BioWare Aurora Engine

Area File (ARE, GIT, GIC) Formats

1. Introduction

In the BioWare Aurora engine, each Area in a module is described by three files. Each file has the same filename, but a different extension. The following are brief summaries of the area file types.

ARE - static area information: area properties and tile layout.

GIT - dynamic area information - game object instances and sound properties.

GIC - area comments - contains toolset-only comments on game object instances.

The file type are in BioWare's Generic File Format (GFF) and it is assumed that the reader has some familiarity with GFF.

Conventions

This document describes the GFF structures used by the various area files, as well as any 2DA files that are referenced by Fields in those GFF files.

For GFF List Fields, the tables indicate the StructID used by the List elements.

Certain numerical GFF Fields have a range of allowable values, and any application that sets these values should respect the range limitations because there are no guarantees regarding how the game or toolset treats invalid values.

2. ARE Format

An ARE file contains static information about an Area. The Area properties that are stored in an ARE file cannot be modified via scripting.

Savegame (.SAV) files are ERFs that contain all the contents of the module, but with modifications and additions to the some of the original GFF files in the module. The ARE files are *not* one of the modified files, though. If the savegame file is for a module having the .nwm extension (from the nwm folder), then the ARE files are not even included in the savegame. When a nwm-saved game is reloaded, the game reads the ARE files from the original NWM. For all other saved games, it reads the ARE files from the SAV file.

2.1. ARE Top Level Struct

In the GFF header of an ARE file, the FileType value is "ARE".

Table 2.1: Fields in ARE Top Level Struct

Label	Туре	Description
ChanceLightning	INT	Percent chance of lightning (0-100)
ChangeRain	INT	Percent chance of rain (0-100)
ChanceSnow	INT	Percent chance of snow (0-100)
Comments	CExoString	Module designer comments
Creator_ID	INT	Deprecated; unused. Always -1.
DayNightCycle	BYTE	1 if day/night transitions occur, 0 otherwise

Expansion_List	List	Danracatad: unusad
Flags	DWORD	Deprecated; unused Set of bit flags specifying area terrain type:
riags	DWOKD	0x0001 = interior (exterior if unset)
		0x0001 = Interior (exterior if unset) 0x0002 = underground (aboveground if unset)
		0x0002 = underground (aboveground if unset) 0x0004 = natural (urban if unset)
		These flags affect game behaviour with respect to
		ability to hear things behind walls, map exploration
		visibility, and whether certain feats are active, though
		not necessarily in that order. They do not affect how the
		toolset presents the area to the user.
Height	INT	Area size in the y-direction (north-south direction)
11019110	1111	measured in number of tiles
ID	INT	Deprecated; unused. Always -1.
IsNight	BYTE	1 if the area is always night, 0 if area is always day.
121119110	BIIL	Meaningful only if DayNightCycle is 0.
LightingScheme	BYTE	Index into environment.2da
LoadScreenID	WORD	Index into loadscreens.2da. Default loading screen to
		use when loading this area.
		Note that a Door or Trigger that has an area transition
		can override the loading screen of the destination area.
ModListenCheck	INT	Modifier to Listen akill checks made in area
ModSpotCheck	INT	Modifier to Spot skill checks made in area
MoonAmbientColor	DWORD	Nighttime ambient light color (BGR format)
MoodDiffuseColor	DWORD	Nighttime diffuse light color (BGR format)
MoonFogAmount	BYTE	Nighttime fog amount (0-15)
MoonFogColor	DWORD	Nighttime fog color (BGR format)
MoonShadows	BYTE	1 if shadows appear at night, 0 otherwise
Name	CExoLocString	Name of area as seen in game and in left-hand module
		contents treeview in toolset. If there is a colon (:) in the
		name, then the game does not show any of the text up
		to and including the first colon.
NoRest	BYTE	1 if resting is not allowed, 0 otherwise
OnEnter	CResRef	OnEnter event
OnExit	CResRef	OnExit event
OnHeartbeat	CResRef	OnHeartbeat event
OnUserDefined	CResRef	OnUserDefined event
PlayerVsPlayer	BYTE	Index into pypsettings.2da. Note that the settings are
		actually hard-coded into the game, and pvpsettings.2da
		serves only to provide text descriptions of the settings
		(ie., do not edit pvpsettings.2da).
ResRef	CResRef	Should be identical to the filename of the area
SkyBox	BYTE	Index into skyboxes.2da (0-255). 0 means no skybox.
ShadowOpacity	BYTE	Opacity of shadows (0-100)
SunAmbientColor	DWORD	Daytime ambient light color (BGR format)
SunDiffuseColor	DWORD	Daytime diffuse light color (BGR format)
SunFogAmount	DATE	Daytime fog amount (0-15)
_	BYTE	, ,
SunFogColor	DWORD	Daytime fog color (BGR format)
SunFogColor SunShadows	DWORD BYTE	Daytime fog color (BGR format) 1 if shadows appear during the day, 0 otherwise
SunFogColor SunShadows Tag	DWORD BYTE CExoString	Daytime fog color (BGR format) 1 if shadows appear during the day, 0 otherwise Tag of the area, used for scripting
SunFogColor SunShadows Tag Tile_List	DWORD BYTE CExoString List	Daytime fog color (BGR format) 1 if shadows appear during the day, 0 otherwise Tag of the area, used for scripting List of AreaTiles used in the area. StructID 1.
SunFogColor SunShadows Tag Tile_List TileSet	DWORD BYTE CExoString List CResRef	Daytime fog color (BGR format) 1 if shadows appear during the day, 0 otherwise Tag of the area, used for scripting List of AreaTiles used in the area. StructID 1. ResRef of the tileset (.SET) file used by the area
SunFogColor SunShadows Tag Tile_List	DWORD BYTE CExoString List	Daytime fog color (BGR format) 1 if shadows appear during the day, 0 otherwise Tag of the area, used for scripting List of AreaTiles used in the area. StructID 1. ResRef of the tileset (.SET) file used by the area Revision number of the area. Initially 1 when area is
SunFogColor SunShadows Tag Tile_List TileSet	DWORD BYTE CExoString List CResRef	Daytime fog color (BGR format) 1 if shadows appear during the day, 0 otherwise Tag of the area, used for scripting List of AreaTiles used in the area. StructID 1. ResRef of the tileset (.SET) file used by the area

Width	INT	Area size in the x-direction (west-east direction)
		measured in number of tiles
WindPower	INT	Strength of the wind in the area. None, weak, or strong (0-2).

Some of the color values in the above table are in BGR format. The bytes within a BGR DWORD value are arranged as 0BGR, where the 0 indicates that the first byte is 0x00. Since GFF integer data is stored on disk in little-endian/Intel byte ordering, with the least significant byte first, the bytes on disk will actually appear in the order R G B 0.

2.2. Environment and LightingScheme

The LightingScheme Field is an index into Environment.2da, which contains a list of preset visual area properties. When a user initially selects one of the lighting schemes in the toolset, it automatically sets default values for all the area properties that correspond to columns in environment.2da.

The user's choice of environment scheme is stored with the LightingScheme ARE Field, but has no further effect on any other area property beyond what they were initially set to on first choosing the lighting scheme. At any time, the user can manually edit the visual area properties and the edited values will be the ones that are saved.

Table 2.2a: environment.2da columns

Column	Туре	Description
LABEL	String	Programmer label
STRREF	Integer	StrRef of localized text description to show to user in the
		listview on the Visual page of the Area Properties dialog.
DAYNIGHT	String	"cycle", "light", or "night". Assume night if none of the
		above.
LIGHT_AMB_RED	Integer	ambient day color (0-255)
LIGHT_AMB_GREEN		
LIGHT_AMB_BLUE		
LIGHT_DIFF_RED	Integer	diffuse day color (0-255)
LIGHT_DIFF_GREEN		
LIGHT_DIFF_BLUE		
LIGHT_SHADOWS	Integer	flag for shadows on/off during the day (1 or 0)
DARK_AMB_RED	Integer	ambient night color (0-255)
DARK_AMB_GREEN		
DARK_AMB_BLUE		
DARK_DIFF_RED	Integer	diffuse night color (0-255)
DARK_DIFF_GREEN		
DARK_DIFF_BLUE		
DARK_SHADOWS	Integer	setting for shadows on/off during the day (1 or 0)
LIGHT_FOG_RED	Integer	day fog color (0-255)
LIGHT_FOG_GREEN		
LIGHT_FOG_BLUE	_	
DARK_FOG_RED	Integer	night fog color (0-255)
DARK_FOG_GREEN		
DARK_FOG_BLUE	Ŧ .	(0.15)
LIGHT_FOG	Integer	day fog amount (0-15)
DARK_FOG	Integer	night fog amount (0-15)
MAIN1_COLOR1 to	Integer	List of colors to use for MainLight1 property of tiles in the
MAIN1_COLOR4		area (see description of the AreaTile struct). MainLight
		colors are chosen randomly when painting a tile or
		randomly generating the initial tiles for an area.
		Values are indices into lightcolor.2da.

MAIN2_COLOR1 to	Integer	same as above, but for MainLight2
MAIN2_COLOR4		
SECONDARY_COLOR1 to	Integer	same as above, but for SourceLight1 and SourceLight2
SECONDARY_COLOR4		
WIND	Integer	wind strength (0-2)
SNOW	Integer	chance of snow (0-100)
RAIN	Integer	chance of rain (0-100)
LIGHTNING	Integer	chance of lightning (0-100)
SHADOW_ALPHA	Integer	shadow opacity (0.0 to 1.0)
SKYBOX	Integer	Index into skyboxes.2da (0 to 255)

The SkyBox Field is an index into skyboxes.2da, which describes the day/night appearance and behaviour of each skybox.

Table 2.2b: skyboxes.2da columns

Column	Type	Description
LABEL	String	Programmer label
STRING_REF	Integer	StrRef of localized text description to show for the skybox
		when selecting it in the toolset's Visual Area Properties
		dialog. If the StrRef is ****, then use the LABEL instead.
CYCLICAL	Integer	1 if the sky changes during transitions from day to night or
		vice versa.
		0 if the sky never changes. If 0, then the DAWN, DAY,
		DUSK, and NIGHT columns should all have the same
		value.
DAWN	String	ResRef of the TGA texture to apply to the skybox at the
DAY		specified time of day.
DUSK		Dawn and Dusk each last for one game hour. The duration
NIGHT		of a game hour and the start times for dawn and dusk are
		specified in the module properties in module.ifo.

2.3. Loading Screens

The LoadScreenID Field is an index into loadscreens.2da.

The first row in loadscreens.2da is special, and specifies that the loading screen to use is random.

Table 2.3: loadscreens.2da columns

Column	Type	Description
Label	String	Programmer label, also used for display to user if the StrRef is ****
ScriptingName	String	Identifier for the integer constant to use as the first argument to the NWScript function void SetAreaTransitionBMP(int nPredefinedAreaTransition, string sCustomAreaTransitionBMP="").
BMPResRef	String	ResRef of the TGA file to display for this loading screen
TileSet	String	ResRef of tileset that this loading screen represents. If the loading screen for an area is Random (index 0), then the game will randomly pick a loading screen whose TileSet Field matches the ResRef of the area being transitioned to. If there are no loading screens that match the destination area's tileset, then the game will pick any loading screen at random.
StrRef	Integer	StrRef of text to display in toolset to describe the loading screen.

2.4. PvP Settings

The PlayerVsPlayer Field is an index into pvpsettings.2da.

Table 2.4: pvpsettings.2da columns

Column	Type	Description
label	String	programmer label
value	Integer	matches up to hardcoded pvp settings in the game
strref	Integer	StrRef of text to display to describe the setting to the user

2.5. Area Tile List

The Tile_List Field is a list of the AreaTiles in the area. An AreaTile describes what tile is located at a given position in the area, how it is oriented, and what graphical effects exist on it.

To determine the x-y grid coordinates of a tile, use the following formulae:

x = i % w

y = i / w

where:

i = index of the AreaTile Struct in Tile_List

w = the value stored in the area's Width Field

% = modulus operator, ie., the remainder after the left-hand side by the right-hand side

/ = integer division, with result round down to nearest integer

The Fields in an AreaTile are given in the table below:

Table 2.5a: Fields in AreaTile Struct (StructID 1)

Label	Туре	Description
Tile_AnimLoop1	INT	Boolean values to indicate if the "AnimLoop01",
Tile_AnimLoop2		"AnimLoop02", or "AnimLoop03" animations on the
Tile_AnimLoop3		tile model should play (1) or not (0).
		An AnimLoop can only be set if the correspondingly
		named animation actually exists on the tile model.
		Otherwise, the Field value is 0.
Tile_Height	INT	Number of height transitions that this tile is located at.
		Should never be negative.
Tile_ID	INT	Index into the tileset file's list of tiles, to specify what
		tile to use
Tile_MainLight1	BYTE	Index into lightcolor.2da to specify mainlight color on
Tile_MainLight2		the tile. A tile can have up to 2 mainlights. If a
		mainlight does not exist on a tile or is off, the Field
		value is 0.
Tile_Orientation	INT	Orientation of tile model.
		0 = normal orientation
		1 = 90 degrees counterclockwise
		2 = 180 degrees counterclockwise
		3 = 270 degrees counterclockwise
Tile_SrcLight1	BYTE	0 if SourceLight is off or does not exist.
Tile_SrcLight2		1-15 to specify color and animation of sourcelight.
		See "Source Lights" section below for more details.

Some of the Fields in the AreaTile Struct are indices into lightcolor.2da, which is given below.

Table 2.5b: lightcolor.2da columns

Column	Type	Description
RED	Float	RGB colors (0.00 to 2.00).
GREEN		
BLUE		
LABEL	String	Programmer label
TOOLSETRED	Float	RGB colors (0.00 to 1.00).
TOOLSETBLUE		Used by toolset to display a color in the Color Selection dialog
TOOLSETGREEN		that appears when clicking a color square in the Tile Properties
		dialog.

Main Lights

A mainlight is an overall colored lighting effect that can be applied to a tile.

To determine if a tile has a mainlight, the application must inspect its model file. There should be a light node whose name is the tile model's ResRef in all-caps with "ml1" or "ml2" appended to the end. For example, if the tile model is tts_a01_01.mdl, then mainlight 1 on that tile would be a light node called "TTZ_A01_01ml1".

If a light node does not exist, then the toolset does disables the controls to set the light's value.

Source Lights

A sourcelight is a point on a tile where the graphics engine can attach the model of a flaming light source.

Not all tiles have sourcelights. To determine if a sourcelight is present, the application must inspect the tile's model file. There should be a dummy node whose name is the tile model's ResRef in all-caps with "sl1" or "sl2" appended to the end.

To create a sourcelight in the graphics engine, spawn "fx_flame01.mdl" at the position of the tile's "sl1" or "sl2" dummy node and play the animation whose name matches the value of the appropriate Tile_SrcLight Field.

Example: if the tile model as specified by the TileSet and Tile_ID is ttz_a01_01.mdl, and Tile_SrcLight2 is 14, then spawn fx_flame01.mdl at the "TTZ_A01_01s12" dummy node in the tile and set it to play the animation named "14".

To get a rough visual representation of the source light color associated with a given number, take the Tile_SrcLight value, multiply it by 2, and use it as an index into lightcolor.2da, which contains the RGB values. Note however, that editing lightcolor.2da will have no effect on the colors shown in the graphics engine; the colors are entirely determined by the animations embedded in fx_flame01.mdl. In other words, lightcolor.2da should match up with the model, not the other way around.

3. GIT Format

A GIT file contains dynamic information about an area. The main purpose of a GIT is to store lists of all the object instances in the area, but it is also used to store a few area properties that can be changed via scripting. When a game is saved, the game includes an updated version of the GIT file for each area in the .SAV file.

In the GFF header of a GIT file, the FileType value is "GIT".

Table 3.1: Fields in GIT Top Level Struct

Label	Туре	Description
AreaProperties	Struct	StructID 100. See table below.
Creature List	List	List of Creature instances. StructID 4
Door List	List	List of Door instances. StructID 8
Encounter List	List	List of Encounter instances. StructID 7
List	List	List of Item instances. StructID 0
Placeable List	List	List of Placeable instances. StructID 9
SoundList	List	List of Sound instances. StructID 6
StoreList	List	List of Store instances. StructID 11
TriggerList	List	List of Trigger instances. StructID 1
WaypointList	List	List of Waypoint instances. StructID 5

The Lists in a GIT are lists of all the game object instances in the area. The actual format of each of those Structs is specified in the respective file format documents for each game object type.

The AreaProperties Field is a Struct containing dynamic area properties, and contains the Fields given in the table below.

Table 3.2: Fields in GIT AreaProperties Struct (StructID 100)

Label	Туре	Description
AmbientSndDay	INT	Ambient sound to play during daytime. Index into
		ambientsound.2da.
AmbientSndDayVol	INT	Volumne of ambient sound during the day (0-127)
AmbientSndNight	INT	Ambient sound to play at night. Index into
		ambientsound.2da.
AmbientSndNitVol	INT	Volume of ambient sound at night (0-127)
EnvAudio	INT	Environmental audio effects to use for positional
		sounds instances in the area. Index into soundeax.2da.
MusicBattle	INT	Index into ambientmusic.2da. Should only point to
		indices where the "Resource" 2da value starts with
		"mus_bat_"
MusicDay	INT	Index into ambientmusic.2da.
MusicDelay	INT	Index into ambientmusic.2da.
MusicNight	INT	Index into ambientmusic.2da.

The 2da files referenced by the the Fields in the AreaProperties Struct are described below:

Table 3.3: ambientsound.2da columns

Column	Type	Description
Description	Integer	StrRef of localized text description to show to user
Resource	String	Looping sound to play. ResRef of wav file in ambient folder.
PresetInstance0 to PresetInstance7	String	ResRef of sound blueprints (UTS file) to automatically spawn into area when this 2da row is selected as the ambient sound for the area.
DisplayName	String	Alternative to Description, if Description is ****.

Table 3.4: ambientmusic.2da columns

Column	Type	Description
Description	Integer	StrRef of localized text description
Resource	String	Music to play. ResRef of bmu file in music folder.
Stinger1	String	Music to play when combat ends, if this entry is being played as
Stinger2		combat music. Can have up to 3 different stinger variants that
Stinger3		the game chooses randomly.
DisplayName	String	Alternative to Description, if Description is ****.

Table 3.5: soundeax.2da columns

Column	Type	Description
Label	String	Programmer label. Displayed to user if Description is ****.
Description	Integer	StrRef of string to display to user.

SaveGame GIT properties

When a game is saved, a GIT file for every area in the module is saved into the .SAV file. The savegame GIT serves the same purpose as the GIT created by the toolset, although of course, the contents of its object lists may differ significantly, and the AreaProperties Struct may have different Field values.

The game also saves some additional fields to the Toplevel GIT Struct, as given below:

Table 3.6: Fields in GIT Top Level Struct, added by SaveGame

Label	Туре	Description
AreaEffectList	List	List of AreaEffects.
		StructID 13.
CurrentWeather	BYTE	Weather conditions currently in area.
		0 = Clear
		1 = Rain
		2 = Snow
VarTable	List	List of variables stored on area.
		StructID 0. See Section 3 of the Common GFF
		Structs document.
WeatherStarted	BYTE	1 if weather specified by CurrentWeather is starting, 0
		otherwise.

4. GIC Format

The GIC file is used purely to store the Comment properties of all the object instances in the area.

The game does not read or use the toolset-saved Comments in any fashion, so including them in the GIT file with the rest of the object instance information would unnecessarily increase the size of SaveGame files (recall that SaveGame ERFs contain GIT files for each area).

There is a one-to-one correspondence between list elements in the GIC file and those in the GIT file for the same area. Note that Table 4.1 below and Table 3.1 above contain identically named lists.

In the GFF header of a GIC file, the FileType value is "GIC".

Table 4.1: Fields in GIC Top Level Struct

Label	Туре	Description
Creature List	List	List of Creature instances. StructID 4
Door List	List	List of Door instances. StructID 8
Encounter List	List	List of Encounter instances. StructID 7
List	List	List of Item instances. StructID 0
Placeable List	List	List of Placeable instances. StructID 9
SoundList	List	List of Sound instances. StructID 6
StoreList	List	List of Store instances. StructID 11
TriggerList	List	List of Trigger instances. StructID 1
WaypointList	List	List of Waypoint instances. StructID 5

The GIC Lists all contain the same type of Struct. The format of that Struct is given in the table below.

Table 4.2: Fields in Comment Struct

Label	Туре	Description
Comment	CExoString	Module designer's comment