```
//GENERATED
CODE================================================================================
====
//
//  Copyright Blizzard Entertainment 2003+
//
//  ActorHints.txt generated from Actor.xml using HintText.xtc
//
//  !!! WARNING !!! This is generated code. Changes made here will be overwritten.
//
//GENERATED
CODE================================================================================
====
```

EDSTR_ENUMHINT_e_actorPlayerIdSourceAuto=The actor gets its player id from its creator. This player id can come from the creating actor, or from game synchronous code, via Abil, Behavior, Effect and Item messages (and their various subclass messages, like AbilTransport and BehaviorLevel).
EDSTR_ENUMHINT_e_actorPlayerIdSourceContainingScope=Legacy behavior. The actor gets its player id from its owning scope at creation time. If a scope belongs to a unit, it has the player id of that unit. The player ids of other kinds of scopes vary by situation.
EDSTR_ENUMHINT_e_actorBaselineStand=The baseline that plays when a CActorUnit is not moving.
EDSTR_ENUMHINT_e_actorBaselineWalk=The baseline that plays when a CActorUnit is moving.
EDSTR_ENUMHINT_e_actorFlagAddToExternalFinder=Add this actor to the external finder when it is created. The external finder enables any actor within a scene to find any other actor within that scene, regardless of which scope it is in.
EDSTR_ENUMHINT_e_actorFlagSuppressCreationErrors=Prevents this actor from generating an error message when it cannot be created. Error messages are typically helpful during development, but some kinds of code and/or data can benefit from being able to have objects silently fail to create.
EDSTR_ENUMHINT_e_actorFlagSuppressSaveLoad=Prevents this actor from being restored during the load of a save game. Can improve save/load speed and shrink save file size when used on actors the player won't notice are missing after a load.
EDSTR_ENUMHINT_e_actorFlagRespondsToUnitPlayerChange=Enables an actor to update its player id dynamically in response to UnitPlayerChange messages sent from the game. Can be used to keep colors in sync when a unit changes sides.
EDSTR_ENUMHINT_e_actorFlagVisionTestCenterOnly=Tests only the center part of the actor for vision tests, without taking the radius into account.
EDSTR_ENUMHINT_e_actorFlagSkipPreload=Skips preload.
EDSTR_ENUMHINT_e_actorDoodadFlagDefaultToModelAABBBounds=Specifies that the doodad gets its bounding radius from its axially-aligned bounding box (AABB), which often results in smaller, higher performance bounds. This only applies if the RadiusLoose field is not specified in the model's ModelData.xml entry.
EDSTR_ENUMHINT_e_actorDoodadFlagFootprintOnly=Marks a doodad to apply its footprint to the map, but not create any visuals.
EDSTR_ENUMHINT_e_actorDoodadFlagPauseAnimsWhileFogged=Largely static models generally look fine when paused, but highly dynamic models, such as waterfalls and windmills, do not.
EDSTR_ENUMHINT_e_actorScopeBearingsTrackingAutomaticBySituation=Causes actors to determine whether they track their parent scope based on context. For instance, by default, actors in a unit's scope or a doodad's scope in the editor automatically track the bearings of their parent scope. (This is so they automatically follow their owning scope if no other data is added, since this is frequently the preferred behavior.)
EDSTR_ENUMHINT_e_actorScopeBearingsTrackingForceOn=Force an actor to track the bearings of its parent scope, if it is not configured otherwise (for example by hosting or using Site Ops).
EDSTR_ENUMHINT_e_actorScopeBearingsTrackingForceOff=Force an actor to NOT track the bearings of its parent scope, unless configured otherwise, perhaps via a Site Op.
EDSTR_ENUMHINT_e_actorModelFlagAllowHitTest=Determines whether an actor can be clicked on or selected by the player.

EDSTR_ENUMHINT_e_actorModelFlagAutomatedGlobalLoopsIgnoreOrphan=Causes automated global loops to not transition out when the CActorModel's owning actor scope is orphaned.
EDSTR_ENUMHINT_e_actorModelFlagAutomateGlobalLoops=Cause the actor to look through the animations of its model and automatically start them when the actor is created. Global loops are animation brackets with GLbirth, GLstand, GLdeath and/or GLdead animations that can be played automatically when a model is created. They are typically used for continuous 'background' anims that need to play regardless of what the unit is doing. The spinning rotor blades on the Banshee are an example.
EDSTR_ENUMHINT_e_actorModelFlagCloakedUndetectedIsHidden=Causes a model to be invisible to enemies when it is not detected. When present, overrides the ModelData.xml version of the flag.
EDSTR_ENUMHINT_e_actorModelFlagIgnoreWalkables=Prevents 'walkable' models from adjusting the height of this actor. Can be useful when positioning doodad actors such that they intersect with a walkable model to varying degrees, rather than always being on top of it (this can make maps feel more natural).
EDSTR_ENUMHINT_e_actorModelFlagIgnoreTipabilityWithHostOrSiteOps=While the CActorModel is getting its bearings from a host or from siteOps, it ignores the model's tipability setting in ModelData.xml.
EDSTR_ENUMHINT_e_actorModelFlagNeedsVolumeMonitoring=Marks an actor as containing target attach volumes (TAVs) that the engine must monitor during attacks. (TAVs cause an attack to appear to hit the surface of its target, rather than its center.) For instance, a unit attacking a Command Center during construction must be aware of the construction scaffolding TAVs -- even though they are part of a separate model -- so that it can appear to impact the scaffolding while it is present.
EDSTR_ENUMHINT_e_actorModelFlagSuppressMissingAttachErrors=Prevents errors when an attach query does not find any results and attempts to use a fallback attach point that is missing. Useful for widely used default CActorModels that may be used with some models that do not have a complete set of basic attach points.
EDSTR_ENUMHINT_e_actorModelFlagSuppressPlayerDecals=Prevents decal textures from appearing on the model contained by this actor.
EDSTR_ENUMHINT_e_actorModelFlagUpdateVisibility=Enables the actor to update its visibility in response to whether it is under the fog of war and various other factors. It is almost always true, but is often turned off for death models. In this case, the visibility is set once during the actor's creation and then never updated. (This is so the player can see the full death animation, even when the target disappears under the fog of war.)
EDSTR_ENUMHINT_e_actorModelFlagUseConstSyncSeed=Causes the model to use a synchronous random number seed from the game when choosing its model variation. This seed is always the same for the lifetime of the actor's parent scope and cannot be accidentally made asynchronous. Enables different Dark Templar variations to always appear the same on all machines participating in the game.
EDSTR_ENUMHINT_e_actorModelFlagUseSyncSeed=Causes the model to use a synchronous random number seed from the game when choosing its model variation. After it is used, the seed is modified and written back to the parent scope, such that if all model-setting events are synchronous, then all model variations on that particular model actor will be the same on all machines. Used to ensure that some unit-related models like building attachments look the same on all machines.
EDSTR_ENUMHINT_e_actorModelFlagWireframeRender=Causes the model to render as a wireframe, rather than with textures. Good for displaying debugging actors, such as those that represent a camera's bearings.
EDSTR_ENUMHINT_e_actorModelFlagOutlineEmitter=The model generates an outline when obscured by a model with the OutlineOccluder flag set. In other words, the player can see an outline of the model through other models, almost as if he had X-ray vision.
EDSTR_ENUMHINT_e_actorModelFlagOutlineOccluder=The model causes models marked by the OutlineEmitter flag to generate an outline that appears through the OutlineOccluder. Useful for large buildings and other models behind which players might lose units.
EDSTR_ENUMHINT_e_actorModelFlagAnimationsDontResetOnUnhide=The animations on this actor model will not be reset when another animation that is hiding them is removed.
EDSTR_ENUMHINT_e_actorModelFlagUseSprayForDecal=Uses the player's selected spray reward as the decal texture.
EDSTR_ENUMHINT_e_actorModelAspectPersonAny=Any kind of participant or observer. Prevents a model aspect from being filtered on this field at all.
EDSTR_ENUMHINT_e_actorModelAspectPersonParticipant=A player actively playing the game.
EDSTR_ENUMHINT_e_actorModelAspectPersonSpectator=A player that is observing the game in the standard way.
EDSTR_ENUMHINT_e_actorModelAspectPersonReferee=A player that is observing the game with special referee

status.
EDSTR_ENUMHINT_e_actorModelAspectPersonObserver=Any observer, whether spectator or referee.
EDSTR_ENUMHINT_e_actorModelAspectObservingPoVAny=Does not matter from which perspective the observer is viewing. Prevents a model aspect from being filtered on this field at all.
EDSTR_ENUMHINT_e_actorModelAspectObservingPoVEveryone=Viewing from the everyone perspective, as implemented in SC2.
EDSTR_ENUMHINT_e_actorModelAspectObservingPoVDefaultObservedPlayerId=Viewing from the special observer playerId, as implemented in Heroes of the Storm.
EDSTR_ENUMHINT_e_actorModelAspectObservingPoVPlayerId=Viewing from a specific playerId.
EDSTR_ENUMHINT_e_actorModelAspectObservingPoVTeamId=Viewing from a player belonging to a specific teamId.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsAny=Irrelevant who owns the target model. Prevents a model aspect from being filtered on this field at all.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsSelf=The model is owned by the viewing player.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsAlly=The model is an ally of the viewing player, but not the viewing player himself.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsEnemy=The model is an enemy of the viewing player.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsNeutral=The model is neutral with respect to the viewing player.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsFriendly=The model is either owned by or is an ally of the viewing player. Logically equivalent to a combination of Self and Ally.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsNotSelf=The model is not owned by the viewing player, and can be ally, enemy or neutral.
EDSTR_ENUMHINT_e_actorModelAspectRegardsAsTeammate=The model is owned by a teammate of the viewing player. Models owned by the viewing player are not considered teammates.
EDSTR_ENUMHINT_e_actorModelAspectDuringAny=Any kind of game or replay. Prevents a model aspect from being filtered on this field at all.
EDSTR_ENUMHINT_e_actorModelAspectDuringLiveGame=Viewing a live game, including one that may have been hijacked from a replay.
EDSTR_ENUMHINT_e_actorModelAspectDuringLiveGameHijacked=Viewing a live game that was hijacked from a replay.
EDSTR_ENUMHINT_e_actorModelAspectDuringLiveGameNotHijacked=Viewing a live game that was not hijacked from a replay.
EDSTR_ENUMHINT_e_actorModelAspectDuringReplay=Viewing a replay.
EDSTR_ENUMHINT_e_actorModelAspectObservedPlayerTypeAny=Any kind of player, whether human or computer. Prevents a model aspect from being filtered on this field at all.
EDSTR_ENUMHINT_e_actorModelAspectObservedPlayerTypeHuman=The observed player is a human.
EDSTR_ENUMHINT_e_actorModelAspectObservedPlayerTypeHumanParticipant=The observed player is a human that is playing the game (or was, in the case of a replay).
EDSTR_ENUMHINT_e_actorModelAspectObservedPlayerTypeHumanObserver=The observed player is a human that is observing the game (or was, in the case of a replay).
EDSTR_ENUMHINT_e_actorModelAspectObservedPlayerTypeComputer=The observed player is an AI. (This does not include human players that are having a computer stand in for them temporarily because they have disconnected.)
EDSTR_ENUMHINT_e_actorModelAspectModelOwnerTypeAny=Any kind of player, whether human or computer. Prevents a model aspect from being filtered on this field at all.
EDSTR_ENUMHINT_e_actorModelAspectModelOwnerTypeHuman=The owning player is a human.
EDSTR_ENUMHINT_e_actorModelAspectModelOwnerTypeComputer=The owning player is an AI. (This does not include human players that are having a computer stand in for them temporarily because they have disconnected.)
EDSTR_ENUMHINT_e_actorModelAspectTestNone=The default. No additional tests filter whether the aspect is chose.
EDSTR_ENUMHINT_e_actorModelAspectTestMessage=The aspect sends a ModelAspectTest message with an optional subName. If the client data responds to the message with a ModelAspectConfirm message, then that aspect passes the message test and may be selected if it passes its remaining criteria. This test enables arbitrary sets of terms to help select a given model aspect.
EDSTR_ENUMHINT_e_labelExpressionOpRequireAll=Returns true if the player has all of the labels in the comma-

separated list.
EDSTR_ENUMHINT_e_labelExpressionOpRequireAny=Returns true if the player has any of the labels in the comma-separated list.
EDSTR_ENUMHINT_e_labelExpressionOpExcludeHasAll=Returns true if the player doesn't have all of the labels in the comma-separated list.
EDSTR_ENUMHINT_e_labelExpressionOpExcludeHasAny=Returns true if the player has none of the labels in the comma-separated list.
EDSTR_ENUMHINT_e_actorModelMaterialTypeGlaze=The new material is drawn on top of the material it affects, without hiding the underlying material. A frost material applied to a leaf would use this type of material operation.
EDSTR_ENUMHINT_e_actorModelMaterialTypeReplacement=The material visually appears to replace the material it affects, causing the underlying material to disappear entirely while the new material is applied. A petrification material might use this type of operation, since the stony surface would completely replace the material of the character that had been turned to stone.
EDSTR_ENUMHINT_e_actorSelectionFlagIsSyncSelection=The selection responds to synchronous system messages to determine its color and visibility. The player's unit selection is an example of a synchronous selection, whereas mouseover unit highlighting is an example of an asynchronous selection.
EDSTR_ENUMHINT_e_actorSelectionFlagIsFlash=The selection responds to flash messages. These come from sources like alerts, triggers, and transmissions.
EDSTR_ENUMHINT_e_actorSelectionFlagUseCheapQuadSplat=The selection draws itself onto a simple quad, rather than trying to project itself onto more complex geometry. This is often much faster, but it can cause the splat to intersect terrain in a visually unappealing way.
EDSTR_ENUMHINT_e_actorSelectionFlagUseCrescent=The selection draws itself as a crescent, which reduces visual clutter.
EDSTR_ENUMHINT_e_actorSiteFlagRequiresManualKill=Site does not automatically destroy itself after two updates if no child actors or siteOps are hosting from it. Sometimes it is useful to keep a site around as actors come and go, but still need to use it.
EDSTR_ENUMHINT_e_actorUnitFlagAutomateWalkDirections=Tells the unit to automatically find and play directional Walk anims like Walk, A when present in the model.
EDSTR_ENUMHINT_e_actorUnitFlagShowBlobShadow=Marks a unit actor as displaying a blob shadow on the low shadow graphics setting. This feature is implemented in data, not code.
EDSTR_ENUMHINT_e_actorUnitFlagStandAnimBirthVariation=Causes the unit to start at a random offset into its stand animation when it is first created. With this flag, units that are created from a construction building do not all appear in the exact same pose when they first appear.
EDSTR_ENUMHINT_e_actorUnitFlagSuppressStandIntro=Tells the unit to not play Stand, Start animations when it stops moving, even if it has them in its model.
EDSTR_ENUMHINT_e_actorUnitFlagSuppressWalkIntro=Tells the unit to not play Walk, Start animations when it starts moving, even if it has them in its model.
EDSTR_ENUMHINT_e_actorUnitFlagSuppressWalkOutro=Tells the unit to not play Walk, End animations when it starts moving, even if it has them in its model.
EDSTR_ENUMHINT_e_actorUnitFlagSuppressDefaultStatusBar=Prevents health bars from appearing on the unit. Useful for large boss creatures, where a standard health bar would appear tiny in proportion.
EDSTR_ENUMHINT_e_actorRequestCreateSharingNone=Always created by requests.
EDSTR_ENUMHINT_e_actorRequestCreateSharingAlways=Created if it doesn't exist, otherwise always shared.
EDSTR_ENUMHINT_e_actorRequestCreateSharingPerEffectTree=Functions like the Always setting, but only shared by effects in the same effect tree.
EDSTR_ENUMHINT_e_actorSplatHeightAutomatic=Acts as the MinimumTerrain option for splats that are marked as AOELayer, and acts as the Normal option for splats that are not.
EDSTR_ENUMHINT_e_actorSplatHeightNormal=Unrestricted; the splat can draw itself at whatever height it finds terrain to map itself to.
EDSTR_ENUMHINT_e_actorSplatHeightMinimumActor=At its lowest, the splat draws itself at the z value of its owning actor.
EDSTR_ENUMHINT_e_actorSplatHeightMinimumTerrain=At its lowest, the splat draws itself at the z value of the map's base height.
EDSTR_ENUMHINT_e_actorHostedPropInheritNone=Do not inherit any hosted props ever.

EDSTR_ENUMHINT_e_actorHostedPropInheritOneShot=Inherit hosted props once during initialization, but never again after that.
EDSTR_ENUMHINT_e_actorHostedPropInheritContinuous=Inherit hosted props on an ongoing basis, whenever the actor has a host.
EDSTR_ENUMHINT_e_actorSiteOpBillboardUnobstructable=Positions actors hosted off the billboard site right next to the camera, but scales them to appear to be at the site's host's location. Makes the Ghost's Nuke laser spot show up, regardless of what is covering it.
EDSTR_ENUMHINT_e_actorSiteOpBillboardUseTowardsCameraDistance=Positions actors hosted off the billboard site a specified distance along the ray to the camera. Can be used to make flying units appear to be on top of their ground position (so enemies may target them more easily), while also appearing to be flying.
EDSTR_ENUMHINT_e_actorSiteOrbiterTypeRelative=Orbiting is relative to an input rotation. For CActorSiteOrbiter, this is typically the rotation of the parent actor. For CActorSiteOpOrbiter it is the rotation passed into the site op.
EDSTR_ENUMHINT_e_actorSiteOrbiterTypeAbsolute=Orbiting is relative to the default rotation in world space, regardless of incoming rotations.
EDSTR_ENUMHINT_e_actorTiltTypeMover=Gets velocity from the CUnit's mover associated with the scope that contains the CActorSiteOpTilter.
EDSTR_ENUMHINT_e_actorTiltTypeNoMover=Gets velocity from the CActorSiteOpTilter's parent actor.
EDSTR_ENUMHINT_e_actorTiltTypeAny=Attempts to use the Mover option, but if there is no mover, uses the NoMover option.
EDSTR_ENUMHINT_e_actorSiteOpActionCenter=Selects the center of the corresponding CActorAction, which is typically the center of the game Effect that created it.
EDSTR_ENUMHINT_e_actorSiteOpActionImpact=Selects the synchronous game world impact point of the corresponding CActorAction.
EDSTR_ENUMHINT_e_actorSiteOpActionLaunch=Selects the synchronous game world launch point of the corresponding CActorAction.
EDSTR_ENUMHINT_e_actorSiteOpAttachSourceAutomatic=Act like the Host value in this enum if possible, otherwise act like the Chain value.
EDSTR_ENUMHINT_e_actorSiteOpAttachSourceChain=Look for the attach point inside a model actor referred to by a site op earlier in the site op chain containing the attach site op. Starts at the beginning of the site op chain and asks each site op if it has a host variable pointing to a model actor. The first site op with a valid model actor wins.
EDSTR_ENUMHINT_e_actorSiteOpAttachSourceHost=Look for the attach point inside the model actor that the site op's corresponding host reference is referring to.
EDSTR_ENUMHINT_e_actorSiteOpBasicWorldPosition=Outputs an absolute world position specified in data, via the WorldPosition field.
EDSTR_ENUMHINT_e_actorSiteOpBasicActorBearings=Outputs the bearings of the parent actor. This can be used by hosted attaches to offset attach point positions from the origin of the model.
EDSTR_ENUMHINT_e_actorSiteOpBasicActorPosition=Outputs only the position of the parent actor.
EDSTR_ENUMHINT_e_actorSiteOpBasicCreatorBearings=Outputs the bearings of the actor that created the site op's parent actor.
EDSTR_ENUMHINT_e_actorSiteOpBasicCreatorPosition=Outputs only the position of the actor that created the site op's parent actor.
EDSTR_ENUMHINT_e_actorSiteOpBasicInitialBearings=Outputs the bearings of the parent actor at the site op's initialization time. The bearings output by the site op do not change even if the parent actor's bearings change later.
EDSTR_ENUMHINT_e_actorSiteOpBasicScopeBearings=Outputs the bearings of the scope containing the site op's parent actor.
EDSTR_ENUMHINT_e_actorSiteOpBasicScopePosition=Outputs only the position of the scope containing the site op's parent actor.
EDSTR_ENUMHINT_e_actorSiteOpDeltaSumFlagZOffsetOnly=Only measure the local space z offset between the specified attach points. When not on, the delta sum site op sums the local x, y, and z coordinates of each offset.
EDSTR_ENUMHINT_e_actorSiteOpDeltaSumFlagRespectIncomingRotation=Applies the delta sum to the rotation entering the siteOp rather than the rotation of the owning actor.
EDSTR_ENUMHINT_e_actorSiteOpPhysicsImpactNormal=Output the position of the physics impact.
EDSTR_ENUMHINT_e_actorSiteOpTerrainAndWaterFlagFollowWaves=Outputs a position that accounts for wave and trough height. Enable actors to appear to float realistically on the surface of the water. If this is not set, the output

position merely captures the height of the water plane at the position sent into the site op.
EDSTR_ENUMHINT_e_actorSiteOpTerrainAndWaterFlagPassThroughIfNoWater=Causes the site op to act as if it was not in the site op chain if there is no water at the position sent into the site op. Can be used to have blood appear on the ground if there is no water, but appear on the surface of the water if the point where the blood appears is underwater.
EDSTR_ENUMHINT_e_actorSiteOpOrientAttachPointToPosition=Output a position that causes the site op's parent actor's specified attach point to be at the position sent into the site op. Can be used to connect a child model to a parent model at a child attach point that is other than the Origin. With this site op, client data could attach a weapon handle attach point (rather than the weapon model's Origin attach point) to a parent model's Hand attach point.
EDSTR_ENUMHINT_e_actorSiteOpOrientAttachPointToRotation=Output a rotation that causes the site op's parent actor's specified attach point to have the same rotation as the rotation sent into the site op. (This does not just rotate the attach point itself but instead orients the entire model around that attach point.)
EDSTR_ENUMHINT_e_actorSiteOpOrientAttachPointToBearings=Combines the functionality of the Position and Rotation options.
EDSTR_ENUMHINT_e_actorSiteOpRotatorAccumulateAndSetRotation=For cases where the rotation is the same every time it is passed to the site op. We need to appear to be continuously rotating and advancing the rotation each frame, so we must accumulate previous rotations, because the rotation coming into the site op is always the same. For instance, when we are rotating a simple standalone model in its own scope and the site op gets its input bearings from that scope.
EDSTR_ENUMHINT_e_actorSiteOpRotatorAddRotationChange=For cases where the rotation is different every time it is passed to the site op, because of the site op itself. For instance, when a CActorSiteOpRotator rotates an actor and the CActorSiteOpRotator is the first site op in the site op chain. In cases like this, the accumulated rotation is stored externally to the site op, rather that in the site op itself, as with the AccumulateAndSetRotation value.
EDSTR_ENUMHINT_e_actorSiteOpShadowFlagNoLowerThanBaseHeight=Ensures the site op never outputs positions that are lower the map's base height.
EDSTR_ENUMHINT_e_actorSiteOpShadowFlagNoLowerThanWater=Ensures the site op never outputs positions that are lower than the water at the input position.
EDSTR_ENUMHINT_e_actorSiteOpTetherFlagDisabled=Leaves coordinate unmodified.
EDSTR_ENUMHINT_e_actorSiteOpTetherFlagMinMax=Clamps coordinate to both the minimum and maximum values.
EDSTR_ENUMHINT_e_actorSiteOpTetherFlagMin=Limits lower range of coordinate only.
EDSTR_ENUMHINT_e_actorSiteOpTetherFlagMax=Limits upper range of coordinate only.
EDSTR_ENUMHINT_e_actorSiteOpTipabilityFlagNoLowerThanBaseHeight=Ensures the site op never outputs positions that are lower the map's base height. Good for cursors that have to respect tipability, but should not disappear when the mouse moves over canyons.
EDSTR_ENUMHINT_e_actorSiteOpTipabilityFlagNoLowerThanScope=Ensures the site op never outputs positions that are lower than z value of their containing scope. Good for cursors that have to respect tipability, but need to stay aligned with whatever other visuals are in the cursor's scope.
EDSTR_ENUMHINT_e_actorRadialDistributionUniform=Positions are equally distributed across the radius.
EDSTR_ENUMHINT_e_actorRadialDistributionExponential=Positions follow an exponential distribution away from the center of the circle. Exponential distributions often have very high values in the center and very low values at the radius.
EDSTR_ENUMHINT_e_actorRadialDistributionGaussian=Positions follow a Gaussian distribution (i.e. a bell curve) away from the center of the circle. Gaussian distributions often produce similar results to exponential distributions but with slightly different control parameters.
EDSTR_ENUMHINT_e_actorCrossbarDistributionUniform=Equally distributes positions along the crossbar.
EDSTR_ENUMHINT_e_actorCrossbarDistributionGaussian=Generates positions with a Gaussian distribution. Tends to distribute positions towards the center of the crossbar.
EDSTR_ENUMHINT_e_actorCrossbarDistributionGaussianInverse=Generates positions with a Gaussian distribution, but flips the resulting distribution curve. Tends to distribute positions towards the ends of the crossbar.
EDSTR_ENUMHINT_e_actorShieldFlashTypeFacer=The shield faces directly back towards the incoming attack vector. The is the most realistic option.
EDSTR_ENUMHINT_e_actorShieldFlashTypeFull=Creates a full shield flash around the entire target. Useful for when the attack does not have an obvious incoming direction, like when a Protoss unit is hit by an AoE spell such as Psi Storm. Also useful if directional shield impacts would be confusing, such as when a unit attacks a Colossus from inside

its shields.
EDSTR_ENUMHINT_e_actorShieldFlashTypeHeader=The shield impact faces horizontally toward the incoming attack. Functions like the Facer option, but causes the shield flash to always be oriented facing parallel to the XY plane. Useful when the Facer option is visually confusing, such as when a small unit like a Zergling attacks a large building from a very low position. This causes a Facer impact to be oriented in a way that is partially obscured by terrain and not easy to visually parse to begin with.
EDSTR_ENUMHINT_e_actorShieldFlashTypeNone=Display no shield impact. Useful for game actions that pass through them, like mind control.
EDSTR_ENUMHINT_e_actorTransferFlagAnimator=Copy as much of the source actor's animation stack as possible, given that the model of the new actor may not have all of the same animations available.
EDSTR_ENUMHINT_e_actorTransferFlagAnimProps=Copy persistent anim props that have been assigned via the AnimGroupApply message.
EDSTR_ENUMHINT_e_actorTransferFlagFOWColor=Copy the source actor's fog of war tinting color.
EDSTR_ENUMHINT_e_actorTransferFlagFOWShader=Copy from the source actor whether to use a shader to render fog of war.
EDSTR_ENUMHINT_e_actorTransferFlagModel=Copy the model contained by the source actor.
EDSTR_ENUMHINT_e_actorTransferFlagModelMaterials=Copy the model materials applied to the source actor.
EDSTR_ENUMHINT_e_actorTransferFlagPosition=Copy the source actor's position.
EDSTR_ENUMHINT_e_actorTransferFlagRotation=Copy the source actor's rotation.
EDSTR_ENUMHINT_e_actorTransferFlagTextures=Copy the source actor's dynamic textures, to the extent that it is possible. For example, this copies the source actor's decal textures, provided it is possible to display them on the newly created actor.
EDSTR_ENUMHINT_e_actorTransferFlagStatus=Copy the source actor's status variables, which can be created and manipulated with messages like StatusSet, StatusIncrement and StatusDecrement.
EDSTR_ENUMHINT_e_actorTextAlignmentLeft=Left justification. Considered top for vertical alignment.
EDSTR_ENUMHINT_e_actorTextAlignmentTop=Aligns text to the top of the containing region. Considered left for horizontal alignment.
EDSTR_ENUMHINT_e_actorTextAlignmentCenter=Centers text.
EDSTR_ENUMHINT_e_actorTextAlignmentRight=Right justification. Considered bottom for vertical alignment.
EDSTR_ENUMHINT_e_actorTextAlignmentBottom=Aligns text to the bottom of the containing region. Considered right for horizontal alignment.
EDSTR_ENUMHINT_e_actorTextOptionConformToTerrain=If on, draws the text as if it were texture mapped onto the terrain itself. If off, draws text just above the terrain and billboards it towards the camera.
EDSTR_ENUMHINT_e_actorTextOptionUseTerrainHeight=Unsupported.
EDSTR_ENUMHINT_e_actorTextOptionUseWalkableHeight=Unsupported.
EDSTR_ENUMHINT_e_actorActionFlagAlignedImpactsAndDamages=Used for beam attacks to ensure that subsequent damage pulses produce FX at the beam's impact position, which is where the beam first struck the target.
EDSTR_ENUMHINT_e_actorActionFlagIgnoresShields=Causes attacks to bypass shields entirely. The Viper's Consume beam is an example, since it drains health from the core building, not it's shields.
EDSTR_ENUMHINT_e_actorActionFlagImpactForceSite=Forces the CActorAction to create a CActorSite at the impact position. This enables multiple FX to easily share a single bearings for as long as any of the FX exist.
EDSTR_ENUMHINT_e_actorActionFlagImpactMonitorsTeleports=Forces the CActorAction to remain alive at least as long as any impact-related beams or impact model FX that it has spawned remain alive. This enables it to route ActionTargetTeleport messages to these actors, so they can be configured to kill themselves when a target has teleported far out of range.
EDSTR_ENUMHINT_e_actorActionFlagImpactSuppressUnitSound=Prevents the creation of the additional generic impact sound that is configured on the target CActorUnit.
EDSTR_ENUMHINT_e_actorActionFlagLaunchForceSite=Forces the CActorAction to create a CActorSite at the launch position. This enables multiple FX to easily share a single bearings for as long as any of the FX exist.
EDSTR_ENUMHINT_e_actorActionFlagVictimRevealsAttacker=Marks the attack as temporarily revealing the attacker when a player can see the victim of the attack. Useful for tentacles. If a Viper that is under the fog of war Abducts a victim unit the player can see, it makes more sense visually for the victim unit's owner to be able to see the owner of the tentacle while this is happening.
EDSTR_ENUMHINT_e_actorRangeFlagGameWorld=Draws the range indicator in the game world.

EDSTR_ENUMHINT_e_actorRangeFlagMinimap=Draws the range indicator on the minimap.
EDSTR_ENUMHINT_e_actorRegionFlagUseEffectRange=When a CActorRegion is being used to drive the highlighting of units affected by AoE cursors, this automatically scales the region to match the range of the effect that the targeting ability would create.
EDSTR_ENUMHINT_e_actorModelMaterialFlagForceAnimateWhileFogged=Forces the material to animate even when it is on a CActorSnapshot or a CActorDoodad that is under the fog of war. This can be helpful if a highly active material (such as fire) would look visually jarring if displayed in a frozen state.
EDSTR_ENUMHINT_e_actorQuadFlagAlignVerticallyOnOrigin=Causes the quad to get its height from the origin of the quad, rather than the terrain height at the quad's center.
EDSTR_ENUMHINT_e_actorQuadFlagPreserveProportionsDuringWidthScaling=Preserves the initial proportions of the target CActorQuad's launch, center and impact actors when the CActorQuad has been set to respect width scaling, and it has an X scale that is not 1.0.
EDSTR_ENUMHINT_e_actorQuadFlagRespectWidthScaling=By default, CActorQuad ignores X and Z scaling, so user data can easily change its size with a simple uniform scale. When this option is set, X scaling affects the width of the target CActorQuad as is common with other actors.
EDSTR_ENUMHINT_e_actorQuadFlagForwardAnimMsgsLaunchActor=If true, any animation messages the quad actor receives will be passed onto the launch actor.
EDSTR_ENUMHINT_e_actorQuadFlagForwardAnimMsgsCenterActor=If true, any animation messages the quad actor receives will be passed onto the center actor.
EDSTR_ENUMHINT_e_actorQuadFlagForwardAnimMsgsImpactActor=If true, any animation messages the quad actor receives will be passed onto the impact actor.
EDSTR_ENUMHINT_e_actorQuadFlagForwardAnimMsgsDecorationActor=If true, any animation messages the quad actor receives will be passed onto the decoration actor.
EDSTR_ENUMHINT_e_actorQuadDecorationFlagUseLaunchActorScale=If true, the decoration will use the scale of the launch actor.
EDSTR_ENUMHINT_e_actorQuadDecorationFunctionLinear=Travels linearly from point A to point B
EDSTR_ENUMHINT_e_actorQuadDecorationFunctionBias=Uses the Bias function to travel from point A to point B. A bias parameter value of 0.5 is linear. A value greater than 0.5 will start fast and end slow. A value less than 0.5 will start slow and end fast.
EDSTR_ENUMHINT_e_actorQuadDecorationFunctionGain=Uses the Gain function to travel from point A to point B. A gain parameter value of 0.5 is linear. A value greater than 0.5 will be fast on the ends and slow in the middle. A value less than 0.5 will be slow on the ends and fast in the middle.
EDSTR_ENUMHINT_e_actorHeightSourceAsyncFast=Raycasts against asynchronous terrain. Like AsyncPrecise, but returns cliff level 1 height over cliff level 0.
EDSTR_ENUMHINT_e_actorHeightSourceAsyncPrecise=Raycasts against asynchronous terrain. Uses interpolation to smooth results between points in the height map.
EDSTR_ENUMHINT_e_actorHeightSourceSyncAir=Raycasts against the synchronous flier height map. This height map has significant smoothing applied to it, to make air units gently rise over cliffs.
EDSTR_ENUMHINT_e_actorHeightSourceSyncGround=Raycasts against the synchronous ground unit height map.
EDSTR_ENUMHINT_e_actorHeightTestAverage=Use the average of all the heights in the sample set.
EDSTR_ENUMHINT_e_actorHeightTestHighest=Use the highest height in the sample set.
EDSTR_ENUMHINT_e_actorHeightTestLowest=Use the lowest height in the sample set.
EDSTR_ENUMHINT_e_actorIncomingDirect=The attack the site op is gathering data from is a direct ranged attack that happens instantly. Examples: Marine and Viking ground attacks.
EDSTR_ENUMHINT_e_actorIncomingMissile=The attack the site op is gathering data from is a ranged attack with a missile. Examples: Stalker and Hydralisk attacks.
EDSTR_ENUMHINT_e_actorSoundFlagUpdateVisibility=Enables the actor to update its visibility in response to whether or not it is under the fog of war, and various other factors. It is almost always true, but is often turned off for death sounds. In this case, the visibility is set once during the actor's creation and then never updated. (This is so the player can hear the full death sound, even when the target disappears under the fog of war.)
EDSTR_ENUMHINT_e_actorSoundPlayModeAll=Play all the sound layers at once
EDSTR_ENUMHINT_e_actorSoundPlayModeSequence=Play the layers one after another in order
EDSTR_ENUMHINT_e_actorSoundPlayModeRandom=Play the layers one after another in a randomized order
EDSTR_ENUMHINT_e_actorSoundValueSourceCombine=The values from the actor and the sound will be combined.

EDSTR_ENUMHINT_e_actorSoundValueSourceActor=The value from the actor will be used.
EDSTR_ENUMHINT_e_actorSoundValueSourceSound=The value from the sound will be used.
EDSTR_ENUMHINT_e_actorTerrainDeformerFlagDestroysFoliage=When the CActorTerrainDeformer modifies terrain, it destroys foliage within its area of effect. Foliage is small ambient models like bits of grass that are auto-generated on the terrain to make the map feel more immersive.
EDSTR_ENUMHINT_e_actorTerrainDeformerFlagUseUnitFootprint=If this option is set, and the CActorTerrainDeformer belongs to a unit scope, it acquires the game synchronous footprint of the unit and uses that as its area of effect.
EDSTR_ENUMHINT_e_actorTerrainDeformerFlagRestoreOnHidden=Restores terrain to its original state when the CActorTerrainDeformer is hidden.
EDSTR_ENUMHINT_e_actorTerrainDeformerFlagRestoreOnMovement=Removes and then reapplies terrain deformation as the actor moves. With this flag off, moving the CActorTerrainDeformer will appear to paint terrain deformation on the map. When the flag is on, a single terrain deformation appears to move with the CActorTerrainDeformer.
EDSTR_ENUMHINT_e_actorTerrainDeformerFlagRestoreOnDestroy=Restores terrain to its original state when the CActorTerrainDeformer dies. Enables spells to crater terrain temporarily, so that the map developer does not need to worry that repeated spell use will damage the playability and clarity of the map.
EDSTR_ENUMHINT_e_unitStatusGroupOwner=When off, prevents the health bar from showing up when the 'Show Your Life Bars' hotkey command is active.
EDSTR_ENUMHINT_e_unitStatusGroupAllied=When off, prevents the health bar from showing up when the 'Show Allied Life Bars' hotkey command is active.
EDSTR_ENUMHINT_e_unitStatusGroupEnemy=When off, prevents the health bar from showing up when the 'Show Enemy Life Bars' hotkey command is active.
EDSTR_ENUMHINT_e_unitStatusGroupAll=When off, prevents the health bar from showing up when the 'Show All Life Bars' hotkey command is active.
EDSTR_ENUMHINT_e_actorCloakEffectNone=The unit is not cloaked.
EDSTR_ENUMHINT_e_actorCloakEffectAlly=The unit is cloaked and being observed by an ally.
EDSTR_ENUMHINT_e_actorCloakEffectDetected=The unit is cloaked and being observed by an enemy that has detected it.
EDSTR_ENUMHINT_e_actorCloakEffectEnemy=The unit is cloaked and being observed by an enemy that has no detector within detection range.
EDSTR_ENUMHINT_e_actorProximityCenterActor=The model stores itself in the proximity query system using its parent actor's position.
EDSTR_ENUMHINT_e_actorProximityCenterModelBounds=The model stores itself in the proximity query system using the center of its model bounds. Can produce more accurate query results when a model is very large and its model bounds center is quite distant from its actor's center.
EDSTR_ENUMHINT_e_actorPhysicsDataFlagExpectsOtherForces=Signifies that this attack triggers physics deaths but relies on other forces to actually move the ragdolls it might create.
EDSTR_ENUMHINT_e_actorForceFlagFallOff=When on, decays the strength of a force rapidly as the distance increases between the center of the force and the point of its application. When off, force strength is uniform throughout its entire volume of effect.
EDSTR_ENUMHINT_e_actorForceFlagStacks=When on, CActorForces with the same catalog link name have an cumulative effect on a target physics body when applied simultaneously (as in the real world). When off, only the CActorForce (with a given catalog id) with the strongest affect on a given physics body applies to that body at any one time. Turning this flag on with forces that are likely to be applied in large numbers simultaneously at the same point can produce unintended and visually jarring (though sometimes hilarious) results.
EDSTR_ENUMHINT_e_actorForceFieldDirectional=Force travels from one side of the CActorForce to the other, generally in the direction of the CActorForce's forward vector (though this direction can be configured on some actors).
EDSTR_ENUMHINT_e_actorForceFieldRadial=Force travels from the center position of the CActorForce outwards. The center is typically the geometric center of the volume, but it is at the pointy end of the CActorForceConeRoundedEnd, and at the center of the cap next to the origin for CActorForceCylinder.
EDSTR_ENUMHINT_e_actorForceDirectionUp=Force points in the direction of the parent actor's up vector.
EDSTR_ENUMHINT_e_actorForceDirectionForward=Force points in the direction of the parent actor's forward vector.
EDSTR_ENUMHINT_e_actorForceOriginAtBottom=The bottom of the force box is centered at the actor's position (i.e.

its origin).
EDSTR_ENUMHINT_e_actorForceOriginAtCenter=The center of the force box is at the actor's position.
EDSTR_ENUMHINT_e_actorCombatRevealDurationOneShot=A single attack reveals the attacker for a short period of time.
EDSTR_ENUMHINT_e_actorCombatRevealDurationByVictim=The attacker is combat revealed for as long as a player can see the victim of the attack. This makes it possible to see an Infestor that Neural Parasites a player's unit from under the fog of war, so that the Infestor's tentacle does not appear to hang in space, disconnected from its body.
EDSTR_ENUMHINT_e_serpentBasic=The serpent behaves like a string dragged from one end.
EDSTR_ENUMHINT_e_serpentSwimming=The serpent behaves like a snake swimming in water; it undulates side to side in a sinusoidal pattern as it moves forward.
EDSTR_ENUMHINT_e_actorSerpentFlagUseAttachDistances=When animating attach points in a single model, this enables the data to specify how far the different attach points are supposed to be from one another during serpent motion by getting the attach point distances from each other when the model is first initialized. When this flag is off, attach point radii must be specified in data, which can be time-consuming and error prone if there are numerous attach points being animated.
EDSTR_ENUMHINT_e_actorTerrainSquibVisualFlagOneShot=When on, terrain squibs are separate models that are created periodically as the unit moves. When off, a single model plays continuously for as long as the unit moves across a particular terrain type. OneShot terran squibs can look better than continuous ones, but they are typically far more costly from a performance perspective.
EDSTR_ENUMHINT_e_actorOverkillTestPercentageOfMaxLife=Measures overkill as a percentage of maximum health. Example: A unit being killed by an attack that inflicts damage beyond what was necessary to kill the target by at least 20% of the target unit's maximum health.
EDSTR_ENUMHINT_e_actorOverkillTestPercentageOfMaxLifePlusShields=Like the PercentageOfMaxLife option, but measures overkill as a percentage of maximum health plus maximum shields.
EDSTR_ENUMHINT_e_actorOverkillTestFixed=A killing blow counts as an overkill if it inflicts X points of damage or more past what was necessary to kill the unit.
EDSTR_ENUMHINT_e_minimapShapeQuad=Draws as a square.
EDSTR_ENUMHINT_e_minimapShapeCircle=Draws as a circle.
EDSTR_ENUMHINT_e_actorAnimTransitionTypeClearToPlay=Applies a blend time when a specific animation A is finishing and being replaced by specific animation B. Specifies the blend out time for animation A and the blend in time for animation B.
EDSTR_ENUMHINT_e_actorAnimTransitionTypeClearWhilePlaying=Applies when animation A is cleared while animation B is playing. Specifies the blend out time for animation A.
EDSTR_ENUMHINT_e_actorAnimTransitionTypePlayWhilePlaying=Applies when animation A is played while animation B is playing. Specifies the blend in time for animation A.
EDSTR_ENUMHINT_e_actorAnimPropMatchTypeContains=Returns true if a set of anim props contains all the anim props being tested.
EDSTR_ENUMHINT_e_actorAnimPropMatchTypeFullMatch=Returns true if a set of anim props has exactly the anim props being tested, no more and no less, excluding variations.
EDSTR_ENTRYHINT_SActorHostedDelta=Specifies each subject actor that SiteOpDeltaSum checks to accumulate the total offset to apply to its parent actor.
EDSTR_ENTRYHINT_SActorModelAspect=A rule that may cause a target CActorModel to display a substitute model to the viewing player, rather than the core model specified in the CActorModel's data or a model set dynamically by the ModelSwap message.
EDSTR_ENTRYHINT_SActorModelAspectSet=A set of model aspects that activate when a particular model (or models) is set on the containing CActorModel. An aspect causes the CActorModel to appear as a different model in a specified situation.
EDSTR_ENTRYHINT_SActorDeathBodySquib=A body squib is a model like a blood burst particle system that typically gets created and attached to some number of attach points on a ragdoll death model. This enables models (such as ragdolls) that cannot as easily have hand animations built into them to still spray blood, catch on fire, and be burned by acid. Technically, body squibs can be used with any custom death model, not just ragdolls. Body squibs can also have an associated sound, so that fire can crackle and acid can hiss.
EDSTR_ENTRYHINT_CActorCutscene=An actor for making it easier to integrate cutscenes with the game world from game data.

EDSTR_ENTRYHINT_CActorLookAt=Controls how a source model moves to point at specified target. The source model can have one or more turret groups that move together with different blend weights to face the target in a natural way. The system supports conventional tank turrets along with more sophisticated spine, head and eye configurations.
EDSTR_ENTRYHINT_CActorSiteOpDeltaSum=A siteOp that enables actors to position themselves by adding the result of the sum of the differences of attach point positions in other actors. Typically used to position health bars on units composed of several models.
EDSTR_ENTRYHINT_CActorSiteOpCursor=A siteOp that rotates the host to face the terrain point under the mouse cursor.
EDSTR_ENTRYHINT_CActorSiteOpYawLimiter=A siteOp to restrict the yaw rotation angle with respect to host.
EDSTR_FIELDHINT_SLookAtTypeInfo_Group=The name used to identify the turret group in data.
EDSTR_FIELDHINT_SLookAtTypeInfo_Weight=The maximum amount of blend realized by the turret group.
EDSTR_FIELDHINT_SLookAtTypeInfo_Time=The time it takes the LookAtStart or LookAtStop to complete, regardless of the size of the angle the operation must cover.
EDSTR_FIELDHINT_SLookAtTypeInfo_Rate=Tells the turret group to rotate at the specified angular velocity, measured in degrees per second.
EDSTR_FIELDHINT_SLookAtType_Id=The label used to identify this particular look at type. Used by various fields and messages to control the behavior of a CActorLookAt.
EDSTR_FIELDHINT_SLookAtType_Name=A localized string describing the LookAtType in the editor UI.
EDSTR_FIELDHINT_SLookAtType_Start=Controls how the turret blends and moves when it is commanded to look at a target.
EDSTR_FIELDHINT_SLookAtType_Stop=Controls how the turret blends and moves when it is commanded to return to its default orientation.
EDSTR_FIELDHINT_SActorRequest_Subject=Actor reference name for finding or creating an actor to connect with.
EDSTR_FIELDHINT_SActorHostedDelta_Subject=An actor to inspect for an offset between attach points.
EDSTR_FIELDHINT_SActorHostedDelta_LocalOffset=An optional additional data-driven offset to apply to the offset between attach points.
EDSTR_FIELDHINT_SActorHostedDelta_AttachQuerySource=The attach point at which the measurement begins.
EDSTR_FIELDHINT_SActorHostedDelta_AttachQueryTarget=The attach point at which the measurement ends.
EDSTR_FIELDHINT_SActorModelAspect_Person=Filters on the kind of person sitting in the chair, playing or watching the game.
EDSTR_FIELDHINT_SActorModelAspect_ObservingPoV=Filters on the point of view (PoV) through which an observer is watching the game. Only applies for observers.
EDSTR_FIELDHINT_SActorModelAspect_RegardsAs=Filters on how the person in the chair regards the owner of a target CActorModel, in terms of alliances.
EDSTR_FIELDHINT_SActorModelAspect_During=Filters on the context in which the current game is being observed, in terms of whether it is a live game or a replay.
EDSTR_FIELDHINT_SActorModelAspect_LabelExpression=Filters which label(s) the person in the chair must or must not have.
EDSTR_FIELDHINT_SActorModelAspect_ObservedPlayerType=While observing, filters on whether the observed player is a human or a computer.
EDSTR_FIELDHINT_SActorModelAspect_ObservedPlayerHasPlayerId=While observing, filters on the playerId of the observed player. Accepts input in a form like 1 or 1,12 or 1-5, or any combination thereof.
EDSTR_FIELDHINT_SActorModelAspect_ObservedPlayerHasTeamId=While observing, filters on the team belonged to by the observed player. Accepts input in a form like 1 or 1,12 or 1-5, or any combination thereof.
EDSTR_FIELDHINT_SActorModelAspect_ModelOwnerType=Filters on whether the owner of the target model is a human or a computer.
EDSTR_FIELDHINT_SActorModelAspect_ModelOwnerHasPlayerId=Filters on the owning playerId of the target model. Accepts input in a form like 1 or 1,12 or 1-5, or any combination thereof.
EDSTR_FIELDHINT_SActorModelAspect_ModelOwnerHasTeamId=Filters on the team belonged to by the owner of the target model. Accepts input in a form like 1 or 1,12 or 1-5, or any combination thereof.
EDSTR_FIELDHINT_SActorModelAspect_ModelOwnerLabelExpression=Filters which label(s) the owner of the target model must or must not have.
EDSTR_FIELDHINT_SActorModelAspect_Test=Specifies additional custom tests to be performed before validating the aspect.

EDSTR_FIELDHINT_SActorModelAspect_Model=If the aspect rule passes, the model to swap into the owning CActorModel.
EDSTR_FIELDHINT_SActorModelAspect_Name=Optional name for the aspect. Sent as an identifying parameter by several ModelAspect system messages.
EDSTR_FIELDHINT_SActorModelAspectSet_TriggerModel=If specified, indicates that the aspects in this set apply when the specified model is directly set, either via the ModelSwap message or the Model field in the CActorModel's ActorData.xml entry. When empty, means that the aspects in the set apply whenever any model not specified by another TriggerModel field is directly set or swapped into the CActorModel.
EDSTR_FIELDHINT_SActorModelAspectSet_Aspects=A list of rules that govern how aspects apply to the CActorModel, when it is set with a trigger model. Rules are evaluated from highest index to lowest index. The first rule to succeed wins.
EDSTR_FIELDHINT_SActorDeathBodySquib_Name=The name of the body squib specified by this field and its sibling data. Makes it easy for custom deaths to share body squibs without duplicating data repeatedly.
EDSTR_FIELDHINT_SActorDeathBodySquib_ActorModel=The name of the actor that will be used to play the model associated with this body squib. This value can be overridden to customize how and when the model plays.
EDSTR_FIELDHINT_SActorDeathBodySquib_Model=The model to create for this body squib. If empty, no model is created, though this is rarely done.
EDSTR_FIELDHINT_SActorDeathBodySquib_ModelSiteOps=Used to customize the position and rotation of the body squib model.
EDSTR_FIELDHINT_SActorDeathBodySquib_ModelAttachQuery=Used to control which attachment points play a body squib model. The Heart of the Swarm ragdoll deaths typically use Damage attach points for this.
EDSTR_FIELDHINT_SActorDeathBodySquib_ActorSound=The name of the actor that will be used to play the sound associated with this body squib. This value can be overridden to customize how and when the sound plays.
EDSTR_FIELDHINT_SActorDeathBodySquib_Sound=The sound to play with this body squib. If empty, no sound plays. If this value is set, but SoundSiteOps and SoundAttachQuery are left empty, then a sound is created for and hosted off of each body squib model.
EDSTR_FIELDHINT_SActorDeathBodySquib_SoundSiteOps=Used to customize the position of the body squib sound.
EDSTR_FIELDHINT_SActorDeathBodySquib_SoundAttachQuery=Used to control which attachment points play a body squib sound. It is often sufficient to simply play one on the Center attachment point. This is not exact for gib-style ragdolls where pieces fly everywhere, because the player might see a body part with a body squib model that does not have an associated sound, and conversely, he might see a body part with no body squib that plays a body squib sound. But in most cases it fairly accurate and it is hard to notice when it is not, so this technique is typically good enough.
EDSTR_FIELDHINT_SActorUnitImpactSoundExtras_TriggerModel=Triggers the usage of the associated sound data when the specified model is the current one for a given CActorUnit.
EDSTR_FIELDHINT_SActorUnitImpactSoundExtras_SoundActor=Overrides the harness actor that will house the associated generic impact sound for the trigger model.
EDSTR_FIELDHINT_SActorUnitImpactSoundExtras_Sound=Overrides the sound that will be played for the trigger model, upon impact from any CActorAction that does not have the ImpactSuppressUnitSound flag set.
EDSTR_FIELDHINT_SActorSoundLayer_PlayDelay=How long to delay playing this sound (in ms)
EDSTR_FIELDHINT_SActorQuadDecoration_Actor=The actor to create.
EDSTR_FIELDHINT_SActorQuadDecoration_SpawnInterval=How often in seconds to spawn an actor.
EDSTR_FIELDHINT_SActorQuadDecoration_TravelSpeed=How fast the actor travels, in units per second.
EDSTR_FIELDHINT_SActorQuadDecoration_TravelFunction=Used to modify how the actor moves along the path. It will still take the same amount of total time to travel from point A to point B, but these modify how it travels there.
EDSTR_FIELDHINT_SActorQuadDecoration_TravelFunctionParam=The parameter to the travel function. 0.5 is always linear. With Bias, a value greater than 0.5 will start fast and end slow, and a value less than 0.5 will start slow and end fast. With Gain, a value greater than 0.5 will be fast on the sides and slow in the middle, and a value less than 0.5 will be slow on the sides and fast in the middle.
EDSTR_FIELDHINT_SStatusChargeData_AbilCmd=The ability command to show charge info for
EDSTR_FIELDHINT_CActor_Flags=Flags apply to all actors. They allow you to change several actor behaviors, such as whether an actor is added to the External Finder and whether it will be added to a saved game.
EDSTR_FIELDHINT_CActor_FogVisibility=Controls how the actor appears under the fog of war. Hidden means it is

invisible. Snapshot means hidden for CActorUnit, but dimmed for everything else (though it also means CActorDoodads preserve themselves when told to do so). Dimmed means that the actor is tinted to match the fog of war tinting. Visible means the model shows through the fog of war.
EDSTR_FIELDHINT_CActorModel_ModelAspectSets=Configures model aspects for the CActorModel. Model aspects cause different models to appear to different players, depending on the rules in the aspect set.
EDSTR_FIELDHINT_CActorModel_ModelFlags=Flags that apply to all model actors. These flags change certain aspects of model behavior such as whether it can be hit tested (clicked on), whether it ignores walkables (pass through doodads rather than stand on top of them), and whether visibility tests are performed at all.
EDSTR_FIELDHINT_CActorModel_ModelMaterialGlazeDisplayLimit=Optionally limits the number of glazes that are drawn on the model at any given time. Can be used to maintain performance in cases where a large number of materials might occasionally be applied simultaneously.
EDSTR_FIELDHINT_CActorModel_LookAtPriorityList=Prioritizes CActorLookAts that might drive turret groups on this CActorModel. In this way, one CActorLookAt can drive a turret group, but then be temporarily overridden by another, higher priority CActorLookAt. For instance, a tank turret that normally points at an attack target might be temporarily overridden by an ability that casts a spell with visual FX that come from the barrel of the turret.
EDSTR_FIELDHINT_CActorModelMaterial_MaterialType=Determines whether the material replaces materials on the target model or serves as an overlay on top of them, like a glaze.
EDSTR_FIELDHINT_CActorModelMaterial_ModelMaterialFlags=Flags that apply to all CActorModelMaterial actors. These flags change certain aspects of a material instance, such as whether it animates when displayed under the fog of war.
EDSTR_FIELDHINT_CActorQuad_Decoration=Used to spawn a decorative actor that travels down the length of the quad along the Height axis.
EDSTR_FIELDHINT_CActorSite_SiteFlags=Flags that apply to all site actors. These flags change certain aspects of site behavior, such as whether the site automatically kills itself if no actors are hosting off it.
EDSTR_FIELDHINT_CActorSound_PlayMode=Controls the way that the sound layers are played.
EDSTR_FIELDHINT_CActorSound_LoopCount=-1 means loop forever, 0 means play once. In 'Sequence' play mode, each individual sound counts as 1 play. 'All' play mode not currently supported, will only play once.
EDSTR_FIELDHINT_CActorAction_ActionFlags=Flags for customizing various details about how an attack's FX behave. These flags handle a number of specific (though sometimes consequential) edge cases and are therefore frequently unnecessary.
EDSTR_FIELDHINT_CActorCutscene_FilePath=The path of the cutscene to use with this actor.
EDSTR_FIELDHINT_CActorGlobalConfig_LookAtTypes=Shared lookAt types that can be used by any CActorLookAt actors. The data or triggers can set one of these types on a target CActorLookAt to control its behavior.
EDSTR_FIELDHINT_CActorLookAt_HostTarget=The target CActorBearings that the look at will point to when activated.
EDSTR_FIELDHINT_CActorLookAt_HostTargetSiteOps=Site ops that modify or control where the look at will point to when activated.
EDSTR_FIELDHINT_CActorLookAt_Type=Points at data that specifies the turret groups controlled by the CActorLookAt, and how they blend together to look at or away from a given target.
EDSTR_FIELDHINT_CActorLookAt_Types=A local array of look at types that enable the CActorLookAt to switch behaviors on the fly. The data specified here is only selectable by the containing actor.
EDSTR_FIELDHINT_CActorSelection_SelectionFilter=This field only affects actors that have the IsSyncSelection flag.
EDSTR_FIELDHINT_CActorSiteOpDeltaSum_Deltas=A list of actors to inspect for attach point measurements. The siteOp then positions its owning actor by adding the sum of each attach point difference.
EDSTR_FIELDHINT_CActorSiteOpDeltaSum_DeltaSumFlags=Flags specific to this siteOp for configuring its operation.
EDSTR_FIELDHINT_CActorSiteOpPersistentOffset_HostInitial=Combined with HostInitialSiteOps, provides the initial bearings that are transformed into local space.
EDSTR_FIELDHINT_CActorSiteOpPersistentOffset_HostInitialSiteOps=Combined with HostInitial, provides the initial bearings that are transformed into local space.
EDSTR_FIELDHINT_CActorSiteOpYawLimiter_HostYawLimiter=Provides the frame of reference that the YawHalfAngle is with respect to.
EDSTR_FIELDHINT_CActorSiteOpYawLimiter_HostYawLimiterSiteOps=Optional siteOp chain to adjust the frame

of reference.
EDSTR_FIELDHINT_CActorSiteOpYawLimiter_YawHalfAngle=Specifies the maximum allowable yaw angle (rotation about the up axis) with respect to the frame of reference.
EDSTR_FIELDHINT_CActorSiteOpYawLimiter_ExtraYawAngle=If the incoming angle is less than YawHalfAngle, then it is left unchanged. If it is between YawHalfAngle and YawHalfAngle+ExtraYawAngle, then it is limited to YawHalfAngle. If it exceeds YawHalfAngle+ExtraYawAngle, then the angle is adjusted to face the reference frame forward direction.
EDSTR_FIELDHINT_CActorSiteOpPitchLimiter_HostPitchLimiter=Provides the frame of reference that the PitchHalfAngle is with respect to.
EDSTR_FIELDHINT_CActorSiteOpPitchLimiter_HostPitchLimiterSiteOps=Optional siteOp chain to adjust the frame of reference.
EDSTR_FIELDHINT_CActorSiteOpPitchLimiter_PitchHalfAngle=Specifies the maximum allowable pitch angle (rotation about the right axis) with respect to the frame of reference.
EDSTR_FIELDHINT_CActorUnit_ImpactSoundActor=Specifies the harness actor that will house the generic impact sound for this unit, should it exist.
EDSTR_FIELDHINT_CActorUnit_ImpactSound=Specifies the sound that will be played upon impact from any CActorAction that does not have the ImpactSuppressUnitSound flag set.
EDSTR_FIELDHINT_CActorUnit_ImpactSoundExtras=Specifies extra generic impact sounds and sound actors that are played depending on the model that is currently swapped into this CActorUnit.
EDSTR_FIELDHINT_CActorUnit_RingRadius=Overrides the radius used to autoscale a CActorUnitRing to this actor.
EDSTR_FIELDHINT_CActorUnit_WalkAnimMoveSpeed=The number of game units per second that the walk animation is animated to, so that the unit's model doesn't get 'slippy feet' visual artifacts. A WalkAnimMoveSpeed of 3 means the unit is animated to walk at a rate of 3 tiles per second in the default case. (The artist creating the asset actually types a value that is 100 times this into the asset in MAX, because in that context, a tile is 100 inches.)
EDSTR_FIELDHINT_CActorUnit_WalkAnimTimeScalingAsFlyer=Enables a ground unit to avoid time scaling its walk animations, as if it was a flyer. Normally, ground units automatically modify their walk animation time scale as they accelerate and decelerate, to ensure their feet do not appear to slide on the ground. Flyers do not need this time scaling, because they do not have feet that need to be matched to the ground. However, a ground unit like a hovercraft does not need this time scaling either, because it also does not have feet and appears to hover over the ground like a flyer. In short, this flag exists to handle the hovercraft scenario.
EDSTR_FIELDHINT_CActorUnit_StatusChargeData=Information about charges to show above the unit
EDSTR_FIELDHINT_CActorUnitRing_RingRadius=Radius at which the ring model was authored at.