

BioWare Aurora Engine

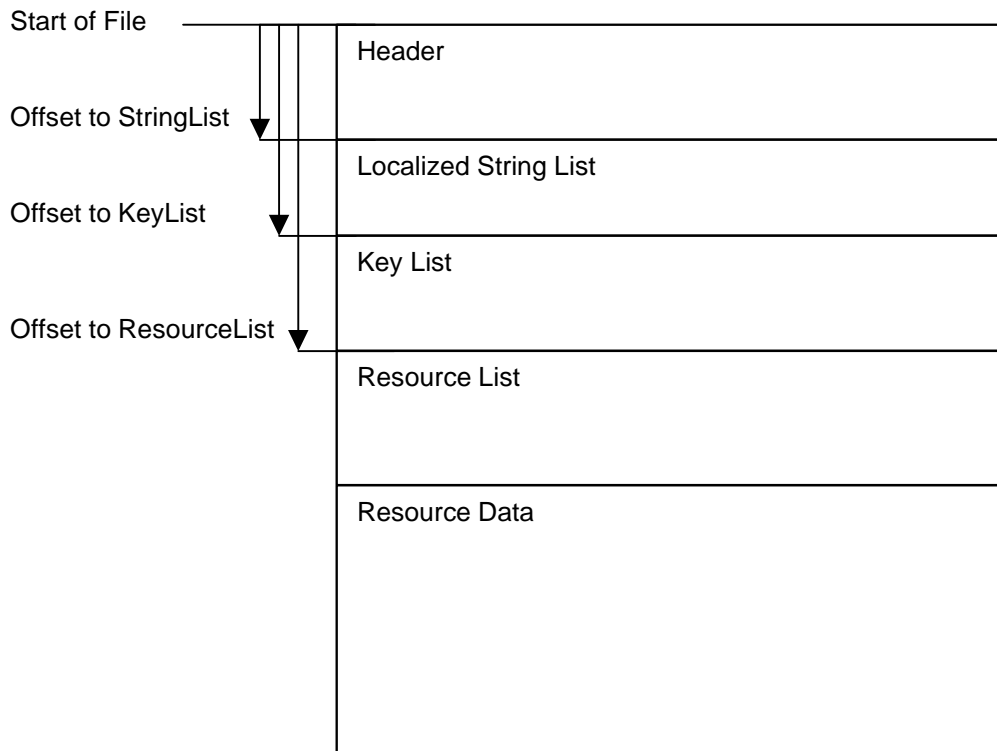
Encapsulated Resource File Format

NOTICE: This documentation provides information about specific file formats used with the BioWare Aurora Engine. It is intended for the use of software developers to create third-party software to work with the BioWare Aurora Engine and the BioWare Aurora Toolset. While this documentation is provided as a service, we may or may not provide updates, fixes and additions to the information provided herein. We reserve the right to change file formats without updating the documentation. Please refer to the For Developers FAQ (<http://nwn.bioware.com/developers/faq.html>) and the NWN End User License Agreement (EULA) for more information.

The Encapsulated Resource File (ERF) format is one of BioWare's methods of packing multiple files into a single file so that they may be treated as a single unit. In this regard, it is similar to .zip, .tar, or .rar.

BioWare Aurora Engine/Toolset files that use the ERF format include the following: .erf, .hak, .mod, and .nwm.

Global ERF Structure



ERF Header format

FileType	4 char	"ERF ", "MOD ", "SAV ", "HAK " as appropriate
Version	4 char	"V1.0"
LanguageCount	32 bit	number of strings in the Localized String Table
LocalizedStringSize	32 bit	total size (bytes) of Localized String Table
EntryCount	32 bit	number of files packed into the ERF
OffsetToLocalizedString	32 bit	from beginning of file, see figure above
OffsetToKeyList	32 bit	from beginning of file, see figure above
OffsetToResourceList	32 bit	from beginning of file, see figure above
BuildYear	4 bytes	since 1900
BuildDay	4 bytes	since January 1st

DescriptionStrRef	4 bytes	strref for file description
Reserved	116 bytes	NULL

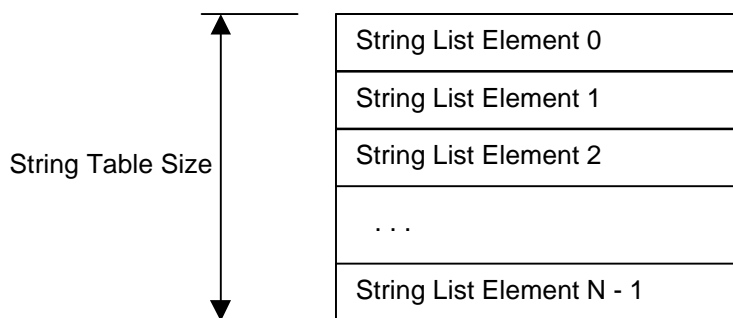
The Reserved part of the ERF header allows for additional properties to be added to the file format later while maintaining backward compatibility with older ERFs.

ERF Localized String List

The Localized String List is used to provide a description of the ERF. In .mod files, this is where the module description is stored. For example, during the Load Module screen in NWN (a BioWare Aurora Engine game), the module descriptions shown in the upper right corner are taken from the Localized String List. The game obtains the current Language ID from dialog.tlk, and then displays the ERF String whose LanguageID matches the dialog.tlk language ID.

String List Format

The String List contains a series of ERF String elements one after another. Note that each element has a variable size, encoded within the element itself. The LanguageCount from the ERF Header specifies the number of String List Elements.



String List Element Structure

Each String List Element has the following structure:

LanguageID	32 bit
StringSize	32 bit
String	Variable size as specified by the StringSize field

In .erf and .hak files, the String portion of the structure is a NULL-terminated character string. In .mod files, the String portion of the structure is a non-NULL-terminated character string. Consequently, when reading the String, a program should rely on the StringSize, not on the presence of a null terminator.

Language IDs

The following is a list of languages and their IDs:

Language	ID
English	0
French	1
German	2
Italian	3
Spanish	4

Polish	5
Korean	128
Chinese Traditional	129
Chinese Simplified	130
Japanese	131

Note, however, that the LanguageID actually stored in an ERF does not match up exactly to the LanguageIDs shown in the above table. Instead, it is 2 times the Language ID, plus the Gender (0 for neutral or masculine, 1 for feminine).

ERF Key List

The ERF Key List specifies the filename and filetype of all the files packed into the ERF.

Key List Format

The Key List consists of a series of Key structures one after another. Unlike the String List elements, the Key List elements all have the same size. The EntryCount in the ERF header specifies the number of Keys.

Key List Element 0
Key List Element 1
Key List Element 2
...
Key List Element N - 1

Key Structure

Each Key List Element has the following structure:

ResRef	16 bytes	Filename
ResID	32 bit	Resource ID, starts at 0 and increments
ResType	16 bit	File type
Unused	16 bit	NULLs

The ResRef is the name of the file with no null terminator and in lower case. A ResRef can only contain alphanumeric characters or underscores. It must have 1 to 16 characters, and if it contains less than 16 characters, the remaining ones are nulls.

The ResID in the key structure is redundant, because it's possible to get the ResID for any ERF Key by subtracting the OffsetToKeyList from its starting address and dividing by the size of a Key List structure.

When a file is extracted from an ERF, the ResRef is the name of the file after it is extracted, and the ResType specifies its file extension. For a list of ResTypes, see the section on ResTypes later in this document.

ERF Resource List

The Resource List specifies where the data for each file is located and how big it is.

Resource List Format

The Resource List looks just like the Key list, except that it has Resource List elements instead of Key List elements. The ERF header's EntryCount specifies the number of elements in both the Key List and the Resource List, and there is a one-to-one correspondence between Keys and Resource List elements.

Resource List Element 0
Resource List Element 1
Resource List Element 2
...
Resource List Element N - 1

Resource List Element Structure

Each Resource List Element corresponds to a single file packed into the ERF. The Resource structure specifies where the data for the file begins inside the ERF, and how many bytes of data there are.

OffsetToResource	32 bit	offset to file data from beginning of ERF
ResourceSize	32 bit	number of bytes

Resource Data

After the Resource List, all the data in an ERF consists of raw data for all the files that the ERF contains. The data for one file is packed right up against the data for the next file. The offsets and sizes in the Resource List specify where one file ends and another begins.

Resource Types

In an ERF Key, the ResType field specifies the file type of the associated file. See [Section 1.3 of the Key and BIF File Format document](#) for a table containing ResType values and their meanings.