gciso Documentation

Joel Schumacher

Jun 25, 2018

Contents

CHAPTER 1

IsoFile

class gciso.**IsoFile**(*isoPath*)

The central class representing an .iso file. For information about many of it's attributes, see here: http://hitmen.c02.at/files/yagcd/yagcd/chap13.html#sec13

Parameters **isoPath** (*str*) – The path to the iso file.

gameCode
bytes

makerCode
bytes

diskId
int

version
int

gameName
bytes

dolOffset
int – Offset to the main executable DOL ("start.dol")

dolSize
int – Size of the main executable DOL ("start.dol")

fstOffset
int – Offset to the file system table

fstSize
int – Size of the file system table

maxFstSize
int – Maximum size of the file system table (relevant for games with multiple disks)

apploaderDate
bytes – Date (version) of the apploader (ASCII).

apploaderEntryPoint
int

apploaderCodeSize
int

apploaderTrailerSize
int

numFstEntries
int – The number of file system table entries (files and directories)

stringTableOffset
int – The offset to the FST string table

files
OrderedDict – A dictionary with keys being paths to all files in the .iso and values being tuples of the form *(offset, size)* (offset and size of the file). See Notes for added system files!

Notes

A couple of files are added to the files attribute, that are not listed in the FST:

- *boot.bin* - The header of the .iso file
- *bi2.bin* - More disk information (containing dol/FST offsets etc.)
- *fst.bin* - The file system table
- *start.dol* - The main executable DOL. See *DolFile* and *getDolFile()*

IsoFile may also be used as a context manager:

```
with IsoFile("melee.iso") as isoFile:
    data = isoFile.readFile("opening.bnr", 0)
    with isoFile.open("opening.bnr") as bnrFile:
        print(bnrFile.read())
```

Also all files that take a file path may raise `TypeError`, if the given path is not of type *bytes*.

close()
Closes the file

static fileInDir(*filePath*, *dirPath*)

Parameters

- **filePath** (*bytes*) –
- **dirPath** (*bytes*) –

Returns Whether the file *filePath* is inside the directory *dirPath*

Return type bool

fileOffset(*path*)

Parameters **path** (*bytes*) –

Returns The offset of the file with the given path inside the .iso file.

Return type int

fileSize(*path*)

Parameters **path** (*bytes*) –

Returns The size of the file with the given path inside the .iso file.

Return type bool

getBannerFile(*path*)
Creates a *BannerFile* from the file with the given *path*

Parameters **path** (*bytes*) –

Returns

Return type *BannerFile*

getDolFile(*path=b'start.dol'*)
Creates a *DolFile* from the file with the given *path*

Parameters **path** (*bytes*) – If no path is given, the main executable DOL *start.dol* is used.

Returns

Return type *DolFile*

isDir(*path*)

Parameters **path** (*bytes*) –

Returns Whether the given path belongs to a directory that exists inside the .iso and contains files.

Return type bool

isFile(*path*)

Parameters **path** (*bytes*) –

Returns Whether the given path belongs to a file that exists inside the .iso.

Return type bool

listDir(*path*)
Lists all files in a directory (including files in subdirectories, not including other directories).

Parameters **path** (*bytes*) –

Yields *bytes* – Filenames of the files in the directory. Relative to the directory being listed.

open(*path*)

Parameters **path** (*bytes*) –

Returns A wrapper of the given file.

Return type *IsoInternalFileWrapper*

Notes

See the notes of *IsoFile* and *IsoInternalFileWrapper* for examples.

readFile(*path*, *offset*, *count=-1*)
Reads *count* bytes from *offset* inside the file with path *path*

Parameters

- **path** (*bytes*) –

- **offset** (*int*) –
- **count** (*int*) – If count is negative, none or omitted, read until end of file

Returns The data read

Return type bytes

Raises

- IndexError – If *offset* is negative or greater than the file size.
- ValueError – If the read would go past the end of the file.

writeFile(*path*, *offset*, *data*)

Writes *data* to the file with path *path* inside the .iso at offset *offset*.

Parameters

- **path** (*bytes*) –
- **offset** (*int*) –
- **data** (*bytes*) –

Returns The number of bytes written

Return type int

Raises

- IndexError – If *offset* is negative or greater than the file size.
- TypeError – If *path* or *data* is not *bytes*
- ValueError – If the write would go past the end of the file, since it cannot change size.

CHAPTER 2

IsoInternalFileWrapper

class gciso.**IsoInternalFileWrapper**(*isoFile*, *offset*, *size*)

Wraps a file inside the .iso. You probably don't want to call this yourself. See *IsoFile.open()*.

Parameters

- **isoFile** (*IsoFile*) –
- **offset** (*int*) – Offset of the file inside the .iso file.
- **size** (*int*) – Size of the file

Notes

This class may also be used as a context manager, similar to *open* from the standard library.:

```
with isoFile.open(b'PlSs.dat') as f:
    f.seek(0x1000)
    data = f.read(0x30)
```

close()

Internally this is a NOP, but it exists to provide more compatibility with Python's own file objects.

read(*size=-1*)

Reads data starting from the current position.

Parameters **size** (*int*) – How many bytes to read. If *None*, negative or ommited, read until end of file

Returns The data read from the file

Return type bytes

Notes

See *IsoFile.readFile()* for exceptions this function might raise.

seek(*offset*, *whence=0*)

Moves the current position inside the file according to the given offset.

Parameters

- **offset** (*int*) – The offset
- **whence** (*int*) – One of either 0, 1 or 2. See notes.

Returns The current position after seeking.

Return type int

Raises ValueError – If whence is not in *{0, 1, 2}*

Notes

If *whence* is 0, the offset will be interpreted relative to **the start of the file**.

If *whence* is 1, the offset will be interpreted relative to **the current position**.

If *whence* is 2, the offset will be interpreted relative to **the end of the file**. Usually in this case *offset* is negative.

You may also seek before or after the end of the file, though write and read operations will most likely fail.

tell()

Returns The current position inside the file

Return type int

write()

Writes data to the current position.

Parameters **data** (*bytes*) –

Returns The number of bytes written

Return type int

Notes

See *IsoFile.writeFile()* for exceptions this function might raise.

CHAPTER 3

DolFile

class gciso.**DolFile**(*data*)

Represents a DOL executable file. You probably don't want to instance this class yourself, but rather call *IsoFile.getDolFile()*. See here for some more information about the attributes of this class: http://hitmen.c02.at/files/yagcd/yagcd/chap14.html#sec14.2 The DOL file is loaded into memory by the Apploader, section by section. This may or may not include permutation or fragmentation (the sections stay contiguous, but there may be gaps between the sections after being loaded.)

Parameters **data** (bytes or *IsoInternalFileWrapper*) – The file to be interpreted as a DOL file. See description of this class.

data
bytes – The DOL file as bytes

bssMemAddress
int

bssSize
int

entryPoint
int

bodyOffset
int – Offset to the the data after the header of the DOL

textSections
list of *Section* – The text sections of the DOL

dataSections
list of *Section* – The data sections of the DOL

sections
list of *Section* – Just a joined list of *DolFile.textSections* and *DolFile.dataSections*

sectionsDolOrder
list of *Section* – *DolFile.sections* but sorted by DOL offset

sectionsMemOrder
list of *Section* – *DolFile.sections* but sorted by memory address

class Section(*index*, *sectionType*, *dolOffset*, *memAddress*, *size*)
A section in a DOL file. You probably never want to instantiate this class yourself.

Parameters

- **index** (*int*) – The index of the section
- **sectionType** (*DolFile.SectionType*) – The type of the section
- **dolOffset** (*int*) –
- **memAddress** (*int*) –
- **size** (*int*) –

index
int

type
DolFile.SectionType

dolOffset
int

endDolOffset
int

memAddress
int

endMemAddress
int

size
int

isBefore(*other*)

class SectionType
The type of section in the DOL file.

DATA = 'data'

TEXT = 'text'

dolOffsetToMemAddress(*dolOffset*)

Parameters **dolOffset** (*int*) – An offset inside the DOL file.

Returns The memory address the data pointed to by dolOffset is loaded to if it belongs to a DOL section. *None* otherwise.

Return type int or None

getSectionByDolOffset(*dolOffset*)

Parameters **dolOffset** (*int*) – A offset inside the DOL file

Returns The section *dolOffset* points to or *None* if that offset does not point to a DOL section.

Return type *Section* or None

getSectionByMemAddress(*memAddress*)

Parameters **memAddress** (*int*) – Memory address

Returns The section the memory address points to or *None* if that address does not point to a DOL section.

Return type *Section* or None

isMappedContiguous(*dolOffsetStart*, *dolOffsetEnd*)

This function determines whether a range of (contiguous) memory in the DOL file is loaded contiguously to memory.

Parameters

- **dolOffsetStart** (*int*) – The start of the data range inside the DOL
- **dolOffsetEnd** (*int*) – The end of the data range (non-inclusive).

Returns

Return type bool

Notes

dolOffsetEnd not being inclusive means that if isContiguous(0, 4) is True, then the byte at offset 4 not be 4 bytes after the byte at offset 0 in memory. (Only byte 0, 1, 2 and 3 are).

isMappedContiguousMem(*memAddressStart*, *memAddressEnd*)

See *isMappedContiguous()*, but starting with memory. This essentially just maps the memory addresses to DOL offsets and then calls *isMappedContiguous()*.

memAddressToDolOffset(*memAddress*)

Parameters **memAddress** (*int*) – Memory address

Returns The offset inside the DOL of the data that is loaded to *memAddress* if it belongs to a DOL section. *None* otherwise.

Return type int or None

CHAPTER 4

BannerFile

class gciso.**BannerFile**(*data*)

Represents a .bnr file. Mostly there is just one *opening.bnr* in an .iso. In PAL .isos there are multiple. You probably don't want to instance this class yourself, but rather call *IsoFile.getBannerFile()*.

Parameters **data** (bytes or *IsoInternalFileWrapper*) – The file to interpret as a banner file. See the description of this class.

magicBytes

bytes

pixelData

bytes – The pixel data of the image in RGB5A1 format.

meta

MetaData or list of *MetaData* – For PAL isos meta may be a list with multiple *MetaData* objects

class MetaData(*data*, *offset=0*)

Contains metadata of a banner. See here for more information about the fields: http://hitmen.c02.at/files/yagcd/yagcd/chap14.html#sec14.1

gameName

bytes

developerName

bytes

fullGameTitle

bytes

fullDeveloperName

bytes

gameDescription

bytes

getPILImage()

PIL will be imported lazily by this function. So PIL or Pillow is only a requirement if you use this function.

Returns

Return type `PIL.Image`

CHAPTER 5

Indices and tables

- genindex

Python Module Index

g

Index

N

O

P

R

S

T

V

W