BioWare Aurora Engine

Situated Object (Door and Placeable Object) Format

1. Introduction

A **Situated Object** is a base object type from which the **Door** and **Placeable Object** types are derived. Situated objects contain all the shared properties of Doors and Placeable objects. This document will first discuss these shared properties, then discuss the properties that are specific to doors and placeable objects themselves.

A **Door** is an object type that serves several functions. When closed, it blocks movement of creatures, and may block line of sight. Additionally, doors may serve as an area transition targets and trap locations.

A **Placeable Object** is an object type that adds to the ambience of an area, beyond the level provided by the tiles in the area's tileset. They may be purely decorational and inactive, or they may be set to be useable by players, whether as operable switches, as containers, or as traps.

Doors and Placeable objects are stored in the game and toolset using BioWare's Generic File Format (GFF), and it is assumed that the reader of this document is familiar with GFF.

Doors and Placeable objects can be blueprints or instances. Door blueprints are saved as GFF files having a UTD extension and "UTD" as the FileType string in their header. Door instances are stored as Door Structs within a module's GIT files. Similarly, Placeable object blueprints are UTP files with "UTP" as their FileType string, and Placeable instances are Placeable Structs within a module's GIT files.

2. Base Situated Object Struct

The tables in this section describe the GFF Struct for a basic Situated object. All Doors and Placeables share the properties of a Situated object.

2.1 Common Situated Fields

The Table below lists the Fields that are present in all Situated Structs, regardless of where they are found.

Table 2.1: Fields in all Door/Placeable Structs

Label	Туре	Description	
AnimationState	BYTE	Specifies animation state of the object. Open, closed,	
		destroyed, activated, etc.	
Appearance	DWORD	Appearance ID. Index into an appearance-related 2da.	
		Either doortypes.2da or placeables.2da.	
AutoRemoveKey	BYTE	I is the net should be desired to in the intention of	
		the creature that opens this object.	
		0 if the key should remain in the inventory of the	
		creature that opens this object.	
		This property applies only if a key item is required to	
		open this object. (ie., KeyRequired is 1)	

CloseLockDC	BYTE	DC to lock the object. If 0, anyone can lock it.	
Conversation	CResRef	ResRef of DLG file to use if a player converses with	
		this object via ActionStartConversation().	
CurrentHP	SHORT	Current HP of the object. For instances and blueprints	
	2110111	in the toolset, this is always the same as HP. In a game,	
		it may be different.	
Description	CExoLocString	Description of this object. Unlike most object types, the	
		game does not show the description for a Door when a	
		player examines it. It does show the description for a	
		Placeable, however.	
DisarmDC	BYTE	DC to disarm the trap, if any, attached to this object.	
Faction	DWORD	Faction ID of this object. Index into the list of faction	
		IDs in the module's repute.fac.	
Fort	BYTE	Fortitude save	
Hardness	BYTE	Hardness of this object. Damage reduction against	
		physical attacks. All slashing, piercing, and	
		bludgeoning damage is reduced by this amount to a	
		minimum of 0.	
HP	SHORT	Max Hit Points of this object.	
Interruptable	BYTE	Conversation can be interrupted.	
Lockable	BYTE	1 if this object can be locked after it has been unlocked.	
		0 if this object cannot be locked again by a Creature or	
		player after it has been unlocked.	
Locked	BYTE	1 if this object is locked.	
		0 if this object is not locked.	
LocName	CExoLocString	Localized name of this object. Appears in the toolset's	
		blueprint palette and as the name of this object in the	
		game.	
OnClosed	CResRef	OnClosed event	
OnDamaged	CResRef	OnDamaged event	
OnDeath	CResRef	OnDeath event	
OnDisarm	CResRef	OnDisarm event	
OnHeartbeat	CResRef	OnHeartbeat event	
OnLock	CResRef	OnLock event	
OnMeleeAttacked	CResRef	OnPhysicalAttacked event. Fires when object is	
		attacked via melee or ranged non-magical attack.	
OnOpen	CResRef	OnOpen event	
OnSpellCastAt	CResRef	OnSpellCastAt event	
OnTrapTriggered	CResRef	OnTrapTriggered event. If blank, then use the trap	
		script specified by the TrapType.	
OnUnlock	CResRef	OnUnlock event	
OnUserDefined	CResRef	OnUserDefined event	
OpenLockDC	BYTE	DC to unlock this object, if this object has been locked	
		by setting Locked to 1.	
Plot	BYTE	Plot flag. Plot objects cannot be damaged or destroyed	
		by Creatures or players.	
PortraitId	WORD	Index into portraits.2da. See Trigger format document,	
		Section 3.2 Portraits.	
Ref	BYTE	Reflex save	
Tag	CExoString	Tag of the object. Up to 32 characters.	
TemplateResRef	CResRef	For blueprints (UTD files), this should be the same as	
TemplateResRef	CResRef	the filename.	
TemplateResRef	CRESKEI		

TrapDetectable	BYTE	1 if Trap can be detected	
		0 if Trap cannot be detected	
TrapDetectDC	BYTE	DC to detect the Trap. Toolset enforces 1-250 range.	
TrapDisarmable	BYTE	1 if Trap can be disarmed.	
		0 if Trap cannot be disarmed.	
TrapFlag	BYTE	1 if the object Trapped.	
		0 if the object is not Trapped.	
TrapOneShot	BYTE	1 if the Trap disappears after firing.	
		0 if the Trap never disappears.	
TrapType	BYTE	Index into traps.2da.	
		Specifies the trap type, if the object has a Trap.	
		See Section 3.3 Traps, in the Trigger format	
		document.	
Will	BYTE	Will save	

2.2. Situated Blueprint Fields

The Top-Level Struct in a UTD or UTP file contains all the Fields in Table 2.1 above, plus those in Table 2.2 below.

Table 2.2: Fields in Door/Placeable Blueprint Structs

Label	Туре	Description	
Comment	CExoString	Module designer comment.	
PaletteID	BYTE	ID of the node that the object Blueprint appears under	
		in the object's blueprint palette.	
TemplateResRef	CResRef The filename of the UTD/UTP file itself. It is an error		
		if this is different. Certain applications check the value	
		of this Field instead of the ResRef of the actual file.	
		If you manually rename a UTD/UTP file outside of the	
		toolset, then you must also update the TemplateResRef	
		Field inside it.	

2.3. Situated Instance Fields

A Door Instance Struct in a GIT file contains all the Fields in Table 2.1, plus those in Table 2.3 below.

Table 2.3: Fields in Door/Placeable Instance Structs

Label	Туре	Description
Bearing	FLOAT Orientation of the object, expressed as a bearing in	
		radians measured counterclockwise from north. This is
		the opposite of the direction of the wireframe arrow
		attached to the object in the toolset's area viewer.
TemplateResRef	CResRef	For instances, this is the ResRef of the blueprint that
		the instance was created from.
X	FLOAT	(x,y,z) coordinates of the object within the Area that it
Y		is located in.
Z		

2.4. Situated Game Instance Fields

After a GIT file has been saved by the game, the Door Instance Struct not only contains the Fields in Table 2.1 and Table 2.3, it also contains the Fields in Table 2.4.

Table 2.4: Fields in Door/Placeable Instance Structs in SaveGames

Label	Туре	Description	
ActionList	List	List of Actions stored on this object	
		StructID 0. See Section 6 of the Common GFF	
		Structs document.	
AnimationDay	DWORD		
AnimationTime	DWORD		
EffectList	List	List of Effects stored on this object	
		StructID 2. See Section 4 of the Common GFF	
		Structs document.	
ObjectId	DWORD	Object ID used by game for this object.	
VarTable	List	List of scripting variables stored on this object.	
		StructID 0. See Section 3 of the Common GFF	
		Structs document.	

3. Doors

3.1. Door Struct

Door Structs contain all the Fields in Table 2.1, plus those in Table 3.1 below.

Table 3.1: Fields Unique to Door Structs

Label	Туре	Description	
AnimationState	BYTE	Specifies animation state of the object.	
		0 = closed	
		1 = opened1 (opened in same direction as wireframe	
		orientation arrow in toolset area viewer)	
		2 = opened2 (opened in opposite direction to wireframe	
		orientation arrow in toolset area viewer)	
Appearance	DWORD	Appearance ID. Index into doortypes.2da.	
		If greater than 0, use this to determine how the door	
		looks.	
		If 0, use GenericType instead to determine how the	
		door looks.	
GenericType	BYTE	If Appearance is 0, then GenericType determines the	
		Door appearance by lookup in genericdoors.2da.	
LinkedTo	CExoString	Tag of the Waypoint or Door that this Door links to in	
		an area transition.	
LinkedToFlags	BYTE	0 if this Door does not link to anything	
		1 if this Door is an Area Transition and links to another	
		Door.	
		2 if this Door is an Area Transition and links to a	
		Waypoint.	
LoadScreenID	WORD	Index into loadscreens.2da.	
		Specifies a loading screen to use when following this	
		Area Transition, if the transition causes the player to go	
		to a different area.	
		If LoadScreenID $== 0$, then use the default loading	
		screen as specified by the destination area.	
		See Section 2.3 in the Area GFF doc.	
OnClick	CResRef	OnAreaTransitionClick event. Fires when a player	
		clicks on this door while it is opened in order to follow	
		its area transition.	
OnFailToOpen	CResRef	OnFailToOpen event	

3.2. Door Blueprint Struct

Door Blueprints contain the Fields listed in Tables 2.1, 2.2, and 3.1.

3.3. Door Instance Struct

3.3.1. Door Instance Fields

Door Instances placed in the toolset contain the Fields listed in Tables 2.1, 2.3, and 3.1.

3.3.2. Relation of Door Instance Models to Appearance and GenericType

The appearance of a door instance is determined by where the door is located on a tile. Doors instances can only be painted at a location that corresponds to a **door hook point** on a tile. The hook point specifies the location and orientation at which a door can be placed, and it also places restrictions on what model the door can have when placed there.

Hook points on a tile are determined by both the tileset .SET file and the actual model .MDL file for the tile. It is an error if the SET file and MDL files do not agree, although the toolset itself does not check for consistency. It only looks at the SET file information. BioWare's BuildTil utility however, enforces consistency between the SET file and the MDL file, modifying the SET file if necessary to make it consistent with the MDL file.

There are two types of hook point: **Unique** (aka. **Tileset-specific**) and **Generic**.

Unique door hookpoints are attached to tiles that contain doorway geometry that demands a door of a specific appearance. The SET file entry for a Unique door hookpoint contains an index into doortypes.2da. Any door instance painted or moved onto a Unique hookpoint will be modified to have its Appearance match the one implied by the hookpoint.

Generic door hookpoints are attached to tiles that contain a generic doorway that can fit a variety of door appearances. The SET file entry for a Generic door hookpoint contains an index into genericdoors.2da. Any door instance painted or moved onto a Generic hookpoint will be modified to have Appearance 0, and its model will then be determined by its GenericType.

If a door is moved from a Generic hookpoint to a Unique hookpoint, its Appearance changes, but its GenericType remains the same, so that when moved back to a Generic hookpoint, it will once again have the model that it had before.

3.4. Door Game Instance Fields

After a GIT file has been saved by the game, the Door Instance Struct contains the Fields in Tables 2.1, 2.3, 2.4, 3.1, and 4.4.

Table 3.4: Fields in Door Instance Structs in SaveGames

Label	Type	Description
SecretDoorDC	BYTE	Obsolete. Always 0.

3.5. The 2DA Files Referenced by Door Fields

3.5.1. Door Tileset-Specific Appearances

If a door has an Appearance Field value greater than 0, then its model is determined by using its Appearance as an index into doortypes.2da.

Table 3.5.1: doortypes.2da columns

Column	Type	Description	
Label	String	Programmer label	
Model	String	ResRef of MDL model file to display for the door	
TileSet	String	ResRef of tileset SET file	
TemplateResRef	String	ResRef of UTD file	
StringRefGame	Integer	StringRef of text to display as the name of the door	
		appearance in the toolset	
BlockSight	Integer	0 if the door does not block sight in-game	
		1 if the door blocks sight in-game	
VisibleModel	Integer	0 if the door model is not visible. Invisible-model doors	
		are always open.	
		1 if the door model is visible. Visible doors can be opened	
		and closed.	
SoundAppType	Integer	Index into placeableobjsnds.2da.	

3.5.2. Door Generic Appearances

If a door has a value of 0 for its Appearance Field, then its model is determined by using its GenericType as an index into genericdoors.2da.

Table 3.5.2: genericdoors.2da columns

Column	Type	Description	
Label	String	Programmer label	
StrRef	String	StringRef of text to display as the name of the door in the	
		game.	
ModelName	String	ResRef of MDL model file to use for the door.	
BlockSight	Integer	0 if the door does not block sight in-game	
		1 if the door blocks sight in-game	
TemplateResRef	String	ResRef of UTD file	
VisibleModel	Integer	0 if the door model is not visible. Invisible-model doors	
		are always open.	
		1 if the door model is visible. Visible doors can be opened	
		and closed.	
SoundAppType	Integer	Index into placeableobjsnds.2da.	
Name	StrRef	StringRef of text to display as the name of the generic door	
		appearance in the toolset	

4. Placeable Objects

4.1. Placeable Struct

Placeable Structs contain all the Fields in Table 2.1, plus those in Table 4.1 below.

Table 4.1.1: Fields Unique to Placeable Structs

Label	Туре	Description
AnimationState	BYTE	Specifies animation state of the object. See Table 4.1.2
BodyBag	BYTE	Index into bodybag.2da.
		Specifies the body bag left behind when this object is
		destroyed while it still contains inventory.
HasInventory	BYTE	1 if the Placeable has inventory
		0 if the Placeable has no inventory
ItemList	List	List of InventoryObject Structs.
		StructID = index of Struct in list.
		This list is saved only if the Placeable contains at least
		1 item in its inventory.
OnInvDisturbed	CResRef	OnInventoryDisturbed event
OnUsed	CResRef	OnUsed event
Static	BYTE	1 if the Placeable is static.
		0 if the Placeable is nonstatic.
		Static objects are loaded client-side and are
		inaccessible through scripting because the server
		forgets about them after telling the client where they
		are located and what they look like. They cannot be
		interacted with in any way by the players or creatures
		in a game, and they cannot have any animation state
		other than the default.
		Make objects Static to improve client and server
		performance, and if they are present only for
		decoration.
Type	BYTE	Obsolete. Not used. Always 0.
Useable	BYTE	1 if the Placeable can be used by a player.

The AnimationState of a placeable object specifies what animations it should play. Table 4.1.2 lists the possible animation states. Not all animation states are available for all Placeable objects. A particular animation state is only available if the model (MDL) file for its appearance actually contains an animation of the name specified in Table 4.1.2.

Table 4.1.2: Placeable Object Animation States

Name	Animation State	Required animation on model
default	0	default
open	1	open
closed	2	close
destroyed	3	dead
activated	4	on
deactivated	5	off

The ItemList of a Placeable Object contains InventoryObject Structs. The Fields in an InventoryObject Struct vary depending on whether the inventory belongs to a Placeable Object Blueprint or Instance. In both cases, however, InventoryObjects contain at least the Fields in Table 4.1.3.

Table 4.1.3: Fields in all InventoryObject Structs

Label	Type	Description
Repos_PosX	WORD	x-position of item in inventory grid
Repos_PosY	WORD	y-position of item in inventory grid

4.2. Placeable Blueprint Struct

Placeable Object Blueprints contain the Fields listed in Tables 2.1.1, 2.2, and 4.1.

Table 4.2.1: Fields Unique to Placeable Structs

Label	Type	Description
ItemList	List	List of InventoryObject blueprint Structs.
		StructID = index of Struct in list.
		This list is saved only if the Placeable contains at least
		1 item in its inventory.

The Fields in an InventoryObject Struct vary depending on whether the inventory belongs to a Placeable Object Blueprint or Instance. For blueprints, an InventoryObject contains the Fields in Table 4.1.3 plus those in Table 4.2.2.

Table 4.2.2: Fields in InventoryObject Blueprint Structs

Label	Туре	Description
InventoryRes	CResRef	ResRef of UTI Item Blueprint file

4.3. Placeable Instance Struct

Placeable Object Instances placed in the toolset contain the Fields listed in Tables 2.1.1, 2.3, and 4.1.1.

Table 4.3.1: Fields Unique to Placeable Structs

Label	Туре	Description
ItemList	List	List of InventoryObject instance Structs.
		StructID = index of Struct in list.
		This list is saved only if the Placeable contains at least
		1 item in its inventory.

The Fields in an InventoryObject Struct vary depending on whether the inventory belongs to a Placeable Object Blueprint or Instance. For instances, an InventoryObject contains the Fields in Table 4.1.3 plus all the Fields normally contained in an Item Instance. See the Item Format document for details.

4.4. Placeable Game Instance Fields

After a GIT file has been saved by the game, the Placeable Instance Struct contains the Fields in Tables 2.1.1, 2.3, 2.4, 4.1.1, and 4.4.

Table 4.4: Fields in Placeable Instance Structs in SaveGames

Label	Туре	Description
Animation	INT	Game animation state.
		0 = default
		72 = destroyed
		73 = activated
		74 = deactivated
		75 = opened
		76 = closed
AnimationState	BYTE	This Field is removed by a savegame and replaced by
		the Animation Field given above.
DieWhenEmpty	BYTE	1 for body bag placeables that fade away after they
		have been fully looted.
		0 for all normal placeable objects and non-fading body

		bags.
GroundPile	BYTE	Obsolete Field. Always 0.
LightState	BYTE	1 if the light attached to this object is on
		0 if the light attached to this object is off, or there is no
		light.
Portal	CExoString	Portal Info. Used by the DM droppable portals.
TrapCreator	DWORD	Object ID of player who placed trap on this object
TrapFaction	DWORD	Faction ID of the trap placed on this object.

4.5. Placeable 2DA files

4.5.1. Placeables

Table 4.5.1: placeables.2da columns

Column	Type	Description
Label	String	Programmer label
StrRef	Integer	StrRef of the name to show in the toolset
ModelName	String	ResRef of MDL file to use as the model of the Placeable.
LightColor	Integer	Index into lightcolor.2da, specifying the color to use for a
		Light object to spawn in with the placeable object's model.
		If ****, then do not spawn in a light.
		Otherwise, spawn in "fx_placeable01.mdl" as a light
		source, and set its colors as specified in lightcolor.2da.
		See Table 2.5b in the Area File Format document for a
		description of lightcolor.2da.
LightOffcatV	Float	Offset in meters by which the light model's "root" node
LightOffsetX	rioat	•
LightOffsetY		should be moved after it has been spawned. Spawn in the
LightOffsetZ		light at the same location as the placeable.
SoundAppType	Integer	Index into placeableobjsnds.2da
ShadowSize	Integer	always 1
BodyBag	Integer	Index into bodybag.2da.
LowGore	String	ResRef of alternate MDL to use if the one listed under
		ModelName is too violent for the current violence settings
		in the game.

4.5.2. Placeable Object Sounds

Table 4.5.2: placeableobjsnds.2da columns

Column	Type	Description
Label	String	Programmer label
ArmorType	String	Determines sound to make when hit in combat. Possible
		values are:
		wood
		stone
		plate
		leather
		Value is used to randomly select a similarly named column
		in weaponsounds.2da, and the actual sound depends on
		which row of weaponsounds.2da is used by the weapon
		that is hitting the object.
Opened	String	ResRef of WAV file to play
Closed	String	ResRef of WAV file to play

Destroyed	String	ResRef of WAV file to play
Used	String	ResRef of WAV file to play
Locked	String	ResRef of WAV file to play

4.5.3. Bodybags

Table 4.5.3: bodybag.2da columns

Column	Туре	Description
LABEL	String	Programmer label
Name	Integer	StrRef of the name to show in the toolset
Appearance	Integer	Index into placeables.2da