## 22.6. `chunk` — Read IFF chunked data¶

This module provides an interface for reading files that use EA IFF 85 chunks. [1] This format is used in at least the Audio Interchange File Format (AIFF/AIFF-C) and the Real Media File Format (RMFF). The WAVE audio file format is closely related and can also be read using this module.

A chunk has the following structure:

| Offset | Length | Contents |
|---|---|---|
| 0 | 4 | Chunk ID |
| 4 | 4 | Size of chunk in big-endian byte order, not including the header |
| 8 | *n* | Data bytes, where *n* is the size given in the preceding field |
| 8 + *n* | 0 or 1 | Pad byte needed if *n* is odd and chunk alignment is used |

The ID is a 4-byte string which identifies the type of chunk.

The size field (a 32-bit value, encoded using big-endian byte order) gives the size of the chunk data, not including the 8-byte header.

Usually an IFF-type file consists of one or more chunks. The proposed usage of the [Chunk](#) class defined here is to instantiate an instance at the start of each chunk and read from the instance until it reaches the end, after which a new instance can be instantiated. At the end of the file, creating a new instance will fail with a [EOFError](#) exception.

*class* chunk.Chunk(*file*[, *align*, *bigendian*, *inclheader*])¶

Class which represents a chunk. The *file* argument is expected to be a file-like object. An instance of this class is specifically allowed. The only method that is needed is [read()](#). If the methods [seek()](#) and [tell()](#) are present and don't raise an exception, they are also used. If these methods are present and raise an exception, they are expected to not have altered the object. If the optional argument *align* is true, chunks are assumed to be aligned on 2-byte boundaries. If *align* is false, no alignment is assumed. The default value is true. If the optional argument *bigendian* is false, the chunk size is assumed to be in little-endian order. This is needed for WAVE audio files. The default value is true. If the optional argument *inclheader* is true, the size given in the chunk header includes the size of the header. The default value is false.

A [Chunk](#) object supports the following methods:

getname()¶
Returns the name (ID) of the chunk. This is the first 4 bytes of the chunk.

getsize()¶
Returns the size of the chunk.

close()¶
Close and skip to the end of the chunk. This does not close the underlying file.

The remaining methods will raise [IOError](#) if called after the [close()](#) method has been called.

isatty()¶
Returns False.

seek(*pos*[, *whence*])¶
Set the chunk's current position. The *whence* argument is optional and defaults to 0 (absolute file positioning); other values are 1 (seek relative to the current position) and 2 (seek relative to the file's end). There is no return value. If the underlying file does not allow seek, only forward seeks are allowed.

tell()¶
Return the current position into the chunk.

read([*size*])¶
Read at most *size* bytes from the chunk (less if the read hits the end of the chunk before obtaining *size* bytes). If the *size* argument is negative or omitted, read all data until the end of the chunk. The bytes are returned as a string object. An empty string is returned when the end of the chunk is encountered immediately.

skip()¶
Skip to the end of the chunk. All further calls to [read()](#) for the chunk will return `''`. If you are not interested in the contents of the chunk, this method should be called so that the file points to the start of the next chunk.

Footnotes

[1] "EA IFF 85" Standard for Interchange Format Files, Jerry Morrison, Electronic Arts, January 1985.

**Previous topic**

**Next topic**

**This Page**

- Show Source

**Navigation**