uClassPointer;
};

void eventDisconnect();
void eventFacebookDialog(class FString Action, TArray<class FString> ParamKeysAndValues);
void eventFacebookRequest(class FString GraphRequest, class FString HTTPMethod,
TArray<class FString> ParamKeysAndValues);
bool eventIsAuthorized();
bool eventAuthorize();
bool eventInit();
};

// Class Engine.InAppMessageBase
// 0x0000 (0x0088 - 0x0088)
class UInAppMessageBase : public UPlatformInterfaceBase
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InAppMessageBase");
}

return uClassPointer;
};

bool eventShowInAppEmailUI(class FString InitialSubject, class FString InitialMessage);
bool eventShowInAppSMSUI(class FString InitialMessage);
void eventInit();
};

// Class Engine.InGameAdManager
// 0x0004 (0x0088 - 0x008C)
class UInGameAdManager : public UPlatformInterfaceBase
{
public:
unsigned long                      bShouldPauseWhileAdOpen : 1;           // 0x0088 (0x0004)
[0x0000000000000000] [0x00000001]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InGameAdManager");
}

return uClassPointer;
};

void SetPauseWhileAdOpen(unsigned long bShouldPause);
void ForceCloseAd();
void HideBanner();
void ShowBanner(unsigned long bShowBottomOfScreen);
void eventInit();
};

// Class Engine.TwitterIntegrationBase
// 0x0000 (0x0088 - 0x0088)
class UTwitterIntegrationBase : public UPlatformInterfaceBase
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TwitterIntegrationBase");
}

return uClassPointer;
};

bool eventTwitterRequest(class FString URL, TArray<class FString> ParamKeysAndValues,
uint8_t RequestMethod, int32_t AccountIndex);
class FString eventGetAccountId(int32_t AccountIndex);
class FString eventGetAccountName(int32_t AccountIndex);
int32_t eventGetNumAccounts();
bool eventAuthorizeAccounts();
bool eventShowTweetUI(class FString InitialMessage, class FString URL, class FString Picture);
bool eventCanShowTweetUI();
void eventInit();
};

// Class Engine.PlatformInterfaceWebResponse
// 0x0088 (0x0060 - 0x00E8)
class UPlatformInterfaceWebResponse : public UObject
{
public:
class FString                          OriginalURL;                          // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                ResponseCode;                         // 0x0070 (0x0004)
[0x0000000000000000]
int32_t                                Tag;                              // 0x0074 (0x0004)
[0x0000000000000000]
struct FMap_Mirror                         Headers;                             // 0x0078 (0x0050)
[0x0000000000001000] (CPF_Native)
class FString                          StringResponse;                       // 0x00C8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<uint8_t>                          BinaryResponse;                       // 0x00D8 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PlatformInterfaceWebResponse");
}

return uClassPointer;
};

class FString GetHeaderValue(class FString HeaderName);
```

```
void GetHeader(int32_t HeaderIndex, class FString& Header, class FString& Value);
int32_t GetNumHeaders();
};

// Class Engine.NetDriverSecurity
// 0x0010 (0x0060 - 0x0070)
class UNetDriverSecurity : public UObject
{
public:
uint8_t                    UnknownData00[0x10];              // 0x0060 (0x0010)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NetDriverSecurity");
}

return uClassPointer;
};

};

// Class Engine.NetConnectionEncryptor
// 0x00A8 (0x0060 - 0x0108)
class UNetConnectionEncryptor : public UObject
{
public:
uint8_t                    UnknownData00[0xA8];              // 0x0060 (0x00A8)
MISSED OFFSET

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NetConnectionEncryptor");
}

return uClassPointer;
};

};

// Class Engine.__ScriptGroup_ORS__CreateObjects_0x1
// 0x0008 (0x0060 - 0x0068)
class U__ScriptGroup_ORS__CreateObjects_0x1 : public UObject
{
```

```
public:
class UObject*                              ObjOuter;                                // 0x0060 (0x0008)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.__ScriptGroup_ORS__CreateObjects_0x1");
}

return uClassPointer;
};

void __ScriptGroup_ORS__CreateObjects_0x1(class UClass* C);
};

// Class Engine.SeqEvent_HitWall
// 0x0004 (0x017C - 0x0180)
class USeqEvent_HitWall : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_HitWall");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_Destroy
// 0x0018 (0x0160 - 0x0178)
class USeqAct_Destroy : public USequenceAction
{
public:
unsigned long                              bDestroyBasedActors : 1;                   // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<class UClass*>                          IgnoreBasedClasses;                         // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
```

```
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Destroy");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_Teleport
// 0x0018 (0x0160 - 0x0178)
class USeqAct_Teleport : public USequenceAction
{
public:
unsigned long                         bUpdateRotation : 1;                    // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                         bCheckOverlap : 1;                      // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                                 TeleportDistance;                       // 0x0164 (0x0004)
[0x0000000000000001] (CPF_Edit)
TArray<class AVolume*>                TeleportVolumes;                        // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_Teleport");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
static bool ShouldTeleport(class AActor* TestActor, struct FVector TeleportLocation, float
TeleportDist, TArray<class AVolume*> Volumes);
};

// Class Engine.SeqAct_SetVelocity
// 0x0014 (0x0160 - 0x0174)
class USeqAct_SetVelocity : public USequenceAction
{
public:
struct FVector                        VelocityDir;                            // 0x0160 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                                 VelocityMag;                            // 0x016C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                         bVelocityRelativeToActorRotation : 1;   // 0x0170
```

```
(0x0004) [0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetVelocity");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_ToggleHidden
// 0x0018 (0x0160 - 0x0178)
class USeqAct_ToggleHidden : public USeqAct_Toggle
{
public:
unsigned long                    bToggleBasedActors : 1;              // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
TArray<class UClass*>                IgnoreBasedClasses;                // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleHidden");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_AttachToActor
// 0x0024 (0x0160 - 0x0184)
class USeqAct_AttachToActor : public USequenceAction
{
public:
unsigned long                    bDetach : 1;                         // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bHardAttach : 1;                     // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bUseRelativeOffset : 1;              // 0x0160 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
```

```
unsigned long                         bUseRelativeRotation : 1;               // 0x0160 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)
struct FName                          BoneName;                               // 0x0164 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FVector                        RelativeOffset;                         // 0x016C (0x000C)
[0x0000000000000001] (CPF_Edit)
struct FRotator                       RelativeRotation;                       // 0x0178 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AttachToActor");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqEvent_MobileTouch
// 0x0004 (0x017C - 0x0180)
class USeqEvent_MobileTouch : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_MobileTouch");
}

return uClassPointer;
};

};

// Class Engine.ApexDestructibleActorSpawnable
// 0x0000 (0x02C8 - 0x02C8)
class AApexDestructibleActorSpawnable : public AApexDestructibleActor
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ApexDestructibleActorSpawnable");
}

return uClassPointer;
};

};

// Class Engine.EmitterSpawnable
// 0x000C (0x027C - 0x0288)
class AEmitterSpawnable : public AEmitter
{
public:
class UParticleSystem*                    ParticleTemplate;                    // 0x0280 (0x0008)
[0x0000000100000020] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EmitterSpawnable");
}

return uClassPointer;
};

void eventReplicatedEvent(struct FName VarName);
void eventSetTemplate(class UParticleSystem* NewTemplate, unsigned long bDestroyOnFinish);
};

// Class Engine.KAssetSpawnable
// 0x0000 (0x0288 - 0x0288)
class AKAssetSpawnable : public AKAsset
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.KAssetSpawnable");
}
return uClassPointer;
};

};

// Class Engine.ActorFactorySkeletalMeshCinematic
// 0x0000 (0x00B8 - 0x00B8)
class UActorFactorySkeletalMeshCinematic : public UActorFactorySkeletalMesh
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactorySkeletalMeshCinematic");
}

return uClassPointer;
};

};

// Class Engine.ActorFactorySkeletalMeshMAT
// 0x0000 (0x00B8 - 0x00B8)
class UActorFactorySkeletalMeshMAT : public UActorFactorySkeletalMesh
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ActorFactorySkeletalMeshMAT");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ToggleGodMode
// 0x0000 (0x0160 - 0x0160)
class USeqAct_ToggleGodMode : public USequenceAction
{
public:

public:
```

```
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleGodMode");
}

return uClassPointer;
};

};

// Class Engine.SplineComponentSimplified
// 0x0000 (0x02A0 - 0x02A0)
class USplineComponentSimplified : public USplineComponent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SplineComponentSimplified");
}

return uClassPointer;
};

};

// Class Engine.AmbientSoundSimpleSplineNonLoop
// 0x0004 (0x02A4 - 0x02A8)
class AAmbientSoundSimpleSplineNonLoop : public AAmbientSoundSimpleSpline
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AmbientSoundSimpleSplineNonLoop");
}

return uClassPointer;
};
```

```
};

// Class Engine.AnimNotify_PlayFaceFXAnim
// 0x0040 (0x0068 - 0x00A8)
class UAnimNotify_PlayFaceFXAnim : public UAnimNotify_Scripted
{
public:
class UFaceFXAnimSet*                    FaceFXAnimSetRef;                         // 0x0068
(0x0008) [0x0000000000000001] (CPF_Edit)
class FString                            GroupName;                                // 0x0070 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class FString                            AnimName;                                 // 0x0080 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
class USoundCue*                         SoundCueToPlay;                           // 0x0090 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UAkEvent*                          AkEventToPlay;                            // 0x0098 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                            bOverridePlayingAnim : 1;                 // 0x00A0 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
float                                    PlayFrequency;                            // 0x00A4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.AnimNotify_PlayFaceFXAnim");
}

return uClassPointer;
};

void eventNotify(class AActor* Owner, class UAnimNodeSequence* AnimSeqInstigator);
};

// Class Engine.OnlineTitleFileCacheInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineTitleFileCacheInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineTitleFileCacheInterface");
}
return uClassPointer;
};

bool DeleteTitleFile(class FString Filename);
bool DeleteTitleFiles(float MaxAgeSeconds);
bool ClearCachedFile(class FString Filename);
bool ClearCachedFiles();
class FString GetTitleFileLogicalName(class FString Filename);
class FString GetTitleFileHash(class FString Filename);
uint8_t GetTitleFileState(class FString Filename);
bool GetTitleFileContents(class FString Filename, TArray<uint8_t>& FileContents);
void ClearSaveTitleFileCompleteDelegate(struct FScriptDelegate SaveCompleteDelegate);
void AddSaveTitleFileCompleteDelegate(struct FScriptDelegate SaveCompleteDelegate);
void OnSaveTitleFileComplete(unsigned long bWasSuccessful, class FString Filename);
bool SaveTitleFile(class FString Filename, class FString LogicalName, TArray<uint8_t>
FileContents);
void ClearLoadTitleFileCompleteDelegate(struct FScriptDelegate LoadCompleteDelegate);
void AddLoadTitleFileCompleteDelegate(struct FScriptDelegate LoadCompleteDelegate);
void OnLoadTitleFileComplete(unsigned long bWasSuccessful, class FString Filename);
bool LoadTitleFile(class FString Filename);
};

// Class Engine.OnlineTitleFileInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineTitleFileInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineTitleFileInterface");
}

return uClassPointer;
};

void ClearRequestTitleFileListCompleteDelegate(struct FScriptDelegate
RequestTitleFileListDelegate);
void AddRequestTitleFileListCompleteDelegate(struct FScriptDelegate
RequestTitleFileListDelegate);
void OnRequestTitleFileListComplete(unsigned long bWasSuccessful, TArray<class FString>
ResultStr);
bool RequestTitleFileList();
bool ClearDownloadedFile(class FString Filename);
bool ClearDownloadedFiles();
uint8_t GetTitleFileState(class FString Filename);
bool GetTitleFileContents(class FString Filename, TArray<uint8_t>& FileContents);
void ClearReadTitleFileCompleteDelegate(struct FScriptDelegate
ReadTitleFileCompleteDelegate);
```

```
void AddReadTitleFileCompleteDelegate(struct FScriptDelegate ReadTitleFileCompleteDelegate);
bool ReadTitleFile(class FString FileToRead, uint8_t FileType);
void OnReadTitleFileComplete(unsigned long bWasSuccessful, class FString Filename);
};

// Class Engine.UserCloudFileInterface
// 0x0000 (0x0060 - 0x0060)
class UUserCloudFileInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UserCloudFileInterface");
}

return uClassPointer;
};

void ClearAllDelegates();
void ClearDeleteUserFileCompleteDelegate(struct FScriptDelegate
DeleteUserFileCompleteDelegate);
void AddDeleteUserFileCompleteDelegate(struct FScriptDelegate
DeleteUserFileCompleteDelegate);
bool DeleteUserFile(class FString UserId, class FString Filename, unsigned long
bShouldCloudDelete, unsigned long bShouldLocallyDelete);
void OnDeleteUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void ClearWriteUserFileCompleteDelegate(struct FScriptDelegate
WriteUserFileCompleteDelegate);
void AddWriteUserFileCompleteDelegate(struct FScriptDelegate
WriteUserFileCompleteDelegate);
bool WriteUserFile(class FString UserId, class FString Filename, TArray<uint8_t>& FileContents);
void OnWriteUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void ClearReadUserFileCompleteDelegate(struct FScriptDelegate
ReadUserFileCompleteDelegate);
void AddReadUserFileCompleteDelegate(struct FScriptDelegate
ReadUserFileCompleteDelegate);
bool ReadUserFile(class FString UserId, class FString Filename);
void OnReadUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void GetUserFileList(class FString UserId, TArray<struct FEmsFile>& UserFiles);
void ClearEnumerateUserFileCompleteDelegate(struct FScriptDelegate
EnumerateUserFileCompleteDelegate);
void AddEnumerateUserFileCompleteDelegate(struct FScriptDelegate
EnumerateUserFileCompleteDelegate);
void EnumerateUserFiles(class FString UserId);
void OnEnumerateUserFilesComplete(unsigned long bWasSuccessful, class FString UserId);
```

```
bool ClearFile(class FString UserId, class FString Filename);
bool ClearFiles(class FString UserId);
bool GetFileContents(class FString UserId, class FString Filename, TArray<uint8_t>&
FileContents);
};

// Class Engine.HttpFactory
// 0x0010 (0x0060 - 0x0070)
class UHttpFactory : public UObject
{
public:
class FString                        HttpRequestClassName;                    // 0x0060 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HttpFactory");
}

return uClassPointer;
};

static class UHttpRequestInterface* CreateRequest();
};

// Class Engine.CloudSaveSystemKVSInterface
// 0x0000 (0x0060 - 0x0060)
class UCloudSaveSystemKVSInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CloudSaveSystemKVSInterface");
}

return uClassPointer;
};

bool WriteKeyValue(int32_t SaveSlotIndex, class FString KeyName, struct
FPlatformInterfaceData& Value);
bool ReadKeyValue(int32_t SaveSlotIndex, class FString KeyName, uint8_t Type, struct
FPlatformInterfaceDelegateResult& Value);
};
```

```
// Class Engine.CloudSaveSystemDataBlobStoreInterface
// 0x0000 (0x0060 - 0x0060)
class UCloudSaveSystemDataBlobStoreInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CloudSaveSystemDataBlobStoreInterface");
}

return uClassPointer;
};

bool DeleteDataBlob(class FString StorageID, class FString BlobName, struct FScriptDelegate
InDeleteDataBlobCallback);
void DeleteDataBlobCallbackDelegate(unsigned long bWasSucessfull, class FString StorageID,
class FString BlobName, class FString Error);
void SetDataBlob(class FString StorageID, class FString BlobName, struct FScriptDelegate
InSetDataBlobCallback, TArray<uint8_t>& DataBlob);
void SetDataBlobCallbackDelegate(unsigned long bWasSucessfull, class FString StorageID, class
FString BlobName, class FString Error);
void GetDataBlob(class FString StorageID, class FString BlobName, struct FScriptDelegate
OnGetDataBlobComplete);
void GetDataBlobCallbackDelegate(unsigned long bWasSuccessful, class FString StorageID,
class FString BlobName, class FString Error, TArray<uint8_t>& DataBlob);
};

// Class Engine.CloudStorageBaseCloudSaveSystemKVS
// 0x0008 (0x0060 - 0x0068)
class UCloudStorageBaseCloudSaveSystemKVS : public UObject
{
public:
class UCloudStorageBase*                    CloudStorage;                         // 0x0060 (0x0008)
[0x0000000000002000] (CPF_Transient)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CloudStorageBaseCloudSaveSystemKVS");
}

return uClassPointer;
};
```

```
class FString GenerateKeyNameForSaveSlot(int32_t SaveSlotIndex, class FString KeyName);
bool WriteKeyValue(int32_t SaveSlotIndex, class FString KeyName, struct
FPlatformInterfaceData& Value);
bool ReadKeyValue(int32_t SaveSlotIndex, class FString KeyName, uint8_t Type, struct
FPlatformInterfaceDelegateResult& Value);
void Init(class UCloudStorageBase* InCloudStorage);
};

// Class Engine.ColorScaleVolume
// 0x0014 (0x02A4 - 0x02B8)
class AColorScaleVolume : public AVolume
{
public:
struct FVector                    ColorScale;                    // 0x02A8 (0x000C)
[0x0000000000000001] (CPF_Edit)
float                    InterpTime;                    // 0x02B4 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ColorScaleVolume");
}

return uClassPointer;
};

void eventUnTouch(class AActor* Other);
void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
};

// Class Engine.OnlineGameInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineGameInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineGameInterface");
}

return uClassPointer;
};

void ClearGamePlayersChangedDelegate(struct FScriptDelegate
GamePlayersChangedDelegate);
void AddGamePlayersChangedDelegate(struct FScriptDelegate GamePlayersChangedDelegate);
void OnGamePlayersChanged(struct FName SessionName, TArray<struct FUniqueNetId>
Players);
void ClearJoinMigratedOnlineGameCompleteDelegate(struct FScriptDelegate
JoinMigratedOnlineGameCompleteDelegate);
void AddJoinMigratedOnlineGameCompleteDelegate(struct FScriptDelegate
JoinMigratedOnlineGameCompleteDelegate);
void OnJoinMigratedOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool JoinMigratedOnlineGame(uint8_t PlayerNum, struct FName SessionName, struct
FOnlineGameSearchResult& DesiredGame);
void ClearMigrateOnlineGameCompleteDelegate(struct FScriptDelegate
MigrateOnlineGameCompleteDelegate);
void AddMigrateOnlineGameCompleteDelegate(struct FScriptDelegate
MigrateOnlineGameCompleteDelegate);
void OnMigrateOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool MigrateOnlineGame(uint8_t HostingPlayerNum, struct FName SessionName);
void ClearRecalculateSkillRatingCompleteDelegate(struct FScriptDelegate
RecalculateSkillRatingGameCompleteDelegate);
void AddRecalculateSkillRatingCompleteDelegate(struct FScriptDelegate
RecalculateSkillRatingCompleteDelegate);
void OnRecalculateSkillRatingComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool RecalculateSkillRating(struct FName SessionName, TArray<struct FUniqueNetId>& Players);
bool AcceptGameInvite(uint8_t LocalUserNum, struct FName SessionName);
void ClearGameInviteAcceptedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
GameInviteAcceptedDelegate);
void AddGameInviteAcceptedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
GameInviteAcceptedDelegate);
void OnGameInviteAccepted(class FString ErrorString, struct FOnlineGameSearchResult&
InviteResult);
TArray<struct FOnlineArbitrationRegistrant> GetArbitratedPlayers(struct FName SessionName);
void ClearArbitrationRegistrationCompleteDelegate(struct FScriptDelegate
ArbitrationRegistrationCompleteDelegate);
void AddArbitrationRegistrationCompleteDelegate(struct FScriptDelegate
ArbitrationRegistrationCompleteDelegate);
void OnArbitrationRegistrationComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool RegisterForArbitration(struct FName SessionName);
void ClearEndOnlineGameCompleteDelegate(struct FScriptDelegate
EndOnlineGameCompleteDelegate);
void AddEndOnlineGameCompleteDelegate(struct FScriptDelegate
EndOnlineGameCompleteDelegate);
void OnEndOnlineGameComplete(struct FName SessionName, unsigned long bWasSuccessful);
bool EndOnlineGame(struct FName SessionName);
void ClearStartOnlineGameCompleteDelegate(struct FScriptDelegate
StartOnlineGameCompleteDelegate);
void AddStartOnlineGameCompleteDelegate(struct FScriptDelegate
StartOnlineGameCompleteDelegate);
```

```
void OnStartOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool StartOnlineGame(struct FName SessionName);
void ClearUnregisterPlayerCompleteDelegate(struct FScriptDelegate
UnregisterPlayerCompleteDelegate);
void AddUnregisterPlayerCompleteDelegate(struct FScriptDelegate
UnregisterPlayerCompleteDelegate);
void OnUnregisterPlayerComplete(struct FName SessionName, struct FUniqueNetId PlayerID,
unsigned long bWasSuccessful);
bool UnregisterPlayers(struct FName SessionName, TArray<struct FUniqueNetId>& Players);
bool UnregisterPlayer(struct FName SessionName, struct FUniqueNetId PlayerID);
void ClearRegisterPlayerCompleteDelegate(struct FScriptDelegate
RegisterPlayerCompleteDelegate);
void AddRegisterPlayerCompleteDelegate(struct FScriptDelegate
RegisterPlayerCompleteDelegate);
void OnRegisterPlayerComplete(struct FName SessionName, struct FUniqueNetId PlayerID,
unsigned long bWasSuccessful);
bool RegisterPlayers(struct FName SessionName, TArray<struct FUniqueNetId>& Players);
bool RegisterPlayer(struct FName SessionName, struct FUniqueNetId PlayerID, unsigned long
bWasInvited);
bool GetResolvedConnectString(struct FName SessionName, class FString& ConnectInfo);
void ClearJoinOnlineGameCompleteDelegate(struct FScriptDelegate
JoinOnlineGameCompleteDelegate);
void AddJoinOnlineGameCompleteDelegate(struct FScriptDelegate
JoinOnlineGameCompleteDelegate);
void OnJoinOnlineGameComplete(struct FName SessionName, unsigned long bWasSuccessful);
bool JoinOnlineGame(uint8_t PlayerNum, struct FName SessionName, struct
FOnlineGameSearchResult& DesiredGame);
bool QueryNonAdvertisedData(int32_t StartAt, int32_t NumberToQuery);
bool FreeSearchResults(class UOnlineGameSearch* Search);
class UOnlineGameSearch* GetGameSearch();
bool BindPlatformSpecificSessionToSearch(uint8_t SearchingPlayerNum, class
UOnlineGameSearch* SearchSettings, uint8_t PlatformSpecificInfo);
bool ReadPlatformSpecificSessionInfoBySessionName(struct FName SessionName, uint8_t&
PlatformSpecificInfo);
bool ReadPlatformSpecificSessionInfo(struct FOnlineGameSearchResult& DesiredGame,
uint8_t& PlatformSpecificInfo);
void ClearQosStatusChangedDelegate(struct FScriptDelegate QosStatusChangedDelegate);
void AddQosStatusChangedDelegate(struct FScriptDelegate QosStatusChangedDelegate);
void OnQosStatusChanged(int32_t NumComplete, int32_t NumTotal);
void ClearCancelFindOnlineGamesCompleteDelegate(struct FScriptDelegate
CancelFindOnlineGamesCompleteDelegate);
void AddCancelFindOnlineGamesCompleteDelegate(struct FScriptDelegate
CancelFindOnlineGamesCompleteDelegate);
void OnCancelFindOnlineGamesComplete(unsigned long bWasSuccessful);
bool CancelFindOnlineGames();
void ClearFindOnlineGamesCompleteDelegate(struct FScriptDelegate
FindOnlineGamesCompleteDelegate);
void AddFindOnlineGamesCompleteDelegate(struct FScriptDelegate
FindOnlineGamesCompleteDelegate);
void OnFindOnlineGamesComplete(unsigned long bWasSuccessful);
bool FindOnlineGames(uint8_t SearchingPlayerNum, class UOnlineGameSearch*
SearchSettings);
void ClearDestroyOnlineGameCompleteDelegate(struct FScriptDelegate
DestroyOnlineGameCompleteDelegate);
void AddDestroyOnlineGameCompleteDelegate(struct FScriptDelegate
DestroyOnlineGameCompleteDelegate);
void OnDestroyOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool DestroyOnlineGame(struct FName SessionName);
class UOnlineGameSettings* GetGameSettings(struct FName SessionName);
void ClearUpdateOnlineGameCompleteDelegate(struct FScriptDelegate
UpdateOnlineGameCompleteDelegate);
void AddUpdateOnlineGameCompleteDelegate(struct FScriptDelegate
UpdateOnlineGameCompleteDelegate);
void OnUpdateOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool UpdateOnlineGame(struct FName SessionName, class UOnlineGameSettings*
UpdatedGameSettings, unsigned long bShouldRefreshOnlineData);
void ClearCreateOnlineGameCompleteDelegate(struct FScriptDelegate
CreateOnlineGameCompleteDelegate);
void AddCreateOnlineGameCompleteDelegate(struct FScriptDelegate
CreateOnlineGameCompleteDelegate);
void OnCreateOnlineGameComplete(struct FName SessionName, unsigned long
bWasSuccessful);
bool CreateOnlineGame(uint8_t HostingPlayerNum, struct FName SessionName, class
UOnlineGameSettings* NewGameSettings);
void SetFriendJoinLocation(struct FUniqueNetId JoinablePlayerID, class FString ServerAddress,
uint8_t Visibility);
void ClearReportMatchmakingInfoDelegate(struct FScriptDelegate OldDelegate);
void AddReportMatchmakingInfoDelegate(struct FScriptDelegate NewDelegate);
void EventReportMatchmakingInfo(class FString NewInfo);
};

// Class Engine.OnlinePlayerInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlinePlayerInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlinePlayerInterface");
}

return uClassPointer;
};

bool ReadBlockList(uint8_t LocalUserNum);
void GetBlockList(uint8_t LocalUserNum, TArray<struct FOnlineFriend>& OutBlockList);
void AddPlayerUnblockedDelegate(uint8_t LocalUserNum, struct FScriptDelegate Delegate);
void OnPlayerUnblocked(uint8_t LocalUserNum, struct FUniqueNetId PlayerID, class UError*
Error);
```

```
void AddPlayerBlockedDelegate(uint8_t LocalUserNum, struct FScriptDelegate Delegate);
void OnPlayerBlocked(uint8_t LocalUserNum, struct FUniqueNetId PlayerID, class UError* Error);
void AddBlockListUpdatedDelegate(uint8_t LocalUserNum, struct FScriptDelegate Delegate);
void OnBlockListUpdated(uint8_t LocalUserNum);
void EpicIDToPlatformID(struct FUniqueNetId EpicAccountId, uint8_t TargetPlatform, struct
FScriptDelegate Callback);
void EpicIDToPlatformIDCallback(struct FUniqueNetId PlatformAccountId, class FString Error);
struct FUniqueNetId GetEpicAccountId(struct FUniqueNetId PlatformId);
TArray<struct FName> GetActiveDiscDLC();
void SetOnlineSubscriptionRequirement(unsigned long bRequiresOnlineSubscription);
int32_t GetControllerID(int32_t LocalPlayerNum);
TArray<struct FName> GetConnectedControllerNames();
void OnLocalPlayerRemoved(int32_t LocalPlayerNum);
void UnregisterController(int32_t LocalPlayerNum);
void RegisterController(int32_t LocalPlayerNum, int32_t ControllerId);
bool CanRegisterController(int32_t LocalPlayerNum);
bool ShowBindings(int32_t ControllerId);
void SetInputAPI(uint8_t TargetAPI);
void ClearInputAPIChangedDelegate(struct FScriptDelegate InputAPIChangedDelegate);
void AddInputAPIChangedDelegate(struct FScriptDelegate InputAPIChangedDelegate);
void OnInputAPIChanged(uint8_t TargetAPI);
void ClearUnregisteredControllerDelegate(struct FScriptDelegate
UnregisteredControllerDelegate);
void ClearRegisteredControllerDelegate(struct FScriptDelegate RegisteredControllerDelegate);
void AddUnregisteredControllerDelegate(struct FScriptDelegate UnregisteredControllerDelegate);
void AddRegisteredControllerDelegate(struct FScriptDelegate RegisteredControllerDelegate);
void OnUnregisteredController(int32_t LocalPlayerNum);
void OnRegisteredController(int32_t LocalPlayerNum, int32_t ControllerId);
bool HasIncomingFriendInvite(uint8_t LocalUserNum, struct FUniqueNetId InviteFrom);
bool HasFriendsFunctionality();
bool CheckParentalControlInfo(unsigned long bShowUi);
bool GetPlayHistoryRegistrationKey(TArray<uint8_t>& Key);
void RemoveCanPlayOnlineChangedDelegate(struct FScriptDelegate Callback);
void AddCanPlayOnlineChangedDelegate(struct FScriptDelegate Callback);
class FString GetPlayerLanguage(uint8_t LocalUserNum);
void GetPlayerCountry(uint8_t LocalUserNum);
void ClearReadPlayerCountryDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadPlayerCountryDelegate);
void AddReadPlayerCountryDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadPlayerCountryDelegate);
uint8_t GetAchievements(uint8_t LocalUserNum, int32_t TitleId, TArray<struct
FAchievementDetails>& Achievements);
void ClearReadAchievementsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadAchievementsCompleteDelegate);
void AddReadAchievementsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadAchievementsCompleteDelegate);
void OnReadAchievementsComplete(int32_t TitleId);
bool ReadAchievements(uint8_t LocalUserNum, int32_t TitleId, unsigned long bShouldReadText,
unsigned long bShouldReadImages);
void ClearUnlockAchievementCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
UnlockAchievementCompleteDelegate);
void AddUnlockAchievementCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
UnlockAchievementCompleteDelegate);
void OnUnlockAchievementComplete(unsigned long bWasSuccessful);
```

```
bool UnlockAchievement(uint8_t LocalUserNum, int32_t AchievementId, float PercentComplete);
bool DeleteMessage(uint8_t LocalUserNum, int32_t MessageIndex);
void ClearFriendMessageReceivedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
MessageDelegate);
void AddFriendMessageReceivedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
MessageDelegate);
void OnFriendMessageReceived(uint8_t LocalUserNum, struct FUniqueNetId SendingPlayer,
class FString SendingNick, class FString Message);
void GetFriendMessages(uint8_t LocalUserNum, TArray<struct FOnlineFriendMessage>&
FriendMessages);
void ClearJoinFriendGameCompleteDelegate(struct FScriptDelegate
JoinFriendGameCompleteDelegate);
void AddJoinFriendGameCompleteDelegate(struct FScriptDelegate
JoinFriendGameCompleteDelegate);
void OnJoinFriendGameComplete(unsigned long bWasSuccessful);
bool JoinFriendGame(uint8_t LocalUserNum, struct FUniqueNetId Friend);
void ClearReceivedGameInviteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReceivedGameInviteDelegate);
void AddReceivedGameInviteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReceivedGameInviteDelegate);
void OnReceivedGameInvite(uint8_t LocalUserNum, class FString InviterName);
bool SendGameInviteToFriends(uint8_t LocalUserNum, TArray<struct FUniqueNetId> Friends,
class FString Text);
bool SendGameInviteToFriend(uint8_t LocalUserNum, struct FUniqueNetId Friend, class FString
Text);
bool SendMessageToFriendW(uint8_t LocalUserNum, struct FUniqueNetId Friend, class FString
Message);
void ClearFriendInviteCanceledDelegate(uint8_t LocalUserNum, struct FScriptDelegate
InviteDelegate);
void AddFriendInviteCanceledDelegate(uint8_t LocalUserNum, struct FScriptDelegate
InviteDelegate);
void OnFriendInviteCanceled(uint8_t LocalUserNum, struct FUniqueNetId CanceledUserId);
void ClearFriendInviteReceivedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
InviteDelegate);
void AddFriendInviteReceivedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
InviteDelegate);
void OnFriendInviteReceived(uint8_t LocalUserNum, struct FUniqueNetId RequestingPlayer, class
FString RequestingNick, class FString Message);
bool RemoveFriend(uint8_t LocalUserNum, struct FUniqueNetId FormerFriend);
void ClearRemoveFriendCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
RemoveFriendDelegate);
void AddRemoveFriendCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
RemoveFriendDelegate);
void OnRemoveFriendComplete(unsigned long bWasSuccessful, struct FUniqueNetId
RemovedID);
bool DenyFriendInvite(uint8_t LocalUserNum, struct FUniqueNetId RequestingPlayer);
void ClearDenyFriendInviteCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
FriendDelegate);
void AddDenyFriendInviteCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
FriendDelegate);
void OnDenyFriendInviteComplete(struct FUniqueNetId FriendId, class UError* Error);
bool AcceptFriendInvite(uint8_t LocalUserNum, struct FUniqueNetId RequestingPlayer);
void ClearAcceptFriendInviteCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
FriendDelegate);
```

```
void AddAcceptFriendInviteCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate FriendDelegate);
void OnAcceptFriendInviteComplete(struct FUniqueNetId FriendId, class UError* Error);
void ClearAddFriendByNameCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate FriendDelegate);
void AddAddFriendByNameCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate FriendDelegate);
void OnAddFriendByNameComplete(unsigned long bWasSuccessful);
bool AddFriendByName(uint8_t LocalUserNum, class FString FriendName, class FString Message);
bool QueryUserByDisplayName(uint8_t LocalUserNum, class FString DisplayName);
void ClearQueryUserByDisplayNameCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate QueryDelegate);
void AddQueryUserByDisplayNameCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate QueryDelegate);
void OnQueryUserByDisplayName(unsigned long bWasSuccessful, class FString QueriedDisplayName, struct FUniqueNetId UserId);
bool AddFriend(uint8_t LocalUserNum, struct FUniqueNetId NewFriend, class FString Message);
void ClearAddFriendCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate FriendDelegate);
void AddAddFriendCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate FriendDelegate);
void OnAddFriendComplete(struct FUniqueNetId NewFriendId, class UError* Error);
class FString GetKeyboardInputResults(uint8_t& bWasCanceled);
void ClearKeyboardInputDoneDelegate(struct FScriptDelegate InputDelegate);
void AddKeyboardInputDoneDelegate(struct FScriptDelegate InputDelegate);
void OnKeyboardInputComplete(unsigned long bWasSuccessful);
bool HideKeyboardUI(uint8_t LocalUserNum);
bool ShowKeyboardUI(uint8_t LocalUserNum, class FString TitleText, class FString DescriptionText, unsigned long bIsPassword, unsigned long bShouldValidate, class FString DefaultText, int32_t MaxResultLength);
void SetOnlineStatus(uint8_t LocalUserNum, int32_t StatusId, TArray<struct FLocalizedStringSetting>& LocalizedStringSettings, TArray<struct FSettingsProperty>& Properties);
uint8_t GetFriendsList(uint8_t LocalUserNum, int32_t Count, int32_t StartingAt, TArray<struct FOnlineFriend>& Friends);
void ClearReadFriendsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate ReadFriendsCompleteDelegate);
void AddReadFriendsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate ReadFriendsCompleteDelegate);
void OnReadFriendsComplete(unsigned long bWasSuccessful);
bool ReadFriendsList(uint8_t LocalUserNum, int32_t Count, int32_t StartingAt);
void ClearWritePlayerStorageCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate WritePlayerStorageCompleteDelegate);
void AddWritePlayerStorageCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate WritePlayerStorageCompleteDelegate);
void OnWritePlayerStorageComplete(uint8_t LocalUserNum, unsigned long bWasSuccessful);
bool WritePlayerStorage(uint8_t LocalUserNum, class UOnlinePlayerStorage* PlayerStorage, int32_t DeviceID);
class UOnlinePlayerStorage* GetPlayerStorage(uint8_t LocalUserNum);
void ClearReadPlayerStorageForNetIdCompleteDelegate(struct FUniqueNetId NetId, struct FScriptDelegate ReadPlayerStorageForNetIdCompleteDelegate);
void AddReadPlayerStorageForNetIdCompleteDelegate(struct FUniqueNetId NetId, struct FScriptDelegate ReadPlayerStorageForNetIdCompleteDelegate);
void OnReadPlayerStorageForNetIdComplete(struct FUniqueNetId NetId, unsigned long
bWasSuccessful);
bool ReadPlayerStorageForNetId(uint8_t LocalUserNum, struct FUniqueNetId NetId, class
UOnlinePlayerStorage* PlayerStorage);
void ClearReadPlayerStorageCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadPlayerStorageCompleteDelegate);
void AddReadPlayerStorageCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadPlayerStorageCompleteDelegate);
void OnReadPlayerStorageComplete(uint8_t LocalUserNum, unsigned long bWasSuccessful);
bool ReadPlayerStorage(uint8_t LocalUserNum, class UOnlinePlayerStorage* PlayerStorage,
int32_t DeviceID);
void ClearWriteProfileSettingsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
WriteProfileSettingsCompleteDelegate);
void AddWriteProfileSettingsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
WriteProfileSettingsCompleteDelegate);
void OnWriteProfileSettingsComplete(uint8_t LocalUserNum, unsigned long bWasSuccessful);
bool WriteProfileSettings(uint8_t LocalUserNum, class UOnlineProfileSettings* ProfileSettings);
class UOnlineProfileSettings* GetProfileSettings(uint8_t LocalUserNum);
void ClearReadProfileSettingsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadProfileSettingsCompleteDelegate);
void AddReadProfileSettingsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadProfileSettingsCompleteDelegate);
void OnReadProfileSettingsComplete(uint8_t LocalUserNum, unsigned long bWasSuccessful);
bool ReadProfileSettings(uint8_t LocalUserNum, class UOnlineProfileSettings* ProfileSettings);
void ClearAvatarChangeDelegate(uint8_t LocalUserNum, struct FScriptDelegate AvatarDelegate);
void AddAvatarChangeDelegate(uint8_t LocalUserNum, struct FScriptDelegate AvatarDelegate);
void GetFriendPresence(struct FOnlineFriend& FriendData);
void ClearFriendPresenceChangeDelegate(uint8_t LocalUserNum, struct FScriptDelegate
PresenceDelegate);
void AddFriendPresenceChangeDelegate(uint8_t LocalUserNum, struct FScriptDelegate
PresenceDelegate);
void ClearFriendsChangeDelegate(uint8_t LocalUserNum, struct FScriptDelegate
FriendsDelegate);
void AddFriendsChangeDelegate(uint8_t LocalUserNum, struct FScriptDelegate
FriendsDelegate);
void ClearMutingChangeDelegate(struct FScriptDelegate MutingDelegate);
void AddMutingChangeDelegate(struct FScriptDelegate MutingDelegate);
void ClearLoginCancelledDelegate(struct FScriptDelegate CancelledDelegate);
void AddLoginCancelledDelegate(struct FScriptDelegate CancelledDelegate);
void ClearLoginStatusChangeDelegate(struct FScriptDelegate LoginStatusDelegate, uint8_t
LocalUserNum);
void AddLoginStatusChangeDelegate(struct FScriptDelegate LoginStatusDelegate, uint8_t
LocalUserNum);
void OnLoginStatusChange(uint8_t NewStatus, struct FUniqueNetId NewId);
void ClearLoginChangeDelegate(struct FScriptDelegate LoginDelegate);
void AddLoginChangeDelegate(struct FScriptDelegate LoginDelegate);
void ClearUserSignInCompleteDelegate(struct FScriptDelegate UserSignInCompleteDelegate);
void AddUserSignInCompleteDelegate(struct FScriptDelegate UserSignInCompleteDelegate);
void ClearUserSwitchCompleteDelegate(struct FScriptDelegate UserSwitchCompleteDelegate);
void AddUserSwitchCompleteDelegate(struct FScriptDelegate UserSwitchCompleteDelegate);
void SetPrimaryPlayerGamepadToLastInput();
bool ShowFriendsUI(uint8_t LocalUserNum);
bool IsMuted(uint8_t LocalUserNum, struct FUniqueNetId PlayerID);
bool AreAnyFriends(uint8_t LocalUserNum, TArray<struct FFriendsQuery>& Query);
bool IsFriend(uint8_t LocalUserNum, struct FUniqueNetId PlayerID);
void RequestRestrictedFeatureMessaging(uint8_t LocalUserNum, uint8_t RestrictedFeature);
uint8_t CanCommunicate(uint8_t LocalUserNum, uint8_t CommMethod, unsigned long
bAttemptToResolve);
bool CanDownloadUserContent(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanUploadFitnessData(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanShareKinectContent(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanShareWithSocialNetwork(uint8_t LocalUserNum, unsigned long bAttemptToResolve,
class FString Reason, uint8_t& PrivilegeLevelHint);
bool CanBrowseInternet(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class FString
Reason, uint8_t& PrivilegeLevelHint);
bool CanAccessPremiumVideoContent(uint8_t LocalUserNum, unsigned long
bAttemptToResolve, class FString Reason, uint8_t& PrivilegeLevelHint);
bool CanAccessPremiumContent(uint8_t LocalUserNum, unsigned long bAttemptToResolve,
class FString Reason, uint8_t& PrivilegeLevelHint);
bool CanUseCloudStorage(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanRecordDVRClips(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanShowPresenceInformation(uint8_t LocalUserNum, unsigned long bAttemptToResolve,
class FString Reason, uint8_t& PrivilegeLevelHint);
bool CanViewPlayerProfiles(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanPurchaseContent(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanShareUserCreatedContent(uint8_t LocalUserNum, unsigned long bAttemptToResolve,
class FString Reason, uint8_t& PrivilegeLevelHint);
bool CanCommunicateVoice(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanCommunicateVideo(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanCommunicateText(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class
FString Reason, uint8_t& PrivilegeLevelHint);
bool CanPlayOnline(uint8_t LocalUserNum, unsigned long bAttemptToResolve, class FString
Reason, uint8_t& PrivilegeLevelHint);
bool IsGuestLogin(uint8_t LocalUserNum);
class FString GetPlayerNickname(uint8_t LocalUserNum);
bool GetUniquePlayerId(uint8_t LocalUserNum, struct FUniqueNetId& PlayerID);
uint8_t GetLoginStatus(uint8_t LocalUserNum);
void ClearLogoutCompletedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
LogoutDelegate);
void AddLogoutCompletedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
LogoutDelegate);
void OnLogoutCompleted(unsigned long bWasSuccessful);
bool Logout(uint8_t LocalUserNum);
void ClearLoginFailedDelegate(uint8_t LocalUserNum, struct FScriptDelegate LoginDelegate);
void AddLoginFailedDelegate(uint8_t LocalUserNum, struct FScriptDelegate LoginDelegate);
void OnLoginFailed(uint8_t LocalUserNum, uint8_t ErrorCode);
bool AutoLogin();
bool Login(uint8_t LocalUserNum, class FString LoginName, class FString Password, unsigned
long bWantsLocalOnly);
```

```
void SetKickPlayerDialogActive(unsigned long Active);
bool IsUserSwitchActive();
void SetKickPreviousUser(uint8_t LocalUserNum);
bool SupportInGameLogin();
bool ShowControllerUI();
bool ShowLoginUIForOrphanedUser(uint8_t LocalUserNum);
bool ShowLoginUI(uint8_t LocalUserNum, unsigned long bShowOnlineOnly);
void CanPlayOnlineChanged(uint8_t LocalUserNum);
void OnPlayerCountryRetrieved(struct FUniqueNetId PlayerID, class FString Country);
void FriendPresenceChange(struct FUniqueNetId PlayerID);
void OnAvatarChange(struct FUniqueNetId PlayerID);
void OnFriendsChange();
void OnMutingChange();
void OnLoginCancelled();
void OnLoginChange(uint8_t LocalUserNum);
void OnUserSignInComplete(uint8_t LocalUserNum);
void OnUserSwitchComplete(uint8_t LocalUserNum);
};

// Class Engine.OnlinePlayerInterfaceEx
// 0x0000 (0x0060 - 0x0060)
class UOnlinePlayerInterfaceEx : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlinePlayerInterfaceEx");
}

return uClassPointer;
};

bool WordFilterSanitizeString(class FString Comment, struct FScriptDelegate SanitizeDelegate,
struct FUniqueNetId PlayerID);
void OnSanitizeStringComplete(struct FWordFilterResult Result);
bool RecordPlayersRecentlyMetKeys(uint8_t LocalUserNum, TArray<struct FFriendHistoryKey>&
PlayerKeys);
bool RecordPlayersRecentlyMet(uint8_t LocalUserNum, class FString GameDescription,
TArray<struct FUniqueNetId>& Players);
void ClearSaveDataNoSpaceDialogCompleteDelegate(struct FScriptDelegate DeviceDelegate);
void AddSaveDataNoSpaceDialogCompleteDelegate(struct FScriptDelegate DeviceDelegate);
void OnSaveDataNoSpaceDialogComplete(unsigned long bContinueWithoutSave);
TArray<bool> GetSyncedAchievements(uint8_t LocalUserNum);
bool UnlockAchievement(uint8_t LocalUserNum, int32_t AchievementId, float PercentComplete);
bool UpdateStat(uint8_t LocalUserNum, struct FName StatName, int32_t Points);
bool AddInGamePost(class FString InPostID, uint8_t LocalUserNum, TArray<class FString>
StringReplaceList);
void SetRichPresence(uint8_t LocalUserNum, class FString PresenceString, class FString
GameDataString);
bool ShowCustomMessageUI(uint8_t LocalUserNum, class FString MessageTitle, class FString
NonEditableMessage, class FString EditableMessage, TArray<struct FUniqueNetId>& Recipients);
void ClearCrossTitleProfileSettings(uint8_t LocalUserNum, int32_t TitleId);
class UOnlineProfileSettings* GetCrossTitleProfileSettings(uint8_t LocalUserNum, int32_t
TitleId);
void ClearReadCrossTitleProfileSettingsCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate ReadProfileSettingsCompleteDelegate);
void AddReadCrossTitleProfileSettingsCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate ReadProfileSettingsCompleteDelegate);
void OnReadCrossTitleProfileSettingsComplete(uint8_t LocalUserNum, int32_t TitleId, unsigned
long bWasSuccessful);
bool ReadCrossTitleProfileSettings(uint8_t LocalUserNum, int32_t TitleId, class
UOnlineProfileSettings* ProfileSettings);
bool ShowCustomPlayersUI(uint8_t LocalUserNum, class FString Title, class FString Description,
TArray<struct FUniqueNetId>& Players);
bool ShowPlayersUI(uint8_t LocalUserNum);
bool ShowFriendsInviteUI(uint8_t LocalUserNum, struct FUniqueNetId PlayerID);
void ClearProfileDataChangedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ProfileDataChangedDelegate);
void AddProfileDataChangedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ProfileDataChangedDelegate);
void OnProfileDataChanged();
bool UnlockGamerPicture(uint8_t LocalUserNum, int32_t PictureId);
bool IsDeviceValid(int32_t DeviceID, int32_t SizeNeeded);
int32_t GetDeviceSelectionResults(uint8_t LocalUserNum, class FString& DeviceName);
void ClearDeviceSelectionDoneDelegate(uint8_t LocalUserNum, struct FScriptDelegate
DeviceDelegate);
void AddDeviceSelectionDoneDelegate(uint8_t LocalUserNum, struct FScriptDelegate
DeviceDelegate);
void OnDeviceSelectionComplete(unsigned long bWasSuccessful);
bool ShowDeviceSelectionUI(uint8_t LocalUserNum, int32_t SizeNeeded, unsigned long
bManageStorage);
bool ShowContentMarketplaceUI(uint8_t LocalUserNum, int32_t CategoryMask, int32_t OfferId);
bool ShowInviteUI(uint8_t LocalUserNum, class FString InviteText);
bool ShowAchievementsUI(uint8_t LocalUserNum);
bool ShowMessagesUI(uint8_t LocalUserNum);
bool ShowGamerCardUI(uint8_t LocalUserNum, struct FUniqueNetId PlayerID, class FString
PlayerName);
bool ShowFeedbackUI(uint8_t LocalUserNum, struct FUniqueNetId PlayerID);
};

// Class Engine.CoverReplicator
// 0x0010 (0x0268 - 0x0278)
class ACoverReplicator : public AReplicationInfo
{
public:
TArray<struct FCoverReplicationInfo>            CoverReplicationData;                     // 0x0268
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
```

```
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.CoverReplicator");
}

return uClassPointer;
};

void ClientReceiveLinkDisabledState(int32_t Index, class ACoverLink* Link, unsigned long
bLinkDisabled);
void ServerSendLinkDisabledState(int32_t Index);
void NotifyLinkDisabledStateChange(class ACoverLink* Link);
void ClientReceiveManualCoverTypeSlots(int32_t Index, class ACoverLink* Link, uint8_t
NumCoverTypesChanged, struct FManualCoverTypeInfo SlotsCoverTypeChanged, unsigned long
bDone);
void ServerSendManualCoverTypeSlots(int32_t Index);
void NotifySetManualCoverTypeForSlots(class ACoverLink* Link, uint8_t NewCoverType,
TArray<int32_t>& SlotIndices);
void ClientReceiveAdjustedSlots(int32_t Index, class ACoverLink* Link, uint8_t
NumSlotsAdjusted, uint8_t SlotsAdjusted, unsigned long bDone);
void ServerSendAdjustedSlots(int32_t Index);
void NotifyAutoAdjustSlots(class ACoverLink* Link, TArray<int32_t>& SlotIndices);
void ClientReceiveDisabledSlots(int32_t Index, class ACoverLink* Link, uint8_t
NumSlotsDisabled, uint8_t SlotsDisabled, unsigned long bDone);
void ServerSendDisabledSlots(int32_t Index);
void NotifyDisabledSlots(class ACoverLink* Link, TArray<int32_t>& SlotIndices);
void ClientReceiveEnabledSlots(int32_t Index, class ACoverLink* Link, uint8_t NumSlotsEnabled,
uint8_t SlotsEnabled, unsigned long bDone);
void ServerSendEnabledSlots(int32_t Index);
void NotifyEnabledSlots(class ACoverLink* Link, TArray<int32_t>& SlotIndices);
void ClientReceiveInitialCoverReplicationInfo(int32_t Index, class ACoverLink* Link, unsigned
long bLinkDisabled, uint8_t NumSlotsEnabled, uint8_t SlotsEnabled, uint8_t NumSlotsDisabled,
uint8_t SlotsDisabled, uint8_t NumSlotsAdjusted, uint8_t SlotsAdjusted, uint8_t
NumCoverTypesChanged, struct FManualCoverTypeInfo SlotsCoverTypeChanged, unsigned long
bDone);
void ServerSendInitialCoverReplicationInfo(int32_t Index);
void ClientSetOwner(class APlayerController* PC);
void ReplicateInitialCoverInfo();
void PurgeOldEntries();
};

// Class Engine.OnlineCommunityContentInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineCommunityContentInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.OnlineCommunityContentInterface");
}

return uClassPointer;
};

void DownloadAllWorkshopData(struct FScriptDelegate Callback);
void OnDownloadedWorkshopData(unsigned long bSuccess, TArray<struct FDownloadedWorkshopData>& Items);
void RateContent(uint8_t PlayerNum, int32_t NewRating, struct FCommunityContentFile& FileToRate);
void ClearGetContentPayloadCompleteDelegate(struct FScriptDelegate GetContentPayloadCompleteDelegate);
void AddGetContentPayloadCompleteDelegate(struct FScriptDelegate GetContentPayloadCompleteDelegate);
void OnGetContentPayloadComplete(unsigned long bWasSuccessful, struct FCommunityContentFile FileDownloaded, TArray<uint8_t>& Payload);
bool GetContentPayload(uint8_t PlayerNum, struct FCommunityContentFile& FileDownloaded);
void ClearDownloadContentCompleteDelegate(struct FScriptDelegate DownloadContentCompleteDelegate);
void AddDownloadContentCompleteDelegate(struct FScriptDelegate DownloadContentCompleteDelegate);
void OnDownloadContentComplete(unsigned long bWasSuccessful, struct FCommunityContentFile FileDownloaded, TArray<uint8_t> Payload);
bool DownloadContent(uint8_t PlayerNum, struct FCommunityContentFile& FileToDownload);
void ClearUploadContentCompleteDelegate(struct FScriptDelegate UploadContentCompleteDelegate);
void AddUploadContentCompleteDelegate(struct FScriptDelegate UploadContentCompleteDelegate);
void OnUploadContentComplete(unsigned long bWasSuccessful, struct FCommunityContentFile UploadedFile);
bool UploadContent(uint8_t PlayerNum, TArray<uint8_t>& Payload, struct FCommunityContentMetadata& MetaData);
bool GetFriendsContentList(uint8_t PlayerNum, struct FOnlineFriend& Friend, TArray<struct FCommunityContentFile>& ContentFiles);
void ClearReadFriendsContentListCompleteDelegate(struct FScriptDelegate ReadFriendsContentListCompleteDelegate);
void AddReadFriendsContentListCompleteDelegate(struct FScriptDelegate ReadFriendsContentListCompleteDelegate);
void OnReadFriendsContentListComplete(unsigned long bWasSuccessful);
bool ReadFriendsContentList(uint8_t PlayerNum, int32_t StartAt, int32_t NumToRead, TArray<struct FOnlineFriend>& Friends);
bool GetContentList(uint8_t PlayerNum, TArray<struct FCommunityContentFile>& ContentFiles);
void ClearReadContentListCompleteDelegate(struct FScriptDelegate ReadContentListCompleteDelegate);
void AddReadContentListCompleteDelegate(struct FScriptDelegate ReadContentListCompleteDelegate);
void OnReadContentListComplete(unsigned long bWasSuccessful, TArray<struct FCommunityContentFile> ContentFiles);
bool ReadContentList(uint8_t PlayerNum, struct FUniqueNetId NetId, class FString Path, int32_t StartAt, int32_t NumToRead);
void Exit();
bool Init();
```

```
};

// Class Engine.OnlineGameDVRInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineGameDVRInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineGameDVRInterface");
}

return uClassPointer;
};

bool CancelRecordingEvent(uint8_t LocalUserNum, class FString EventName);
bool RecordPreviousTimespan(uint8_t LocalUserNum, class FString EventName, float Duration);
bool EndRecordingEvent(uint8_t LocalUserNum, class FString EventName);
void ClearRecordEventCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
RecordEventCompleteDelegate);
void AddRecordEventCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
RecordEventCompleteDelegate);
void OnRecordEventComplete(unsigned long bWasSuccessful, uint8_t LocalUserNum, class
FString EventName);
bool BeginRecordingEvent(uint8_t LocalUserNum, class FString EventName);
void ClearReadRecordedClipsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadRecordedClipsCompleteDelegate);
void AddReadRecordedClipsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadRecordedClipsCompleteDelegate);
void ClearCachedRecordedClips(uint8_t LocalUserNum);
void OnReadRecordedClipsComplete(unsigned long bWasSuccessful, uint8_t LocalUserNum);
bool ReadRecordedClips(uint8_t LocalUserNum);
void DisableRecording();
void EnableRecording();
};

// Class Engine.SharedCloudFileInterface
// 0x0000 (0x0060 - 0x0060)
class USharedCloudFileInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
```

```
{
uClassPointer = UObject::FindClass("Class Engine.SharedCloudFileInterface");
}

return uClassPointer;
};

void ClearWriteSharedFileCompleteDelegate(struct FScriptDelegate
WriteSharedFileCompleteDelegate);
void AddWriteSharedFileCompleteDelegate(struct FScriptDelegate
WriteSharedFileCompleteDelegate);
bool WriteSharedFile(class FString UserId, class FString Filename, TArray<uint8_t>& Contents);
void OnWriteSharedFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename, class FString SharedHandle);
void ClearReadSharedFileCompleteDelegate(struct FScriptDelegate
ReadSharedFileCompleteDelegate);
void AddReadSharedFileCompleteDelegate(struct FScriptDelegate
ReadSharedFileCompleteDelegate);
bool ReadSharedFile(class FString SharedHandle);
void OnReadSharedFileComplete(unsigned long bWasSuccessful, class FString SharedHandle);
bool ClearSharedFile(class FString SharedHandle);
bool ClearSharedFiles();
bool GetSharedFileContents(class FString SharedHandle, TArray<uint8_t>& FileContents);
};

// Class Engine.OnlineSocialInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineSocialInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineSocialInterface");
}

return uClassPointer;
};

void ClearPostLinkCompleted(uint8_t LocalUserNum, struct FScriptDelegate PostLinkDelegate);
void AddPostLinkCompleted(uint8_t LocalUserNum, struct FScriptDelegate PostLinkDelegate);
void OnPostLinkCompleted(uint8_t LocalUserNum, unsigned long bWasSuccessful);
bool PostLink(uint8_t LocalUserNum, struct FSocialPostLinkInfo& PostLinkInfo);
void ClearPostImageCompleted(uint8_t LocalUserNum, struct FScriptDelegate
PostImageDelegate);
void AddPostImageCompleted(uint8_t LocalUserNum, struct FScriptDelegate
PostImageDelegate);
void OnPostImageCompleted(uint8_t LocalUserNum, unsigned long bWasSuccessful);
bool PostImage(uint8_t LocalUserNum, TArray<uint8_t> FullImage, struct FSocialPostImageInfo&
PostImageInfo);
void ClearQuerySocialPostPrivilegesCompleted(struct FScriptDelegate PostPrivilegesDelegate);
void AddQuerySocialPostPrivilegesCompleted(struct FScriptDelegate PostPrivilegesDelegate);
void OnQuerySocialPostPrivilegesCompleted(unsigned long bWasSuccessful, struct
FSocialPostPrivileges PostPrivileges);
bool QuerySocialPostPrivileges();
};

// Class Engine.OnlinePartyChatInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlinePartyChatInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlinePartyChatInterface");
}

return uClassPointer;
};

bool IsInPartyChat(uint8_t LocalUserNum);
bool ShowCommunitySessionsUI(uint8_t LocalUserNum);
bool ShowVoiceChannelUI(uint8_t LocalUserNum);
bool ShowPartyUI(uint8_t LocalUserNum);
int32_t GetPartyBandwidth();
bool SetPartyMemberCustomData(uint8_t LocalUserNum, int32_t Data1, int32_t Data2, int32_t
Data3, int32_t Data4);
void ClearPartyMembersInfoChangedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
PartyMembersInfoChangedDelegate);
void AddPartyMembersInfoChangedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
PartyMembersInfoChangedDelegate);
void OnPartyMembersInfoChanged(class FString PlayerName, struct FUniqueNetId PlayerID,
int32_t CustomData1, int32_t CustomData2, int32_t CustomData3, int32_t CustomData4);
void ClearPartyMemberListChangedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
PartyMemberListChangedDelegate);
void AddPartyMemberListChangedDelegate(uint8_t LocalUserNum, struct FScriptDelegate
PartyMemberListChangedDelegate);
void OnPartyMemberListChanged(unsigned long bJoinedOrLeft, class FString PlayerName,
struct FUniqueNetId PlayerID);
bool GetPartyMemberInformation(struct FUniqueNetId MemberId, struct FOnlinePartyMember&
PartyMember);
bool GetPartyMembersInformation(TArray<struct FOnlinePartyMember>& PartyMembers);
void ClearSendPartyGameInvitesCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate SendPartyGameInvitesCompleteDelegate);
void AddSendPartyGameInvitesCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
SendPartyGameInvitesCompleteDelegate);
void OnSendPartyGameInvitesComplete(unsigned long bWasSuccessful);
```

```
bool SendPartyGameInvites(uint8_t LocalUserNum);
};

// Class Engine.OnlineNewsInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineNewsInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineNewsInterface");
}

return uClassPointer;
};

class FString GetNews(uint8_t LocalUserNum, uint8_t NewsType);
void ClearReadNewsCompletedDelegate(struct FScriptDelegate ReadNewsDelegate);
void AddReadNewsCompletedDelegate(struct FScriptDelegate ReadNewsDelegate);
void OnReadNewsCompleted(unsigned long bWasSuccessful, uint8_t NewsType);
bool ReadNews(uint8_t LocalUserNum, uint8_t NewsType);
};

// Class Engine.OnlineStatsInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineStatsInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineStatsInterface");
}

return uClassPointer;
};

bool RegisterStatGuid(struct FUniqueNetId PlayerID, class FString& ClientStatGuid);
class FString GetClientStatGuid();
void ClearRegisterHostStatGuidCompleteDelegateDelegate(struct FScriptDelegate
RegisterHostStatGuidCompleteDelegate);
void AddRegisterHostStatGuidCompleteDelegate(struct FScriptDelegate
RegisterHostStatGuidCompleteDelegate);
```

```
void OnRegisterHostStatGuidComplete(unsigned long bWasSuccessful);
bool RegisterHostStatGuid(class FString& HostStatGuid);
class FString GetHostStatGuid();
bool WriteOnlinePlayerScores(struct FName SessionName, int32_t LeaderboardId, TArray<struct
FOnlinePlayerScore>& PlayerScores);
void ClearFlushOnlineStatsCompleteDelegate(struct FScriptDelegate
FlushOnlineStatsCompleteDelegate);
void AddFlushOnlineStatsCompleteDelegate(struct FScriptDelegate
FlushOnlineStatsCompleteDelegate);
void OnFlushOnlineStatsComplete(struct FName SessionName, unsigned long bWasSuccessful);
bool FlushOnlineStats(struct FName SessionName);
bool WriteOnlineStats(struct FName SessionName, struct FUniqueNetId Player, class
UOnlineStatsWrite* StatsWrite);
void FreeStats(class UOnlineStatsRead* StatsRead);
void ClearReadOnlineStatsCompleteDelegate(struct FScriptDelegate
ReadOnlineStatsCompleteDelegate);
void AddReadOnlineStatsCompleteDelegate(struct FScriptDelegate
ReadOnlineStatsCompleteDelegate);
void OnReadOnlineStatsComplete(unsigned long bWasSuccessful);
bool ReadOnlineStatsByRankAroundPlayer(uint8_t LocalUserNum, class UOnlineStatsRead*
StatsRead, int32_t NumRows);
bool ReadOnlineStatsByRank(uint8_t LocalUserNum, class UOnlineStatsRead* StatsRead, int32_t
StartIndex, int32_t NumToRead);
bool ReadOnlineStatsForFriends(uint8_t LocalUserNum, class UOnlineStatsRead* StatsRead);
bool ReadOnlineStats(uint8_t LocalUserNum, class UOnlineStatsRead* StatsRead, TArray<struct
FUniqueNetId>& Players);
};

// Class Engine.OnlineGameClipsInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineGameClipsInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineGameClipsInterface");
}

return uClassPointer;
};

void NotifyEventClipErrorOccurred(struct FScriptDelegate InCallback);
void EventClipErrorOccurred(class FString InEpicAccountId, int32_t ClipId, class UErrorType*
InErrorType);
void NotifyEventGeneralErrorOccurred(struct FScriptDelegate InCallback);
void EventGeneralErrorOccurred(class UErrorType* InErrorType);
void NotifyEventMaskStatusChanged(struct FScriptDelegate InCallback);
void EventMaskStatusChanged(uint64_t InMaskAreaHandle, struct FGameClipsMaskArea
InMaskArea, uint8_t InNewMaskStatus);
void NotifyEventClipStatusChanged(struct FScriptDelegate InCallback);
void EventClipStatusChanged(class FString InEpicAccountId, int32_t InClipId, uint8_t
InNewClipStatus);
void NotifyEventConnectionStatusChanged(struct FScriptDelegate InCallback);
void EventConnectionStatusChanged(class FString InEpicAccountId, uint8_t InConnection,
uint8_t InNewConnectionStatus);
void NotifyEventRecordingChanged(struct FScriptDelegate InCallback);
void EventRecordingChanged(uint8_t InNewRecording);
void NotifyEventAvailabilityChanged(struct FScriptDelegate InCallback);
void EventAvailabilityChanged(uint8_t InNewAvailability);
bool IsAccountLinked(class FString InEpicAccountId);
bool IsUploading();
bool IsRecording();
bool IsAvailable();
float GetTimeUntilUnthrottled(class FString InEpicAccountId);
bool IsClipUploadingLimitReached(class FString InEpicAccountId);
void SetUserMaxClipUploadsPerMinute(int32_t InMaxClipUploadsPerMinute);
int32_t CreateClip(class FString InEpicAccountId, class FString InClipType);
void DisableMaskArea(uint64_t InMaskAreaHandle);
uint64_t EnableMaskArea(struct FGameClipsMaskArea& InMaskArea);
void StopRecording();
void StartRecording(uint64_t InClipDuration);
};

// Class Engine.OnlineVoiceInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineVoiceInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineVoiceInterface");
}

return uClassPointer;
};

bool SetVoiceReceiveVolume(float VoiceVolume);
bool UnmuteAll(uint8_t LocalUserNum);
bool MuteAll(uint8_t LocalUserNum, unsigned long bAllowFriends);
bool SetSpeechRecognitionObject(uint8_t LocalUserNum, class USpeechRecognition*
SpeechRecogObj);
bool SelectVocabulary(uint8_t LocalUserNum, int32_t VocabularyId);
void ClearRecognitionCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
RecognitionDelegate);
void AddRecognitionCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
RecognitionDelegate);
```

```
void OnRecognitionComplete();
bool GetRecognitionResults(uint8_t LocalUserNum, TArray<struct FSpeechRecognizedWord>&
Words);
bool StopSpeechRecognition(uint8_t LocalUserNum);
bool StartSpeechRecognition(uint8_t LocalUserNum);
void StopNetworkedVoice(uint8_t LocalUserNum);
void StartNetworkedVoice(uint8_t LocalUserNum);
void ClearPlayerTalkingDelegate(struct FScriptDelegate TalkerDelegate);
void AddPlayerTalkingDelegate(struct FScriptDelegate TalkerDelegate);
void OnPlayerTalkingStateChange(struct FUniqueNetId Player, unsigned long bIsTalking);
bool UnmuteRemoteTalker(uint8_t LocalUserNum, struct FUniqueNetId PlayerID, unsigned long
bIsSystemWide);
bool MuteRemoteTalker(uint8_t LocalUserNum, struct FUniqueNetId PlayerID, unsigned long
bIsSystemWide);
bool SetRemoteTalkerPriority(uint8_t LocalUserNum, struct FUniqueNetId PlayerID, int32_t
Priority);
bool IsHeadsetPresent(uint8_t LocalUserNum);
bool IsRemotePlayerTalking(struct FUniqueNetId PlayerID);
bool IsLocalPlayerTalking(uint8_t LocalUserNum);
bool UnregisterRemoteTalker(struct FUniqueNetId PlayerID);
bool RegisterRemoteTalker(struct FUniqueNetId PlayerID);
bool UnregisterLocalTalker(uint8_t LocalUserNum);
bool RegisterLocalTalker(uint8_t LocalUserNum);
};

// Class Engine.OnlineContentInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineContentInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineContentInterface");
}

return uClassPointer;
};

bool ClearSaveGames(uint8_t LocalUserNum);
bool DeleteSaveGame(uint8_t LocalUserNum, int32_t DeviceID, class FString FriendlyName, class
FString Filename);
void ClearWriteSaveGameDataComplete(uint8_t LocalUserNum, struct FScriptDelegate
WriteSaveGameDataCompleteDelegate);
void AddWriteSaveGameDataComplete(uint8_t LocalUserNum, struct FScriptDelegate
WriteSaveGameDataCompleteDelegate);
void OnWriteSaveGameDataComplete(unsigned long bWasSuccessful, uint8_t LocalUserNum,
int32_t DeviceID, class FString FriendlyName, class FString Filename, class FString
SaveFileName);
```

```
bool WriteSaveGameData(uint8_t LocalUserNum, int32_t DeviceID, class FString FriendlyName, class FString Filename, class FString SaveFileName, TArray<uint8_t>& SaveGameData);
void ClearReadSaveGameDataComplete(uint8_t LocalUserNum, struct FScriptDelegate ReadSaveGameDataCompleteDelegate);
void AddReadSaveGameDataComplete(uint8_t LocalUserNum, struct FScriptDelegate ReadSaveGameDataCompleteDelegate);
void OnReadSaveGameDataComplete(unsigned long bWasSuccessful, uint8_t LocalUserNum, int32_t DeviceID, class FString FriendlyName, class FString Filename, class FString SaveFileName);
bool GetSaveGameData(uint8_t LocalUserNum, int32_t DeviceID, class FString FriendlyName, class FString Filename, class FString SaveFileName, uint8_t& bIsValid, TArray<uint8_t>& SaveGameData);
bool ReadSaveGameData(uint8_t LocalUserNum, int32_t DeviceID, class FString FriendlyName, class FString Filename, class FString SaveFileName);
void GetAvailableDownloadCounts(uint8_t LocalUserNum, int32_t& NewDownloads, int32_t& TotalDownloads);
void ClearQueryAvailableDownloadsComplete(uint8_t LocalUserNum, struct FScriptDelegate QueryDownloadsDelegate);
void AddQueryAvailableDownloadsComplete(uint8_t LocalUserNum, struct FScriptDelegate QueryDownloadsDelegate);
void OnQueryAvailableDownloadsComplete(unsigned long bWasSuccessful);
bool QueryAvailableDownloads(uint8_t LocalUserNum, int32_t CategoryMask);
bool ClearCrossTitleSaveGames(uint8_t LocalUserNum);
void ClearReadCrossTitleSaveGameDataComplete(uint8_t LocalUserNum, struct FScriptDelegate ReadSaveGameDataCompleteDelegate);
void AddReadCrossTitleSaveGameDataComplete(uint8_t LocalUserNum, struct FScriptDelegate ReadSaveGameDataCompleteDelegate);
void OnReadCrossTitleSaveGameDataComplete(unsigned long bWasSuccessful, uint8_t LocalUserNum, int32_t DeviceID, int32_t TitleId, class FString FriendlyName, class FString Filename, class FString SaveFileName);
bool GetCrossTitleSaveGameData(uint8_t LocalUserNum, int32_t DeviceID, int32_t TitleId, class FString FriendlyName, class FString Filename, class FString SaveFileName, uint8_t& bIsValid, TArray<uint8_t>& SaveGameData);
bool ReadCrossTitleSaveGameData(uint8_t LocalUserNum, int32_t DeviceID, int32_t TitleId, class FString FriendlyName, class FString Filename, class FString SaveFileName);
void ClearReadCrossTitleContentCompleteDelegate(uint8_t LocalUserNum, uint8_t ContentType, struct FScriptDelegate ReadContentCompleteDelegate);
void AddReadCrossTitleContentCompleteDelegate(uint8_t LocalUserNum, uint8_t ContentType, struct FScriptDelegate ReadContentCompleteDelegate);
void OnReadCrossTitleContentComplete(unsigned long bWasSuccessful);
uint8_t GetCrossTitleContentList(uint8_t LocalUserNum, uint8_t ContentType, TArray<struct FOnlineCrossTitleContent>& ContentList);
void ClearCrossTitleContentList(uint8_t LocalUserNum, uint8_t ContentType);
bool ReadCrossTitleContentList(uint8_t LocalUserNum, uint8_t ContentType, int32_t TitleId, int32_t DeviceID);
uint8_t GetContentList(uint8_t LocalUserNum, uint8_t ContentType, TArray<struct FOnlineContent>& ContentList);
void ClearContentList(uint8_t LocalUserNum, uint8_t ContentType);
bool ReadContentList(uint8_t LocalUserNum, uint8_t ContentType, int32_t DeviceID);
void ClearReadContentComplete(uint8_t LocalUserNum, uint8_t ContentType, struct FScriptDelegate ReadContentCompleteDelegate);
void AddReadContentComplete(uint8_t LocalUserNum, uint8_t ContentType, struct FScriptDelegate ReadContentCompleteDelegate);
void OnReadContentComplete(unsigned long bWasSuccessful);
void ClearContentChangeDelegate(struct FScriptDelegate ContentDelegate, uint8_t
LocalUserNum);
void AddContentChangeDelegate(struct FScriptDelegate ContentDelegate, uint8_t
LocalUserNum);
void OnContentChange();
};

// Class Engine.OnlineLobbyInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineLobbyInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineLobbyInterface");
}

return uClassPointer;
};

void ClearHostStartPlayTogetherDelegate(struct FScriptDelegate InDelegate);
void AddHostStartPlayTogetherDelegate(struct FScriptDelegate InDelegate);
void OnHostStartPlayTogether(uint8_t LocalUserNum);
void ClearLobbyErrorDelegate(struct FScriptDelegate LobbyErrorDelegate);
void AddLobbyErrorDelegate(struct FScriptDelegate LobbyErrorDelegate);
void OnLobbyError(class FString Error);
void ClearLobbySessionCreatedDelegate(struct FScriptDelegate LobbySessionCreatedDelegate);
void AddLobbySessionCreatedDelegate(struct FScriptDelegate LobbySessionCreatedDelegate);
void OnLobbySessionCreated();
void RemoveLocalPlayerFromSession(struct FUniqueNetId& PartyMember);
void AddLocalPartyMemberToSession(struct FUniqueNetId& NewPartyMember);
bool KickPlayer(uint8_t Reason, struct FUniqueLobbyId& LobbyId, struct FUniqueNetId&
PlayerID);
void ClearLobbyDestroyedDelegate(struct FScriptDelegate LobbyDestroyedDelegate);
void AddLobbyDestroyedDelegate(struct FScriptDelegate LobbyDestroyedDelegate);
void OnLobbyDestroyed(uint8_t Reason, struct FUniqueLobbyId& LobbyId);
bool GetLobbyMembers(struct FUniqueLobbyId& LobbyId, TArray<struct FLobbyMember>&
Members);
bool ShowInviteUI(uint8_t LocalUserNum, struct FUniqueLobbyId& LobbyId);
void ClearLobbyInviteDelegate(struct FScriptDelegate LobbyInviteDelegate);
void AddLobbyInviteDelegate(struct FScriptDelegate LobbyInviteDelegate);
void OnLobbyInvite(unsigned long bAccepted, struct FUniqueLobbyId& LobbyId, struct
FUniqueNetId& FriendId);
bool InviteToLobby(struct FUniqueLobbyId& LobbyId, struct FUniqueNetId& PlayerID);
bool CanInviteToLobby(struct FUniqueLobbyId& LobbyId, struct FUniqueNetId& PlayerID);
bool SetLobbyOwner(struct FUniqueLobbyId& LobbyId, struct FUniqueNetId& NewOwner);
bool SetLobbyLock(unsigned long bLocked, struct FUniqueLobbyId& LobbyId);
bool SetLobbyServer(class FString ServerIP, struct FUniqueLobbyId& LobbyId, struct
FUniqueNetId& ServerUID);
bool RemoveLobbySetting(class FString Key, struct FUniqueLobbyId& LobbyId);
bool SetLobbySetting(class FString Key, class FString Value, struct FUniqueLobbyId& LobbyId);
bool GetLobbyAdmin(struct FUniqueLobbyId& LobbyId, struct FUniqueNetId& AdminId);
void ClearLobbyJoinGameDelegate(struct FScriptDelegate LobbyJoinGameDelegate);
void AddLobbyJoinGameDelegate(struct FScriptDelegate LobbyJoinGameDelegate);
void OnLobbyJoinGame(class FString ServerIP, struct FActiveLobbyInfo& LobbyInfo, struct
FUniqueNetId& ServerId);
void ClearLobbyReceiveBinaryDataDelegate(struct FScriptDelegate
LobbyReceiveBinaryDataDelegate);
void AddLobbyReceiveBinaryDataDelegate(struct FScriptDelegate
LobbyReceiveBinaryDataDelegate);
void OnLobbyReceiveBinaryData(int32_t MemberIndex, struct FActiveLobbyInfo& LobbyInfo,
TArray<uint8_t>& Data);
bool SendLobbyBinaryData(struct FUniqueLobbyId& LobbyId, TArray<uint8_t>& Data);
void ClearLobbyReceiveMessageDelegate(struct FScriptDelegate
LobbyReceiveMessageDelegate);
void AddLobbyReceiveMessageDelegate(struct FScriptDelegate
LobbyReceiveMessageDelegate);
void OnLobbyReceiveMessage(int32_t MemberIndex, class FString Type, class FString Message,
struct FActiveLobbyInfo& LobbyInfo);
bool SendLobbyMessage(class FString Message, struct FUniqueLobbyId& LobbyId);
void ClearLobbyMemberStatusUpdateDelegate(struct FScriptDelegate
LobbyMemberStatusUpdateDelegate);
void AddLobbyMemberStatusUpdateDelegate(struct FScriptDelegate
LobbyMemberStatusUpdateDelegate);
void OnLobbyMemberStatusUpdate(int32_t MemberIndex, int32_t InstigatorIndex, class FString
Status, struct FActiveLobbyInfo& LobbyInfo);
void ClearLobbyMemberSettingsUpdateDelegate(struct FScriptDelegate
LobbyMemberSettingsUpdateDelegate);
void AddLobbyMemberSettingsUpdateDelegate(struct FScriptDelegate
LobbyMemberSettingsUpdateDelegate);
void OnLobbyMemberSettingsUpdate(int32_t MemberIndex, struct FActiveLobbyInfo&
LobbyInfo);
void ClearLobbySettingsUpdateDelegate(struct FScriptDelegate LobbySettingsUpdateDelegate);
void AddLobbySettingsUpdateDelegate(struct FScriptDelegate LobbySettingsUpdateDelegate);
void OnLobbySettingsUpdate(struct FActiveLobbyInfo& LobbyInfo);
bool SetLobbyUserSetting(class FString Key, class FString Value, struct FUniqueLobbyId&
LobbyId);
bool LeaveLobby(struct FUniqueLobbyId& LobbyId);
void ClearJoinLobbyCompleteDelegate(struct FScriptDelegate JoinLobbyCompleteDelegate);
void AddJoinLobbyCompleteDelegate(struct FScriptDelegate JoinLobbyCompleteDelegate);
void OnJoinLobbyComplete(unsigned long bWasSuccessful, class FString Error, struct
FActiveLobbyInfo& LobbyInfo, struct FUniqueLobbyId& LobbyId);
bool JoinLobby(int32_t LocalPlayerNum, struct FUniqueLobbyId& LobbyId);
void ClearFindLobbiesCompleteDelegate(struct FScriptDelegate FindLobbiesCompleteDelegate);
void AddFindLobbiesCompleteDelegate(struct FScriptDelegate FindLobbiesCompleteDelegate);
void OnFindLobbiesComplete(unsigned long bWasSuccessful, TArray<struct FBasicLobbyInfo>&
LobbyList);
bool UpdateFoundLobbies(struct FUniqueLobbyId LobbyId);
bool FindLobbies(int32_t MaxResults, TArray<struct FLobbyFilter> Filters, TArray<struct
FLobbySortFilter> SortFilters, int32_t MinSlots, uint8_t Distance);
void ClearCreateLobbyCompleteDelegate(struct FScriptDelegate
CreateLobbyCompleteDelegate);
```

```
void AddCreateLobbyCompleteDelegate(struct FScriptDelegate CreateLobbyCompleteDelegate);
void OnCreateLobbyComplete(unsigned long bWasSuccessful, class FString Error, struct
FUniqueLobbyId& LobbyId);
bool CreateLobby(int32_t LocalPlayerNum, int32_t MaxPlayers, uint8_t Type, TArray<struct
FLobbyMetaData> InitialSettings);
};

// Class Engine.OnlinePurchaseInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlinePurchaseInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlinePurchaseInterface");
}

return uClassPointer;
};

void ClearMicroTxnResponseDelegate(struct FScriptDelegate ResponseMicroTxnDelegate);
void AddMicroTxnResponseDelegate(struct FScriptDelegate ResponseMicroTxnDelegate);
void OnMicroTxnResponse(unsigned long bAuthorized, uint64_t OrderId);
class FString FormatCurrency(class FString Currency, int32_t Price);
bool GetAppPriceInfo(struct FScriptDelegate Callback, struct FUniqueNetId& PlayerID,
TArray<struct FName>& AppNames);
void EventGetAppPriceInfoComplete(struct FName AppName, class FString Price, class FString
DiscountPrice, int32_t DiscountPercentage);
};

// Class Engine.OnlineSystemInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineSystemInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineSystemInterface");
}

return uClassPointer;
};
```

```
bool RefreshNetworkErrorToggle();
void UpdateSessionStatusFromPlayers(int32_t CurrentPlayerCount, int32_t numBotPlayers);
bool HandleBootMessage();
void ClearErrorDialogClosedDelegate(struct FScriptDelegate InDelegate);
void AddErrorDialogClosedDelegate(struct FScriptDelegate InDelegate);
void OnErrorDialogClosed(int32_t LocalUserNum);
void ClearCommerceDialogClosedDelegate(struct FScriptDelegate InDelegate);
void AddCommerceDialogClosedDelegate(struct FScriptDelegate InDelegate);
void OnCommerceDialogClosed();
void ClearCloseKickPlayerDialogDelegate(struct FScriptDelegate InDelegate);
void AddCloseKickPlayerDialogDelegate(struct FScriptDelegate InDelegate);
void OnCloseKickPlayerDialog();
void ClearUnlockedDLCChangeDelegate(struct FScriptDelegate InDelegate);
void AddUnlockedDLCChangeDelegate(struct FScriptDelegate InDelegate);
TArray<struct FName> GetUnlockedDLC();
void OnUnlockedDLCChange();
bool GetOverlayEnabled();
int32_t GetDLCPurchaseTime(struct FName AppName);
void ClearStorageDeviceChangeDelegate(struct FScriptDelegate StorageDeviceChangeDelegate);
void AddStorageDeviceChangeDelegate(struct FScriptDelegate StorageDeviceChangeDelegate);
void OnStorageDeviceChange();
uint8_t GetNATType();
void ClearConnectionStatusChangeDelegate(struct FScriptDelegate ConnectionStatusDelegate);
void AddConnectionStatusChangeDelegate(struct FScriptDelegate ConnectionStatusDelegate);
void OnConnectionStatusChange(uint8_t ConnectionStatus);
uint8_t GetCurrentConnectionStatus();
bool AnyPlayerChatRestricted();
void InitializeTrophyAPI();
void OpenStoreForItemsAsync(uint8_t LocalUserNum, TArray<class FString> Targets, struct FScriptDelegate Callback);
void OnStorePurchaseCompleteDelegate();
void OpenStoreForItems(uint8_t LocalUserNum, TArray<class FString> Targets);
void OpenStoreForDLC(uint8_t LocalUserNum, struct FName DLC);
void OpenErrorDialog(uint8_t LocalUserNum, uint8_t ErrorCode);
void OpenPS4DisplayMode(uint8_t LocalUserNum, uint8_t DisplayMode, TArray<class FString> Targets, int32_t ServiceLabel);
void ResetControllerColor(int32_t ControllerId);
void SetControllerColor(int32_t ControllerId, struct FColor NewColor);
bool IsControllerConnected(int32_t ControllerId);
void ClearUserRestoredDelegate(struct FScriptDelegate UserRestoredDelegate);
void AddUserRestoredDelegate(struct FScriptDelegate UserRestoredDelegate);
void OnUserRestored(uint8_t ControllerId);
void ClearUserOrphanedDelegate(struct FScriptDelegate UserOrphanedDelegate);
void AddUserOrphanedDelegate(struct FScriptDelegate UserOrphanedDelegate);
void OnUserOrphaned(uint8_t ControllerId);
void ClearControllerChangeDelegate(struct FScriptDelegate ControllerChangeDelegate);
void AddControllerChangeDelegate(struct FScriptDelegate ControllerChangeDelegate);
void OnControllerChange(int32_t ControllerId, unsigned long bIsConnected);
void SetNetworkNotificationPosition(uint8_t NewPos);
uint8_t GetNetworkNotificationPosition();
void ClearExternalUIChangeDelegate(struct FScriptDelegate ExternalUIDelegate);
void AddExternalUIChangeDelegate(struct FScriptDelegate ExternalUIDelegate);
void OnExternalUIChange(unsigned long bIsOpening);
void ClearLinkStatusChangeDelegate(struct FScriptDelegate LinkStatusDelegate);
void AddLinkStatusChangeDelegate(struct FScriptDelegate LinkStatusDelegate);
void OnLinkStatusChange(unsigned long bIsConnected);
bool HasLinkConnection();
};

// Class Engine.OnlineMarketplaceInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineMarketplaceInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineMarketplaceInterface");
}

return uClassPointer;
};

void ClearConsumeInventoryItemCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate ConsumeInventoryItemCompleteDelegate);
void AddConsumeInventoryItemCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate ConsumeInventoryItemCompleteDelegate);
void OnConsumeInventoryItemComplete(class FString ProductID, unsigned long bDidSucceed,
int32_t NewQuantity);
bool ConsumeInventoryItem(uint8_t LocalUserNum, class FString ProductID, int32_t Quantity,
class FString TransactionId);
void ResetInventoryItems(uint8_t LocalUserNum, uint8_t MediaType);
uint8_t GetInventoryItems(uint8_t LocalUserNum, uint8_t MediaType, TArray<struct
FMarketplaceInventoryItem>& InventoryItems);
void ClearReadInventoryItemsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadInventoryItemsCompleteDelegate);
void AddReadInventoryItemsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadInventoryItemsCompleteDelegate);
void OnReadInventoryItemsComplete(uint8_t MediaType);
bool ReadInventoryItems(uint8_t LocalUserNum, uint8_t MediaType);
void ClearReadAdditionalProductDetailsCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate ReadAdditionalProductDetailsCompleteDelegate);
void AddReadAdditionalProductDetailsCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate ReadAdditionalProductDetailsCompleteDelegate);
void OnReadAdditionalProductDetailsComplete(uint8_t MediaType);
bool ReadAdditionalDetailsForProducts(uint8_t LocalUserNum, uint8_t MediaType);
void ResetAvailableProducts(uint8_t LocalUserNum, uint8_t MediaType);
uint8_t GetAvailableProducts(uint8_t LocalUserNum, uint8_t MediaType, TArray<struct
FMarketplaceProductDetails>& AvailableProducts);
void ClearReadAvailableProductsCompleteDelegate(uint8_t LocalUserNum, struct
FScriptDelegate ReadAvailableProductsCompleteDelegate);
void AddReadAvailableProductsCompleteDelegate(uint8_t LocalUserNum, struct FScriptDelegate
ReadAvailableProductsCompleteDelegate);
void OnReadAvailableProductsComplete(uint8_t MediaType);
bool ReadAvailableProducts(uint8_t LocalUserNum, class FString ParentId, uint8_t
ParentMediaType, uint8_t ChildMediaType, uint8_t SortOrder);
};

// Class Engine.OnlineAccountInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineAccountInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineAccountInterface");
}

return uClassPointer;
};

bool GetLocalAccountNames(TArray<class FString>& Accounts);
bool DeleteLocalAccount(class FString Username, class FString Password);
bool RenameLocalAccount(class FString NewUserName, class FString OldUserName, class
FString Password);
bool CreateLocalAccount(class FString Username, class FString Password);
void ClearCreateOnlineAccountCompletedDelegate(struct FScriptDelegate
AccountCreateDelegate);
void AddCreateOnlineAccountCompletedDelegate(struct FScriptDelegate
AccountCreateDelegate);
void OnCreateOnlineAccountCompleted(uint8_t ErrorStatus);
bool CreateOnlineAccount(class FString Username, class FString Password, class FString
EmailAddress, class FString ProductKey);
};

// Class Engine.DynamicPhysicsVolume
// 0x0004 (0x02D8 - 0x02DC)
class ADynamicPhysicsVolume : public APhysicsVolume
{
public:
unsigned long                          bEnabled : 1;                          // 0x02D8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.DynamicPhysicsVolume");
}

return uClassPointer;
};

void eventPostBeginPlay();
};

// Class Engine.DynamicSMActor_Spawnable
// 0x0000 (0x02C8 - 0x02C8)
class ADynamicSMActor_Spawnable : public ADynamicSMActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DynamicSMActor_Spawnable");
}

return uClassPointer;
};

};

// Class Engine.DynamicTriggerVolume
// 0x0004 (0x02A8 - 0x02AC)
class ADynamicTriggerVolume : public ATriggerVolume
{
public:
unsigned long                    bEnabled : 1;                    // 0x02A8 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.DynamicTriggerVolume");
}

return uClassPointer;
};

void eventPostBeginPlay();
};
```

```
// Class Engine.SeqAct_SetParticleSysParam
// 0x0024 (0x0160 - 0x0184)
class USeqAct_SetParticleSysParam : public USequenceAction
{
public:
TArray<struct FParticleSysParam>              InstanceParameters;                        // 0x0160
(0x0010) [0x0000000004480009] (CPF_Edit | CPF_ExportObject | CPF_Component |
CPF_NeedCtorLink | CPF_EditInline)
unsigned long                                 bOverrideScalar : 1;                       // 0x0170 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                                 bOverrideVector : 1;                       // 0x0170 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
float                                         ScalarValue;                               // 0x0174 (0x0004)
[0x0000000000000001] (CPF_Edit)
struct FVector                                VectorValue;                               // 0x0178 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetParticleSysParam");
}

return uClassPointer;
};

};

// Class Engine.EngineContent
// 0x0010 (0x0060 - 0x0070)
class UEngineContent : public UObject
{
public:
TArray<class UObject*>                        Content;                                   // 0x0060 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EngineContent");
}

return uClassPointer;
};

};
```

```
// Class Engine.EpochNow_RealTime
// 0x0000 (0x0060 - 0x0060)
class UEpochNow_RealTime : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.EpochNow_RealTime");
}

return uClassPointer;
};

uint64_t EpochNow();
};

// Class Engine.IEpochNow
// 0x0000 (0x0060 - 0x0060)
class UIEpochNow : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.IEpochNow");
}

return uClassPointer;
};

uint64_t EpochNow();
};

// Class Engine.ExponentialHeightFog
// 0x000C (0x0268 - 0x0274)
class AExponentialHeightFog : public AInfo
{
public:
class UExponentialHeightFogComponent*            Component;                              // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                                    bEnabled : 1;                           // 0x0270 (0x0004)
```

```
[0x0000000100000020] [0x00000001] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ExponentialHeightFog");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.FracturedSMActorSpawnable
// 0x0000 (0x02F0 - 0x02F0)
class AFracturedSMActorSpawnable : public AFracturedStaticMeshActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.FracturedSMActorSpawnable");
}

return uClassPointer;
};

};

// Class Engine.FracturedStaticMeshActor_Spawnable
// 0x0000 (0x02F0 - 0x02F0)
class AFracturedStaticMeshActor_Spawnable : public AFracturedStaticMeshActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.FracturedStaticMeshActor_Spawnable");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_PlayerSpawned
// 0x001C (0x017C - 0x0198)
class USeqEvent_PlayerSpawned : public USequenceEvent
{
public:
struct FVector                      SpawnLocation;                    // 0x0180 (0x000C)
[0x0000000000000000]
struct FRotator                     SpawnRotation;                    // 0x018C (0x000C)
[0x0000000000000000]

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_PlayerSpawned");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ControlMovieTexture
// 0x0008 (0x0160 - 0x0168)
class USeqAct_ControlMovieTexture : public USequenceAction
{
public:
class UTextureMovie*                   MovieTexture;                    // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ControlMovieTexture");
}

return uClassPointer;
};
```

```
void eventActivated();
};

// Class Engine.PrimaryPlayer
// 0x0000 (0x0060 - 0x0060)
class UPrimaryPlayer : public UObject
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PrimaryPlayer");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ToggleInput
// 0x0004 (0x0160 - 0x0164)
class USeqAct_ToggleInput : public USeqAct_Toggle
{
public:
unsigned long                     bToggleMovement : 1;               // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                     bToggleTurning : 1;               // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleInput");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ToggleHUD
// 0x0000 (0x0160 - 0x0160)
class USeqAct_ToggleHUD : public USequenceAction
{
public:
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleHUD");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_ForceFeedback
// 0x0010 (0x0160 - 0x0170)
class USeqAct_ForceFeedback : public USequenceAction
{
public:
class UForceFeedbackWaveform*           FFWaveform;                         // 0x0160
(0x0008) [0x0000000004000001] (CPF_Edit | CPF_EditInline)
class UClass*                           PredefinedWaveForm;                 // 0x0168 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ForceFeedback");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ToggleCinematicMode
// 0x0004 (0x0160 - 0x0164)
class USeqAct_ToggleCinematicMode : public USequenceAction
{
public:
unsigned long                           bDisableMovement : 1;               // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                           bDisableTurning : 1;                // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                           bHidePlayer : 1;                    // 0x0160 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                           bDisableInput : 1;                  // 0x0160 (0x0004)
```

```
[0x0000000000000001] [0x00000008] (CPF_Edit)
unsigned long                           bHideHUD : 1;                           // 0x0160 (0x0004)
[0x0000000000000001] [0x00000010] (CPF_Edit)
unsigned long                           bDeadBodies : 1;                        // 0x0160 (0x0004)
[0x0000000000000001] [0x00000020] (CPF_Edit)
unsigned long                           bDroppedPickups : 1;                    // 0x0160 (0x0004)
[0x0000000000000001] [0x00000040] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleCinematicMode");
}

return uClassPointer;
};

void eventActivated();
};

// Class Engine.SeqAct_ConsoleCommand
// 0x0020 (0x0160 - 0x0180)
class USeqAct_ConsoleCommand : public USequenceAction
{
public:
class FString                           Command;                                // 0x0160 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<class FString>                   Commands;                               // 0x0170 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ConsoleCommand");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
void VersionUpdated(int32_t OldVersion, int32_t NewVersion);
};

// Class Engine.SeqAct_FlyThroughHasEnded
// 0x0000 (0x0160 - 0x0160)
class USeqAct_FlyThroughHasEnded : public USequenceAction
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_FlyThroughHasEnded");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetSoundMode
// 0x000C (0x0160 - 0x016C)
class USeqAct_SetSoundMode : public USequenceAction
{
public:
class USoundMode*                    SoundMode;                    // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
unsigned long                   bTopPriority : 1;               // 0x0168 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetSoundMode");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
void eventActivated();
};

// Class Engine.HeightFog
// 0x000C (0x0268 - 0x0274)
class AHeightFog : public AInfo
{
public:
class UHeightFogComponent*              Component;                    // 0x0268 (0x0008)
[0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)
unsigned long                   bEnabled : 1;                   // 0x0270 (0x0004)
```

```
[0x0000000100000020] [0x00000001] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.HeightFog");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
void eventReplicatedEvent(struct FName VarName);
void eventPostBeginPlay();
};

// Class Engine.InterpActor_ForCinematic
// 0x0000 (0x0318 - 0x0318)
class AInterpActor_ForCinematic : public AInterpActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.InterpActor_ForCinematic");
}

return uClassPointer;
};

};

// Class Engine.IPoolable
// 0x0000 (0x0060 - 0x0060)
class UIPoolable : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.IPoolable");
}

return uClassPointer;
};

void OnPoolReset();
};

// Class Engine.MaterialInstanceTimeVaryingActor
// 0x0008 (0x0268 - 0x0270)
class AMaterialInstanceTimeVaryingActor : public AActor
{
public:
class UMaterialInstanceTimeVarying*          MatInst;                          // 0x0268 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.MaterialInstanceTimeVaryingActor");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_AssignController
// 0x0008 (0x0160 - 0x0168)
class USeqAct_AssignController : public USequenceAction
{
public:
class UClass*                            ControllerClass;                   // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AssignController");
}

return uClassPointer;
};

};
```

```
// Class Engine.NavMeshBoundsVolume
// 0x0004 (0x02A4 - 0x02A8)
class ANavMeshBoundsVolume : public AVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.NavMeshBoundsVolume");
}

return uClassPointer;
};

};

// Class Engine.OnlineEventsInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineEventsInterface : public UInterface
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineEventsInterface");
}

return uClassPointer;
};

bool UpdatePlaylistPopulation(int32_t PlaylistId, int32_t NumPlayers);
bool UploadGameplayEventsData(struct FUniqueNetId UniqueId, TArray<uint8_t>& Payload);
bool UploadPlayerData(struct FUniqueNetId UniqueId, class FString PlayerNick, class
UOnlineProfileSettings* ProfileSettings, class UOnlinePlayerStorage* PlayerStorage);
};

// Class Engine.OnlinePlaylistGameTypeProvider
// 0x0030 (0x009C - 0x00CC)
class UOnlinePlaylistGameTypeProvider : public UUIResourceDataProvider
{
public:
struct FName                    PlaylistGameTypeName;              // 0x00A0 (0x0008)
[0x0000000000004000] (CPF_Config)
```

```
class FString                              DisplayName;                              // 0x00A8 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
class FString                              Description;                             // 0x00B8 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
int32_t                                    GameTypeId;                              // 0x00C8 (0x0004)
[0x0000000000004000] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlinePlaylistGameTypeProvider");
}

return uClassPointer;
};

};

// Class Engine.OnlineRecentPlayersList
// 0x00A8 (0x0060 - 0x0108)
class UOnlineRecentPlayersList : public UObject
{
public:
TArray<struct FUniqueNetId>                RecentPlayers;                           // 0x0060 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FFriendHistoryKey>           RecentPlayerKeys;                        // 0x0070
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)
TArray<struct FRecentParty>                RecentParties;                           // 0x0080 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)
struct FRecentParty                        LastParty;                               // 0x0090 (0x0058)
[0x0000000000400000] (CPF_NeedCtorLink)
int32_t                                    MaxRecentPlayers;                        // 0x00E8 (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                    MaxRecentParties;                        // 0x00EC (0x0004)
[0x0000000000004000] (CPF_Config)
int32_t                                    RecentPlayersAddIndex;                   // 0x00F0 (0x0004)
[0x0000000000000000]
int32_t                                    RecentPartiesAddIndex;                   // 0x00F4 (0x0004)
[0x0000000000000000]
TArray<struct FCurrentPlayerMet>           CurrentPlayers;                          // 0x00F8
(0x0010) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineRecentPlayersList");
}

return uClassPointer;
};

int32_t GetCurrentPlayersListCount();
void SetCurrentPlayersList(TArray<struct FCurrentPlayerMet> Players);
bool ShowCurrentPlayersList(uint8_t LocalUserNum, class FString Title, class FString
Description);
bool ShowLastPartyPlayerList(uint8_t LocalUserNum, class FString Title, class FString
Description);
bool ShowRecentPartiesPlayerList(uint8_t LocalUserNum, class FString Title, class FString
Description);
bool ShowRecentPlayerList(uint8_t LocalUserNum, class FString Title, class FString Description);
void SetLastParty(struct FUniqueNetId PartyLeader, TArray<struct FUniqueNetId>&
PartyMembers);
int32_t GetTeamForCurrentPlayer(struct FUniqueNetId Player);
int32_t GetSkillForCurrentPlayer(struct FUniqueNetId Player);
void GetPlayersFromCurrentPlayers(TArray<struct FUniqueNetId>& Players);
void GetPlayersFromRecentParties(TArray<struct FUniqueNetId>& Players);
void ClearRecentParties();
void AddPartyToRecentParties(struct FUniqueNetId PartyLeader, TArray<struct FUniqueNetId>&
PartyMembers);
void ClearRecentPlayers();
void AddPlayerToRecentPlayers(struct FUniqueNetId NewPlayer, TArray<uint8_t> PlayerKey,
class FString PlayerName);
void ReplacePlayerHistoryForPlayer(struct FUniqueNetId NewPlayer, TArray<uint8_t> PlayerKey);
};

// Class Engine.OnlineStatsWrite_TA
// 0x0000 (0x00D0 - 0x00D0)
class UOnlineStatsWrite_TA : public UOnlineStatsWrite
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineStatsWrite_TA");
}

return uClassPointer;
};

void SetIntStat(int32_t StatId, int32_t Value);
};

// Class Engine.OnlineSuppliedUIInterface
// 0x0000 (0x0060 - 0x0060)
class UOnlineSuppliedUIInterface : public UInterface
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OnlineSuppliedUIInterface");
}

return uClassPointer;
};

bool ShowMatchmakingUI(uint8_t SearchingPlayerNum, class UOnlineGameSearch*
SearchSettings, class UOnlineGameSettings* GameSettings);
void ClearShowOnlineStatsUICompleteDelegate(struct FScriptDelegate
ShowOnlineStatsUICompleteDelegate);
void AddShowOnlineStatsUICompleteDelegate(struct FScriptDelegate
ShowOnlineStatsUICompleteDelegate);
bool ShowOnlineStatsUI(class UOnlineStatsRead* StatsRead, TArray<struct FUniqueNetId>&
Players);
void OnShowOnlineStatsUIComplete();
};

// Class Engine.OwnerReplicatedActor_ORS
// 0x0000 (0x0268 - 0x0268)
class AOwnerReplicatedActor_ORS : public AActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.OwnerReplicatedActor_ORS");
}

return uClassPointer;
};

void eventDestroyed();
void eventOnOwnerChanged();
};

// Class Engine.PathNode_Dynamic
// 0x0000 (0x0388 - 0x0388)
class APathNode_Dynamic : public APathNode
{
```

```
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.PathNode_Dynamic");
}

return uClassPointer;
};

class FString eventGetDebugAbbrev();
};

// Class Engine.SeqEvent_AIReachedRouteActor
// 0x0004 (0x017C - 0x0180)
class USeqEvent_AIReachedRouteActor : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_AIReachedRouteActor");
}

return uClassPointer;
};

};

// Class Engine.RadialBlurActor
// 0x0008 (0x0268 - 0x0270)
class ARadialBlurActor : public AActor
{
public:
class URadialBlurComponent*                    RadialBlur;                              // 0x0268 (0x0008)
[0x0000000004080009] (CPF_Edit | CPF_ExportObject | CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
```

```
uClassPointer = UObject::FindClass("Class Engine.RadialBlurActor");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_ToggleConstraintDrive
// 0x0004 (0x0160 - 0x0164)
class USeqAct_ToggleConstraintDrive : public USequenceAction
{
public:
unsigned long                    bEnableAngularPositionDrive : 1;          // 0x0160 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
unsigned long                    bEnableAngularVelocityDrive : 1;          // 0x0160 (0x0004)
[0x0000000000000001] [0x00000002] (CPF_Edit)
unsigned long                    bEnableLinearPositionDrive : 1;           // 0x0160 (0x0004)
[0x0000000000000001] [0x00000004] (CPF_Edit)
unsigned long                    bEnableLinearvelocityDrive : 1;           // 0x0160 (0x0004)
[0x0000000000000001] [0x00000008] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleConstraintDrive");
}

return uClassPointer;
};

};

// Class Engine.RB_BSJointActor
// 0x0000 (0x02A0 - 0x02A0)
class ARB_BSJointActor : public ARB_ConstraintActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_BSJointActor");
}

return uClassPointer;
```

```
};

};

// Class Engine.RB_ConstraintActorSpawnable
// 0x0000 (0x02A0 - 0x02A0)
class ARB_ConstraintActorSpawnable : public ARB_ConstraintActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_ConstraintActorSpawnable");
}

return uClassPointer;
};

};

// Class Engine.RB_HingeActor
// 0x0000 (0x02A0 - 0x02A0)
class ARB_HingeActor : public ARB_ConstraintActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_HingeActor");
}

return uClassPointer;
};

};

// Class Engine.RB_PrismaticActor
// 0x0000 (0x02A0 - 0x02A0)
class ARB_PrismaticActor : public ARB_ConstraintActor
{
public:

public:
static UClass* StaticClass()
```

```
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_PrismaticActor");
}

return uClassPointer;
};

};

// Class Engine.RB_PulleyJointActor
// 0x0000 (0x02A0 - 0x02A0)
class ARB_PulleyJointActor : public ARB_ConstraintActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.RB_PulleyJointActor");
}

return uClassPointer;
};

};

// Class Engine.ReplicatedActor_ORS
// 0x0008 (0x0268 - 0x0270)
class AReplicatedActor_ORS : public AActor
{
public:
class AActor*                    ReplicatedOwner;                 // 0x0268 (0x0008)
[0x0000000100000020] (CPF_Net)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ReplicatedActor_ORS");
}

return uClassPointer;
};
```

```
void eventDestroyed();
void eventOnOwnerChanged();
void eventReplicatedEvent(struct FName VarName);
};

// Class Engine.ReverbVolumeToggleable
// 0x0000 (0x02F0 - 0x02F0)
class AReverbVolumeToggleable : public AReverbVolume
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.ReverbVolumeToggleable");
}

return uClassPointer;
};

void OnToggle(class USeqAct_Toggle* Action);
};

// Class Engine.SeqAct_AddRemoveFaceFXAnimSet
// 0x0010 (0x0160 - 0x0170)
class USeqAct_AddRemoveFaceFXAnimSet : public USequenceAction
{
public:
TArray<class UFaceFXAnimSet*>                 FaceFXAnimSets;                         // 0x0160
(0x0010) [0x0000000020400000] (CPF_NeedCtorLink | CPF_Deprecated)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AddRemoveFaceFXAnimSet");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_AIAbortMoveToActor
// 0x0000 (0x0160 - 0x0160)
class USeqAct_AIAbortMoveToActor : public USequenceAction
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_AIAbortMoveToActor");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_MITV_Activate
// 0x0004 (0x0160 - 0x0164)
class USeqAct_MITV_Activate : public USequenceAction
{
public:
float                                   DurationOfMITV;                         // 0x0160 (0x0004)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_MITV_Activate");
}

return uClassPointer;
};

void eventActivated();
static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SetMatInstTexParam
// 0x0018 (0x0160 - 0x0178)
class USeqAct_SetMatInstTexParam : public USequenceAction
{
public:
class UMaterialInstanceConstant*                MatInst;                                // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
class UTexture*                         NewTexture;                             // 0x0168 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                            ParamName;                              // 0x0170 (0x0008)
[0x0000000000000001] (CPF_Edit)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetMatInstTexParam");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetMatInstVectorParam
// 0x0020 (0x0160 - 0x0180)
class USeqAct_SetMatInstVectorParam : public USequenceAction
{
public:
class UMaterialInstanceConstant*          MatInst;                          // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FName                              ParamName;                        // 0x0168 (0x0008)
[0x0000000000000001] (CPF_Edit)
struct FLinearColor                       VectorValue;                      // 0x0170 (0x0010)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetMatInstVectorParam");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqAct_SetSkelControlTarget
// 0x0018 (0x0160 - 0x0178)
class USeqAct_SetSkelControlTarget : public USequenceAction
{
public:
struct FName                              SkelControlName;                  // 0x0160 (0x0008)
[0x0000000000000001] (CPF_Edit)
TArray<class UObject*>                    TargetActors;                     // 0x0168 (0x0010)
[0x0000000000400001] (CPF_Edit | CPF_NeedCtorLink)
```

```
public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetSkelControlTarget");
}

return uClassPointer;
};

};

// Class Engine.SeqAct_SetVector
// 0x000C (0x0160 - 0x016C)
class USeqAct_SetVector : public USeqAct_SetSequenceVariable
{
public:
struct FVector                          DefaultValue;                           // 0x0160 (0x000C)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_SetVector");
}

return uClassPointer;
};

void eventActivated();
};

// Class Engine.SeqAct_ToggleAffectedByHitEffects
// 0x0000 (0x0160 - 0x0160)
class USeqAct_ToggleAffectedByHitEffects : public USequenceAction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleAffectedByHitEffects");
}
return uClassPointer;
};

};

// Class Engine.SeqAct_ToggleHiddenGame
// 0x0000 (0x0160 - 0x0160)
class USeqAct_ToggleHiddenGame : public USeqAct_Toggle
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_ToggleHiddenGame");
}

return uClassPointer;
};

void ModifyActorComponentsVisibility(class AActor* ActorToModify);
void eventActivated();
};

// Class Engine.SeqAct_UpdatePhysBonesFromAnim
// 0x0000 (0x0160 - 0x0160)
class USeqAct_UpdatePhysBonesFromAnim : public USequenceAction
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqAct_UpdatePhysBonesFromAnim");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqEvent_Death
// 0x0004 (0x017C - 0x0180)
class USeqEvent_Death : public USequenceEvent
```

```
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_Death");
}

return uClassPointer;
};

};

// Class Engine.SeqEvent_LOS
// 0x0010 (0x017C - 0x018C)
class USeqEvent_LOS : public USequenceEvent
{
public:
float                              ScreenCenterDistance;                  // 0x0180 (0x0004)
[0x0000000000000001] (CPF_Edit)
float                              TriggerDistance;                       // 0x0184 (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                           bCheckForObstructions : 1;              // 0x0188 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_LOS");
}

return uClassPointer;
};

static int32_t eventGetObjClassVersion();
};

// Class Engine.SeqEvent_PickupStatusChange
// 0x0004 (0x017C - 0x0180)
class USeqEvent_PickupStatusChange : public USequenceEvent
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqEvent_PickupStatusChange");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Byte
// 0x0000 (0x00E0 - 0x00E0)
class USeqVar_Byte : public USequenceVariable
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Byte");
}

return uClassPointer;
};

};

// Class Engine.SeqVar_Name
// 0x0000 (0x00E0 - 0x00E0)
class USeqVar_Name : public USequenceVariable
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Name");
}

return uClassPointer;
};

};
```

```
// Class Engine.SeqVar_Union
// 0x0000 (0x00E0 - 0x00E0)
class USeqVar_Union : public USequenceVariable
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SeqVar_Union");
}

return uClassPointer;
};

};

// Class Engine.SkeletalMeshActorMATSpawnable
// 0x0000 (0x02E0 - 0x02E0)
class ASkeletalMeshActorMATSpawnable : public ASkeletalMeshActorMAT
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshActorMATSpawnable");
}

return uClassPointer;
};

};

// Class Engine.SkeletalMeshActorMATWalkable
// 0x0000 (0x02E0 - 0x02E0)
class ASkeletalMeshActorMATWalkable : public ASkeletalMeshActorMAT
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.SkeletalMeshActorMATWalkable");
}

return uClassPointer;
};

};

// Class Engine.StaticCameraActor
// 0x0000 (0x03F8 - 0x03F8)
class AStaticCameraActor : public ACameraActor
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticCameraActor");
}

return uClassPointer;
};

};

// Class Engine.StaticLensFlareSource
// 0x0004 (0x0274 - 0x0278)
class AStaticLensFlareSource : public ALensFlareSource
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.StaticLensFlareSource");
}

return uClassPointer;
};

};

// Class Engine.TimeWindow
// 0x0014 (0x0060 - 0x0074)
```

```
class UTimeWindow : public UObject
{
public:
uint64_t                                   WindowStartTime;                          // 0x0060 (0x0008)
[0x0000000000000001] (CPF_Edit)
int32_t                                    WindowDuration;                           // 0x0068 (0x0004)
[0x0000000000000001] (CPF_Edit)
int32_t                                    WindowResetInterval;                      // 0x006C (0x0004)
[0x0000000000000001] (CPF_Edit)
unsigned long                              bRepeatable : 1;                          // 0x0070 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TimeWindow");
}

return uClassPointer;
};

bool IsActive();
uint64_t GetEndTime(uint64_t inTime);
uint64_t GetEndTimeFromNow();
uint64_t GetStartTime(uint64_t inTime);
uint64_t GetStartTimeFromNow();
};

// Class Engine.Trigger_Dynamic
// 0x0000 (0x0278 - 0x0278)
class ATrigger_Dynamic : public ATrigger
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Trigger_Dynamic");
}

return uClassPointer;
};

};

// Class Engine.Trigger_LOS
// 0x0010 (0x0278 - 0x0288)
class ATrigger_LOS : public ATrigger
{
public:
TArray<class APlayerController*>                PCsWithLOS;                              // 0x0278 (0x0010)
[0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.Trigger_LOS");
}

return uClassPointer;
};

void eventTick(float DeltaTime);
};

// Class Engine.TriggeredPath
// 0x0014 (0x0384 - 0x0398)
class ATriggeredPath : public ANavigationPoint
{
public:
unsigned long                               bOpen : 1;                              // 0x0388 (0x0004)
[0x0000000000000001] [0x00000001] (CPF_Edit)
class AActor*                               MyTrigger;                              // 0x0390 (0x0008)
[0x0000000000000001] (CPF_Edit)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TriggeredPath");
}

return uClassPointer;
};

bool eventSuggestMovePreparation(class APawn* Other);
class AActor* eventSpecialHandling(class APawn* Other);
void OnToggle(class USeqAct_Toggle* inAction);
};

// Class Engine.TriggerStreamingLevel
// 0x0010 (0x0278 - 0x0288)
class ATriggerStreamingLevel : public ATrigger
```

```
{
public:
TArray<struct FLevelStreamingData>              Levels;                                     // 0x0278 (0x0010)
[0x0000000004400001] (CPF_Edit | CPF_NeedCtorLink | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.TriggerStreamingLevel");
}

return uClassPointer;
};

void eventTouch(class AActor* Other, class UPrimitiveComponent* OtherComp, struct FVector
HitLocation, struct FVector HitNormal);
};

// Class Engine.UICharacterSummary
// 0x0038 (0x009C - 0x00D4)
class UUICharacterSummary : public UUIResourceDataProvider
{
public:
class FString                          ClassPathName;                          // 0x00A0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                          CharacterName;                          // 0x00B0 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
class FString                          CharacterBio;                          // 0x00C0 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
unsigned long                          bIsDisabled : 1;                        // 0x00D0 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UICharacterSummary");
}

return uClassPointer;
};

};

// Class Engine.UIGameInfoSummary
// 0x006C (0x009C - 0x0108)
class UUIGameInfoSummary : public UUIResourceDataProvider
```

```
{
public:
class FString                    ClassName;                    // 0x00A0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                    GameAcronym;                    // 0x00B0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                    MapPrefix;                    // 0x00C0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
unsigned long                     bIsTeamGame : 1;                    // 0x00D0 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)
unsigned long                     bIsDisabled : 1;                    // 0x00D0 (0x0004)
[0x0000000000004000] [0x00000002] (CPF_Config)
class FString                    GameSettingsClassName;                    // 0x00D8 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                    GameName;                    // 0x00E8 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
class FString                    Description;                    // 0x00F8 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIGameInfoSummary");
}

return uClassPointer;
};

};

// Class Engine.UIMapSummary
// 0x0044 (0x009C - 0x00E0)
class UUIMapSummary : public UUIResourceDataProvider
{
public:
class FString                    MapName;                    // 0x00A0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                    ScreenshotPathName;                    // 0x00B0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                    DisplayName;                    // 0x00C0 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)
class FString                    Description;                    // 0x00D0 (0x0010)
[0x0000000000408002] (CPF_Const | CPF_Localized | CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIMapSummary");
}

return uClassPointer;
};

};

// Class Engine.UIWeaponSummary
// 0x0038 (0x009C - 0x00D4)
class UUIWeaponSummary : public UUIResourceDataProvider
{
public:
class FString                    ClassPathName;                  // 0x00A0 (0x0010)
[0x0000000000404000] (CPF_Config | CPF_NeedCtorLink)
class FString                    FriendlyName;                   // 0x00B0 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
class FString                    WeaponDescription;              // 0x00C0 (0x0010)
[0x000000000040C002] (CPF_Const | CPF_Config | CPF_Localized | CPF_NeedCtorLink)
unsigned long                     bIsDisabled : 1;                // 0x00D0 (0x0004)
[0x0000000000004000] [0x00000001] (CPF_Config)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.UIWeaponSummary");
}

return uClassPointer;
};

};

// Class Engine.UserCloudFileCloudSaveSystemDataBlobStore
// 0x00A0 (0x0060 - 0x0100)
class UUserCloudFileCloudSaveSystemDataBlobStore : public UObject
{
public:
class UUserCloudFileInterface*          UserCloudFile_Object;              // 0x0060
(0x0008) [0x0000000000002000] (CPF_Transient)
class UUserCloudFileInterface*          UserCloudFile_Interface;           // 0x0068
(0x0008) [0x0000000000002000] (CPF_Transient)
struct FScriptDelegate               GetDataBlobCallback;               // 0x0070 (0x0018)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FScriptDelegate               SetDataBlobCallback;               // 0x0088 (0x0018)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FScriptDelegate               DeleteDataBlobCallback;            // 0x00A0 (0x0018)
[0x0000000000402000] (CPF_Transient | CPF_NeedCtorLink)
struct FScriptDelegate               __GetDataBlobCallbackDelegate__Delegate;     // 0x00B8
```

```
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                    __SetDataBlobCallbackDelegate__Delegate;      // 0x00D0
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)
struct FScriptDelegate                    __DeleteDataBlobCallbackDelegate__Delegate;   // 0x00E8
(0x0018) [0x0000000000400000] (CPF_NeedCtorLink)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class
Engine.UserCloudFileCloudSaveSystemDataBlobStore");
}

return uClassPointer;
};

void OnDeleteUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
bool DeleteDataBlob(class FString StorageID, class FString BlobName, struct FScriptDelegate
InDeleteDataBlobCallback);
void OnWriteUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void SetDataBlob(class FString StorageID, class FString BlobName, struct FScriptDelegate
InSetDataBlobCallback, TArray<uint8_t>& DataBlob);
void OnReadUserFileComplete(unsigned long bWasSuccessful, class FString UserId, class
FString Filename);
void GetDataBlob(class FString StorageID, class FString BlobName, struct FScriptDelegate
InGetDataBlobCallback);
void Init(class UUserCloudFileInterface* InUserCloudFile);
void DeleteDataBlobCallbackDelegate(unsigned long bWasSucessfull, class FString StorageID,
class FString BlobName, class FString Error);
void SetDataBlobCallbackDelegate(unsigned long bWasSucessfull, class FString StorageID, class
FString BlobName, class FString Error);
void GetDataBlobCallbackDelegate(unsigned long bWasSuccessful, class FString StorageID,
class FString BlobName, class FString Error, TArray<uint8_t>& DataBlob);
};

// Class Engine.WindDirectionalSource
// 0x0008 (0x0268 - 0x0270)
class AWindDirectionalSource : public AInfo
{
public:
class UWindDirectionalSourceComponent*         Component;                          // 0x0268
(0x0008) [0x00000000040A000B] (CPF_Edit | CPF_Const | CPF_ExportObject | CPF_EditConst |
CPF_Component | CPF_EditInline)

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;
if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WindDirectionalSource");
}

return uClassPointer;
};

};

// Class Engine.WindDirectionalSourceDynamic
// 0x0000 (0x0270 - 0x0270)
class AWindDirectionalSourceDynamic : public AWindDirectionalSource
{
public:

public:
static UClass* StaticClass()
{
static UClass* uClassPointer = nullptr;

if (!uClassPointer)
{
uClassPointer = UObject::FindClass("Class Engine.WindDirectionalSourceDynamic");
}

return uClassPointer;
};

};

/*
#
===============================================================================
=========== #
#
#
===============================================================================
=========== #
*/

#ifdef _MSC_VER
#pragma pack(pop)
#endif
```

Removed: 1

Added: 1

Generated at https://www.textcompare.org/ on 05/06/2024, 18:04:24