

BioWare Aurora Engine

IFO File Format

1. Introduction

An IFO file is a module InFoRmation file. Every NWN module (.MOD or .NWM) or savegame (.SAV) is an Encapsulated Resource File (ERF) that contains an IFO file called "module.ifo".

The IFO file type is in BioWare's Generic File Format (GFF) and it is assumed that the reader has some familiarity with GFF. Many of the GFF Fields in an IFO file make references to 2-Dimensional Array (2DA) files, so it is also assumed that the reader is familiar with the 2DA format.

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

2. Top-Level Struct

2.1 Fields created by Toolset

When a module is saved by the toolset, the Top-Level GFF Struct of the module.ifo file has the Fields given in the table below.

For List Fields, the table indicates the StructID used by the List elements.

Certain numerical 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.

Table 2.1: Basic Fields in IFO Top Level Struct

Label	Type	Description
Expansion_Pack	WORD	Bit flags specifying what expansion packs are required to run this module. Once a bit is set, it is never unset. Bit 0 = Expansion 1, Bit 1 = Expansion 2, etc.
Mod_Area_List	List	List of Areas in the module. StructID 6.
Mod_CacheNSSList	List	List of scripts that the game should cache. StructID 9
Mod_Creator_ID	INT	Deprecated; unused. Is always set to 2.
Mod_CustomTlk	CExoString	Name of a custom TLK file to use with this module. This name does not include the ".tlk" file extension. Custom tlk files should be located in the "tlk" folder in the main game installation directory. For non-English languages that use dialogF.tlk in addition to dialog.tlk, the custom tlk file must also have a "F.tlk" counterpart in the tlk folder. To refer to a string from the module's custom TLK file, the StrRef's 0x01000000 bit should be set to 1. The application will then mask that bit off to 0 and use the resulting value as the StrRef index into the custom TLK file instead of the usual dialog.tlk. If the custom

		string cannot be found, it will attempt to retrieve the normal dialog.tlk StrRef.
Mod_CutsceneList	List	Deprecated; unused.
Mod_DawnHour	BYTE	Game hour at which dawn begins (0-23). Area lighting will begin transitioning from Night to Day colors over the course of 1 game hour.
Mod_Description	CExoLocString	Description of module
Mod_DuskHour	BYTE	Game hour at which dusk begins (0-23). Area lighting will begin transitioning from Day to Night colors over the course of 1 game hour.
Mod_Entry_Area	CResRef	Module's Starting Area
Mod_Entry_Dir_X Mod_Entry_Dir_Y	FLOAT	x and y components of Start Location's direction vector. This is a unit vector. Or in other words, the cosine and sine, respectively, of the waypoint's bearing in the xy plane, measured as an angle counterclockwise from the positive x-axis.
Mod_Entry_X Mod_Entry_Y Mod_Entry_Z	FLOAT	(x,y,z) coordinates of Module Start Location within the starting area. The toolset will refuse to save a module or area until the start location is located on a walkable portion of a tile. If the start location ends up on an unwalkable spot anyway (this may be caused if the tileset tiles or walkmeshes are in a state of flux) then the game will spawn players in at the closest walkable point.
Mod_Expan_List	List	Deprecated; unused
Mod_GVar_List	List	Deprecated; unused
Mod_Hak	CExoString	(Obsolete) Hak File used by module, without the ".hak" extension in its filename. If Mod_HakList exists, this value is used as the module's Hak Pak. Otherwise, it is ignored.
Mod_HakList	List	List of Hak Files used by module. Resources from the first Hak Paks in the list have the highest priority and override resources in later Hak Paks. StructID 8.
Mod_ID	Binary	Arbitrarily generated 16-byte number sequence assigned when toolset creates a new module. It is never modified afterward by toolset. The game saves out 32 bytes instead of 16. Applications other than the toolset can set this to all null bytes when creating a new IFO file.
Mod_IsSaveGame	BYTE	Boolean indicating if the module is a same game. 0 for modules saved by toolset. 1 for saved games.
Mod_MinGameVer	CExoString	Minimum version of the game and associated game resources required to run the module. Should be in n.nn format (eg., "1.26", "1.30"). The game and toolset will refuse to open a module if the module's minimum version is greater than the user's current version of the game. The value of this Field can only increase or stay the same. If this Field does not exist in the IFO, the default value is "1.22".
Mod_MinPerHour	BYTE	Number of real-time minutes per game hour. (1-255)
Mod_Name	CExoLocString	Name of module
Mod_OnAcquirItem	CResRef	OnAcquireItem event
Mod_OnActvtItem	CResRef	OnActivateItem event

Mod_OnClientEntr	CResRef	OnClientEnter event
Mod_OnClientLeav	CResRef	OnClientLeave event
Mod_OnCutsnAbort	CResRef	OnCutsceneAbort event
Mod_OnHeartbeat	CResRef	OnHeartbeat event
Mod_OnModLoad	CResRef	OnModuleLoad event
Mod_OnModStart	CResRef	OnModuleStart event; deprecated
Mod_OnPlrDeath	CResRef	OnPlayerDeath event
Mod_OnPlrDying	CResRef	OnPlayerDying event
Mod_OnPlrEqItm	CResRef	OnPlayerEquipItem event
Mod_OnPlrLvlUp	CResRef	OnPlayerLevelUp event
Mod_OnPlrRest	CResRef	OnPlayerRest event
Mod_OnPlrUnEqItm	CResRef	OnPlayerUnEquipItem event
Mod_OnSpawnBtnDn	CResRef	OnPlayerRespawn event
Mod_OnUnAcrcItem	CResRef	OnUnAcquireItem event
Mod_OnUsrDefined	CResRef	OnUserDefined event
Mod_StartDay	BYTE	Starting day (1-31)
Mod_StartHour	BYTE	Starting hour (0-23)
Mod_StartMonth	BYTE	Starting month (1-24)
Mod_StartMovie	CResRef	ResRef of movie in 'movies' folder to play when starting module
Mod_StartYear	DWORD	Starting year
Mod_Tag	CExoString	Module's Tag
Mod_Version	DWORD	Module version. Is always set to 3.
Mod_XPScale	BYTE	Percentage by which to multiply all XP gained through killing creatures.

2.2. Fields created by Game

When a module is saved, the game adds additional fields to the module.ifo file, as listed in the table below.

Table 2.2: Save-Game Fields in IFO Top Level Struct

Label	Type	Description
Creature List	List	Deprecated; unused
EventQueue	List	Game events that were queued up at the time the module was saved. StructID 43981
Mod_Effect_NxtId	DWORD64	ID to use for the next Effect
Mod_IsNWMFile	BYTE	Boolean to indicate if the game was saved from a NWM file (1) or MOD file (0).
Mod_NextCharId0	DWORD	Keeps track of which id to give the next character created
Mod_NextCharId1	DWORD	-
Mod_NextObjId0	DWORD	Keeps track of which id to give the next object created
Mod_NextObjId1	DWORD	-
Mod_NWMResName	CExoString	If this game was saved from a nwm module, then this is the filename of the nwm.
Mod_PlayerList	List	List of Players in the module. StructID 48813
Mod_Tokens	List	List of Custom Tokens in the module. StructID 7
Mod_TURDList	List	List of Temporary User Resource Data objects in the module. StructID 13634816

Mod_VarTable	List	List of Variables in the module and their values. StructID 0
--------------	------	---

3. Common Lists and Structs

Below are descriptions of each of the Lists present in an IFO file. Each section is titled by the Field's Label and contains the StructID of the Structs contained in the List

3.1 Mod_Area_List

Module Area List

A list of all the areas present in the module.

Table 3.1: Fields in Area List Struct (StructID 6)

Label	Type	Description
AreaName	CResRef	ResRef of area in module. There must be three files in the module that have this ResRef, with filetypes ARE, GIT, and GIC.
ObjectID	DWORD	ObjectID of the area. (Savegame only; not saved out by toolset)

3.2 Mod_CacheNSSList

Cached Script List

A list of scripts that should be cached by the NWN server while running the module. Typically, these are scripts that will be executed very often.

Table 3.2: Fields in Cached Script List Struct (StructID 9)

Label	Type	Description
ResRef	CResRef	ResRef of a script. Each script has an NSS source file and a corresponding NCS compiled script.

3.3 Mod_HakList

Hak Pak List

List of Hak Paks used by the module. The Hak Paks are listed in descending order of priority. The contents of the earlier Hak Paks in the list will override the contents of later Hak Paks.

Table 3.3: Fields in Hak Pak List Struct (StructID 8)

Label	Type	Description
Mod_Hak	CExoString	Filename of a Hak Pak used by this module, minus the .hak extension.

4. Save-Game Lists and Structs

Below are descriptions of each of the Lists present in an IFO file after the module has been saved by the game. Each section is titled by the Label of the List.

Most things ingame have an ObjectID by which the game references them, so ObjectIDs appear in many of the Structs in the savegame Lists.

4.1 EventQueue

Event Queue

List of Events in the module. See [Section 5](#) of the [Common GFF Structs](#) document.

4.2 Mod_PlayerList

Player List

List of Players in the module. Each Player Struct in the list has a StructID of 48813. The Player Struct itself is too large and complicated to discuss in this document, and merits an entire format-specification document of its own.

4.3 Mod_Tokens

Module Custom Tokens

List of custom tokens defined in the module via the NWScript function.

```
void SetCustomToken(int nCustomTokenNumber, string sTokenValue)
```

Table 4.3: Fields in a Token Struct (StructID 7)

Label	Type	Description
Mod_TokensNumber	DWORD	Custom Token number. nCustomTokenNumber argument from the SetCustomToken() function.
Mod_TokensValue	CExoString	Custom Token value. sTokenValue argument from the SetCustomToken() function.

4.4 Mod_TURDList

Temporary User Resource Data

List of player Temporary User Resource Data objects.

These objects are used to store player information for users who joined the game and then logged out. When a user joins a game, the user's login name and player character's name are checked against those in the current TURD List to determine if the user is a new player, or one who is returning to the game. Returning players have their gamestate information restored according to the information in the TURD.

Table 4.4a: Fields in a TURD Struct (StructID 13634816)

Label	Type	Description
EffectList	List	List of Effects. StructID 2. See Section 4 of the Common GFF Structs document.
Mod_MapAreasData	Binary	-
Mod_MapDataList	List	List of MapData. StructID 0, contains the indented Fields immediately below:
Mod_MapData	Binary	-
ModMapNumAreas	INT	-
TURD_AreaId	DWORD	ObjectID of area in which player logged out.
TURD_CalendarDay	DWORD	Day the TURD was generated
TURD_CommntyName	CExoString	Player Name with which the player logged

		into Multiplayer.
TURD_FirstName	CExoLocString	First name of the player character
TURD_LastName	CExoLocString	Last name of the player character
TURD_OrientatX TURD_OrientatY TURD_OrientatZ	FLOAT	Orientation of the player at logout
TURD_PersonalRep	List	List of Personal Reputations that other creatures hold toward the player. StructID 47787. See Table 4.4b below.
TURD_PlayerID	DWORD	ObjectID of the player
TURD_PositionX TURD_PositionY TURD_PositionZ	FLOAT	Position of the player at logout
TURD_RepList	List	List of reputations with each Faction in the module. StructID 43962, contains the indented Fields immediately below:
TURD_RepAmount	INT	Reputation with faction X, where X is the same as the index of the List element (allowed values are 0-100)
TURD_TimeOfDay	DWORD	Time the TURD was generated
VarTable	List	List of Variables stored on the character. StructID 0. See section 4.5.

Table 4.4b: Fields in a Personal Reputation Struct (StructID 47787)

Label	Type	Description
TURD_PR_Amount	INT	Reputation with the faction
TURD_PR_Day	DWORD	-
TURD_PR_Decays	BYTE	boolean
TURD_PR_Duration	INT	Measured in seconds
TURD_PR_ObjId	DWORD	ObjectID of creature that is considering the owner of this Personal Reputation element
TURD_PR_Time	DWORD	-

4.5 VarTable

Variable Table

List of scripting variables and their values. See [Section 3 of the Common GFF Structs document](#).