GameX

An Asset Guide to Video Games

Sky Morey

Families

The following are the game tables per family

Arkane Family

Arkane Studios SASU is a French video game developer based in Lyon. It was founded in 1999, and released its first game, Arx Fatalis, in 2002. The studio has created the popular Dishonored series as well as developing Prey (2017), Deathloop (2021) and Redfall (2023). Marvel's Blade is under development on 2027.

Besides Arkane Lyon, Arkane Studios also operated Arkane Studios LLC (traded as Arkane Austin) in Austin, Texas, from July 2006 until its closure in May 2024. [Arkane]

References

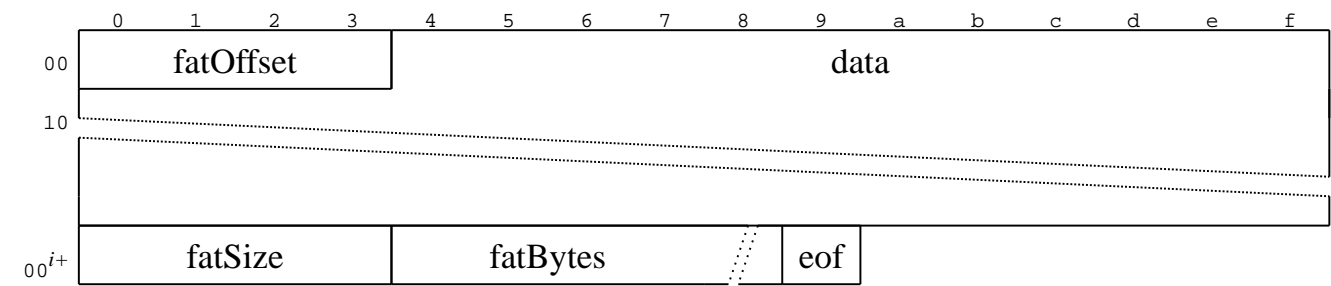- [Arkane]^*Wikipedia*. "Arkane Studios" (2025, December 13). https://en.wikipedia.org/w/index.php?title=Arkane_Studios&oldid=1327245775.

Binary Formats

Binary: Danae

The file format for Danae

ID	Type	Field	Description
HDR			**Header**
	uint:4	fatOffset	Fat Table Offset
FAT	seek: fatOffset		**Fat Table**
	uint:4	fatSize	Fat Size
	bytes:+	fatBytes	Fat Bytes

To deconstruct the fat table:

```
# while there are bytes
while b < fatSize:
    dirPath = readString()
    numFiles = readInt32()
    for _ in range(0, numFiles):
        # get file
        file = FileSource(
            path = dirPath + readString(),
            position = readInt32(),
            compressed = readInt32(),
```

```
                fileSize = readInt32(),
                packedSize = readInt32())
            # special case
            if file.path.endswith('.FTL'): file.compressed = 1
            elif file.compressed == 0: file.fileSize = file.packedSize
            # add file
            files.append(file)
```

To access a FILE

```
r.seek(file.position)
if (file.compressed & 1) != 0: decompressBlast(r, file.packedSize, file.fileSize)
else: r.read(file.packedSize)
```

Source code

```
# Binary_Danae
class Binary_Danae(ArcBinaryT):
    # read
    def read(self, source: BinaryArchive, r: Reader, tag: object = None) -> None:
        source.files = files = []
        key = source.game.key; keyLength = len(key); keyIndex = 0

        # move to fat table
        r.seek(r.readUInt32())
        fatSize = r.readUInt32()
        fatBytes = bytearray(r.readBytes(fatSize)); b = 0

        # read int32
        def readInt32() -> int:
            nonlocal b, keyIndex
            p = b
            fatBytes[p + 0] = fatBytes[p + 0] ^ key[keyIndex]; keyIndex += 1
            if keyIndex >= keyLength: keyIndex = 0
            fatBytes[p + 1] = fatBytes[p + 1] ^ key[keyIndex]; keyIndex += 1
            if keyIndex >= keyLength: keyIndex = 0
            fatBytes[p + 2] = fatBytes[p + 2] ^ key[keyIndex]; keyIndex += 1
            if keyIndex >= keyLength: keyIndex = 0
            fatBytes[p + 3] = fatBytes[p + 3] ^ key[keyIndex]; keyIndex += 1
            if keyIndex >= keyLength: keyIndex = 0
            b += 4
            return int.from_bytes(fatBytes[p:p+4], 'little', signed=True)

        # read string
        def readString() -> str:
            nonlocal b, keyIndex
            p = b
            while True:
                fatBytes[p] = fatBytes[p] ^ key[keyIndex]; keyIndex += 1
                if keyIndex >= keyLength: keyIndex = 0
                if fatBytes[p] == 0: break
                p += 1
            length = p - b
            r = fatBytes[b:p].decode('ascii', 'replace')
            b = p + 1
            return r

        # while there are bytes
        while b < fatSize:
            dirPath = readString().replace('\\', '/')
            numFiles = readInt32()
            for _ in range(numFiles):
                # get file
                file = FileSource(
                    path = dirPath + readString().replace('\\', '/'),
```

```
                    offset = readInt32(),
                    compressed = readInt32(),
                    fileSize = readInt32(),
                    packedSize = readInt32())
                # special case
                if file.path.endswith('.FTL'): file.compressed = 1
                elif file.compressed == 0: file.fileSize = file.packedSize
                # add file
                files.append(file)

    # readData
    def readData(self, source: BinaryArchive, r: Reader, file: FileSource, option: object = None) -> BytesIO:
        r.seek(file.offset)
        return BytesIO(
            decompressBlast(r, file.packedSize, file.fileSize) if (file.compressed & 1) != 0 else \
            r.readBytes(file.packedSize))
```
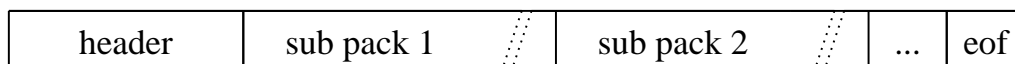
Binary: Void

The file format for Danae

master.index will include other index files as subpaks

ID	Type	Field	Description
HDR			**Header**
	uint:4	magic	assert: 0x04534552
	uint:4	unknown1	Unknown
PAK	repeat: until type == End		**Sub Paks**
	uint:4	type	Sub = 0x18000000, End = 0x18000000
	uint:4	nameSize	optional: type == Sub, Size of Name
	ascii:nameSize	path	The path
	ushort:2	packId	optional: not the first, Pack ID

header	sub pack 1	⫽	sub pack 2	⫽	...	eof

The normal .index files then use this file format

ID	Type	Field	Description
HDR			**Header**
	uint:4	unknown1	Unknown field.
	uint:4	mainFileSize	The Main file size.
	byte:24	unknown2	Unknown bytes.
	uint:4	numFiles	The Number of files.

ID	Type	Field	Description	
FILE	repeat: numFiles		**File**	
	uint:4	id	File ID	
	ascii:l8	tag1	Tag1	
	ascii:l8	tag2	Tag2	
	ascii:l8	path	File path.	
	ulong:8	position	File position.	
	uint:4	fileSize	File size.	
	uint:4	packedSize	Packed size.	
	uint:4	unknown1	Unknown	
	uint:4	flags	Flags	
	ushort:2	flags2	Flags2	

Source code

```python
# Binary_Void
class Binary_Void(ArcBinaryT):

    #region Headers

    class V_File:
        _struct = ('>Q4IH', 26)
        def __init__(self, tuple):
            self.offset, \
            self.fileSize, \
            self.packedSize, \
            self.unknown1, \
            self.flags, \
            self.flags2 = tuple

    #endregion

    # read
    def read(self, source: BinaryArchive, r: Reader, tag: object = None) -> None:
        # must be .index file
        if _pathExtension(source.filePath) != '.index': raise Exception('must be a .index file')

        files = source.files = []

        # master.index file
        if source.filePath == 'master.index':
            MAGIC = 0x04534552
            SubMarker = 0x18000000
            EndMarker = 0x01000000

            magic = r.readUInt32E()
            if magic != MAGIC: raise Exception('BAD MAGIC')
            r.skip(4)
            first = True
            while True:
                pathSize = r.readUInt32()
                if pathSize == SubMarker: first = False; pathSize = r.readUInt32()
                elif pathSize == EndMarker: break
                path = r.readFAString(pathSize).replace('\\', '/')
                packId = 0 if first else r.readUInt16()
                if not path.endswith('.index'): continue
```

```python
                files.append(FileSource(
                    path = path,
                    arc = self.SubArchive(self, None, source, source.game, source.fileSystem, path)))
            return

        # find files
        fileSystem = source.fileSystem
        resourcePath = f'{source.filePath[:-6]}.resources'
        if not fileSystem.fileExists(resourcePath): raise Exception('Unable to find resources extension')
        sharedResourcePath = next((x for x in ['shared_2_3.sharedrsc',
            'shared_2_3_4.sharedrsc',
            'shared_1_2_3.sharedrsc',
            'shared_1_2_3_4.sharedrsc'] if fileSystem.fileExists(x)), None)

        # read
        r.seek(4)
        mainFileSize = r.readUInt32E()
        r.skip(24)
        numFiles = r.readUInt32E()
        files = source.files = []
        for _ in range(numFiles):
            id = r.readUInt32E()
            tag1 = r.readL32Encoding()
            tag2 = r.readL32Encoding()
            path = (r.readL32Encoding() or '').replace('\\', '/')
            file = r.readS(self.V_File)
            useSharedResources = (file.flags & 0x20) != 0 and file.flags2 == 0x8000
            if useSharedResources and not sharedResourcePath: raise Exception('sharedResourcePath not available')
            newPath = sharedResourcePath if useSharedResources else resourcePath
            files.append(FileSource(
                id = id,
                path = path,
                compressed = 1 if file.fileSize != file.packedSize else 0,
                fileSize = file.fileSize,
                packedSize = file.packedSize,
                offset = file.offset,
                tag = (newPath, tag1, tag2)))

    # readData
    def readData(self, source: BinaryArchive, r: Reader, file: FileSource, option: object = None) -> BytesIO:
        pass
```

Family Info

Arkane
name: Arkane Studios
studio: Arkane Studios
description: The file formats used in the Void engine

List of Engines

ID	Name
Danae	Danae
Void	Void
Source	Source

List of Games

ID	Name	Date	Exts	Urls
AF	Arx Fatalis Engine: Danae	Nov 12, 2002	.arc	

asc:AVQF3FCKE50GRIAYXJP2AMEYO5QGA0JGIIH2NHBTVOA1VOGGU5H3GSSIARKP
RQPQKKYEOIAQG1XRX0J4F5OEAEFI4DD3LL45VJTVOA1VOGGUKE50GRIAYX

Files	Value
Gog	1207658680
WinReg	GOG.com/Games/1207658680
Local	Arx Fatalis

ID	Name	Date	Exts	Urls
DOM	Dark Messiah of Might and Magic Engine: Source:07x	Oct 25, 2006	dir.vpk	

Files	Value
Steam	2100
WinReg	Microsoft/Windows/CurrentVersion/Uninstall/Steam App 2100
Local	Dark Messiah Might and Magic Single Player
vpks	

ID	Name	Date	Exts	Urls
KS	KarmaStar	Jan 01, 2009	-	-

ID	Name	Date	Exts	Urls
D	Dishonored Engine: Unreal	Oct 12, 2012	.upk	

Files	Value
Gog	1701063787
WinReg	GOG.com/Games/1701063787
Local	Dishonored
DishonoredGame/CookedPCConsole	

ID	Name	Date	Exts	Urls
D2	Dishonored 2 Engine: Void	Nov 11, 2016	.index	

Files	Value
Gog	1431426311
WinReg	GOG.com/Games/1431426311
Local	Dishonored 2
base	

ID	Name	Date	Exts	Urls
P	Prey Engine: CryEngine	May 04, 2017	.arc	

Files	Value
Gog	1158493447
WinReg	GOG.com/Games/1158493447
Local	Prey
GameSDK	

ID	Name	Date	Exts	Urls
D:DOTO	Dishonored: Death of the Outsider Engine: Void	Sep 15, 2017	.index	

Files	Value
Gog	1707860700
WinReg	GOG.com/Games/1707860700
Local	Dishonored Death of the Outsider
base	

ID	Name	Date	Exts	Urls
W:YB	Wolfenstein: Youngblood Engine: idTech:6	Jul 25, 2019	.xyz	

Files	Value
Steam	1056960
WinReg	Microsoft/Windows/CurrentVersion/Uninstall/Steam App 1056960
Local	
base	

ID	Name	Date	Exts	Urls
W:CP	Wolfenstein: Cyberpilot Engine: idTech:6	Jul 25, 2019	.xyz	

Files	Value
Steam	1056970
WinReg	Microsoft/Windows/CurrentVersion/Uninstall/Steam App 1056970
Local	Wolfenstein Cyberpilot
base	

ID	Name	Date	Exts	Urls
DL	Deathloop Engine: Void	Jul 25, 2019	.index	

Files	Value
Steam	1252330
WinReg	Microsoft/Windows/CurrentVersion/Uninstall/Steam App 1252330
Local	DEATHLOOP
base	

ID	Name	Date	Exts	Urls
RF	Redfall (future)	-	-	