OpenDeathValley Documentation

Release 1.0

*

Dec 10, 2022

Table of Contents

CHAPTER 1

Game information

CHAPTER 2

Game Data

2.1 CPG

Files with a cpg extension are simple ini-files which describe the game's campaign as a set of levels, each having it's own section:

```
[Level Name]
Key1 = Value1
Key2 = Value 2
...

[Level Name 2]
...
```

Possible keys and their meaning:

Key	Meaning
title	the title of the level
level	file name of the level (a file from .\Data\Levels without the file extension)
cinematic	file name of a bik video which is played before the level starts
outro	file name of a bik video which is played after finishing the level
credits	file name of a bik video for the credits
infogrames	file name of a bik video for the infogrames cinematic
spellbound	file name of a bik video for the spellbound cinematic
map	a map? (a file from .\Data\Interfaces\Maps)
Xposition	X position on the map?
Yposition	Y position on the map?

2.2 DVD

2.2.1 AI

2.2.2 BGND

Contains the minimap.

Specification

Structure

```
struct file_header
struct sbpicture
```

File Header

```
+0x00                 :     VERSION             [DWORD]
+0x00                 :     SIZE_FILENAME       [WORD]
+0x02                 :     FILENAME            [BYTE] * SIZE_FILENAME
```

- Version must be equal to 0x04

SBPICTURE

See *SBPicture*

2.2.3 BOND

Contains a list of coordinates?

Structure

```
struct bond_header
for (bond_header.nb_entry) {
    struct bond_entries
}
```

Specification

```
+0x00 :       VERSION       [DWORD]
+0x04 :       NB_ENTRY      [WORD]
```

Entries

```
+0x00 :      X_1           [WORD]
+0x02 :      Y_1           [WORD]
+0x04 :      X_2           [WORD]
+0x06 :      Y_2           [WORD]
+0x08 :      UNK_DWORD_00  [WORD]
+0x0A :      UNK_DWORD_01  [WORD]
+0x0C :      UNK_DWORD_02  [WORD]
```

- Version is not tested

Example (Level_01.dvd)

2.2.4 BUIL

Contains data about buildings, doors, trapdoors etc.

Structure

```
struct BUIL_header
for (BUIL.nb_building) {
    struct BUILDING_entry
    for (BUILDING_entry.nb_door) {
        struct DOOR_entry
        struct DOOR_FRONT_entry
        struct DOOR_ON_entry
        struct DOOR_IN_entry
    }
}
word nb_entry
for (nb_entry) {
    struct unknow_entry
}
```

Specification

Header

```
+0x00 :   VERSION         [DWORD]
+0x04 :   NB_BUILDING     [WORD]
+0x06 :   BUILDING_ENTRY  [...]
+ ... :   NB_ENTRY        [WORD]
+ ... :   UNKNOW_ENTRY    [...]
```

Building Entry

```
+0x00                :    UNK_WORD_00     [WORD]
+0x02                :    UNK_WORD_01     [WORD] /* NB OF SOMETHING */
+0x04                :    ARRAY_NS        [WORD] * UNK_WORD_01
+0x04 + UNK_WORD_01:    NB_DOOR         [WORD]
```

DOOR Entry

```
+0x00                       :     UNK_BYTE_00               [BYTE] /* Type ? : 0x01 DOOR,
↪ 0x02, TRAPDOOR ? */
+0x01                       :     UNK_BYTE_01               [BYTE]
+0x02                       :     UNK_BYTE_02               [BYTE]
+0x03                       :     UNK_BYTE_03               [BYTE]
+0x04                       :     UNK_BYTE_04               [BYTE]
+0x05                       :     UNK_BYTE_05               [BYTE]
+0x06                       :     UNK_BYTE_06               [BYTE]
+0x07                       :     UNK_BYTE_07               [BYTE]
+0x08                       :     UNK_BYTE_08               [BYTE]
+0x09                       :     UNK_BYTE_09               [BYTE]
+0x0A                       :     NB_COORDINATE             [WORD]
+0x0C + NB_COORDINATE * 2:     [Y, X]                    [WORD] * NB_COORDINATE * 2
```

DOOR FRONT Entry

```
+0x00 :     ID_POINTS                 [WORD]
+0x02 :     Y                         [WORD]
+0x04 :     X                         [WORD]
+0x06 :     UNK_WORD_00               [WORD]
+0x08 :     UNK_WORD_01               [WORD]
```

DOOR ON Entry

```
+0x00 :     Y                         [WORD]
+0x02 :     X                         [WORD]
+0x04 :     UNK_WORD_00               [WORD]
+0x06 :     UNK_WORD_01               [WORD]
```

DOOR INSIDE Entry

```
+0x00 :     Y                         [WORD]
+0x02 :     X                         [WORD]
+0x04 :     UNK_WORD_00               [WORD]
+0x06 :     UNK_WORD_01               [WORD]
```

Example (Level_01.dvd)

2.2.5 CART

2.2.6 DLGS

Contains dialog, objectives and debriefing text and sound.

Structure

```
struct dlgs_header
for (dlgs_header.nb_entry) {
    struct text_wave_entries
}
struct dlgs_tricks
struct dlgs_objectives
struct dlgs_debriefing
```

Specifications

```
+0x00 :   VERSION       [DWORD]
+0x04 :   INDEX_TEXT    [DWORD]
+0x08 :   INDEX_WAVE    [DWORD]
+0x0C :   NB_ENTRY      [DWORD]
+0x10 :   TEXT_WAVE_ENTRIES [TEXT_WAVE_ENTRIES] * NB_ENTRY
```

- Version must be equal to 0x04
- Index Text is an Index that you can find in "Texts.res" *RES*, it's a list of text dialog
- Index Wave is an Index that you can find in "Texts.res" *RES*, it's a list of wave file path
- The nb of entry of the index inside "Texts.res" and NB_ENTRY must be equal!

TEXT_WAVE_ENTRIES

```
+0x00 :   UNK_DWORD_00   [DWORD]
+0x04 :   UNK_DWORD_00   [DWORD]
+0x08 :   UNK_DWORD_00   [DWORD]
```

TODO

"Trick"

The file is then followed by:

```
+0x00 :   INDEX          [DWORD]
+0x04 :   NB_ENTRY       [DWORD]
+0x08 :   ID             [DWORD] * NB_ENTRY
```

- Index can be find in "Texts.res"

"Objectives"

The file is then followed by:

```
+0x00 :   INDEX          [DWORD]
+0x04 :   NB_ENTRY       [DWORD]
+0x08 :   ID             [DWORD] * NB_ENTRY
```

- Index can be find in "Texts.res"

"Debriefing"

The file is then followed by:

```
+0x00 :   INDEX          [DWORD]
+0x04 :   NB_ENTRY       [DWORD]
+0x08 :   ID             [DWORD] * NB_ENTRY
+0xXX :   NB_ENTRY_2     [DWORD]
+0xXX :   ID_2           [DWORD] * NB_ENTRY_2
```

- Index can be find in "Texts.res"

2.2.7 ELEM

General

...

IDA VA: 0x4ACD50

Specifications

File Header

```
+0x00:   VERSION    [DWORD]
+0x04:   NB_ELEME   [WORD]
```

- Version must be equal to 0x1C

Type Object

Warning: Those values are just gues!!

- 0x00: Character
- 0x01: Environment object (river, smoke, ...)
- 0x02: Object (Accessories, etc ...)

Elem Header

```
+0x00:   TYPE       [WORD]
```

There is 41 (0x29) different types. But some of them seems similar (are parsed in the same way).

Struct sub_492650

```
+0x00:    Struct DVFRelation [sizeof (Struct DVFRelation)]   // value = 0x1
+0x00:    Struct sub_58B8C0 [sizeof (Struct sub_58B8C0)]   // value = 0x1
+0x00:    UNK_BYTE_00   [BYTE]    // Related to Y coordinate
+0x00:    UNK_BYTE_01   [BYTE]    // Related to X coordinate
+0x00:    UNK_BYTE_02   [BYTE]    // Related to X coordinate
```

Struct sub_58AFF0

```
+0x00:    Struct DVFRelation [sizeof (Struct DVFRelation)]
+0x00:    Struct sub_58B8C0 [sizeof (Struct sub_58B8C0)]
```

Struct DVFRelation

- Information for relationship with [[DVF_File_Format|DVF]] file

```
+0x00                                                                    :     LENGTH_
↪DVF_FILENAME   [WORD]
+0x02                                                                    :     DVF_
↪FILENAME          [BYTE] * LENGTH_DVF_FILENAME
+0x02 + LENGTH_DVF_FILENAME                                              :     LENGTH_
↪OBJECT_NAME    [WORD]
+0x04 + LENGTH_DVF_FILENAME                                              :     OBJECT_
↪NAME           [BYTE] * LENGTH_OBJECT_NAME
if value == 0 or value == 2
+0x04 + LENGTH_DVF_FILENAME + LENGTH_OBJECT_NAME                         :     ALT_
↪PROFILE_PRESENT   [BYTE]
    if ALT_PROFILE_PRESENT == 1
+0x05 + LENGTH_DVF_FILENAME + LENGTH_OBJECT_NAME                         :     LENGTH_
↪DVF_FILENAME_2 [WORD]
+0x07 + LENGTH_DVF_FILENAME + LENGTH_OBJECT_NAME                         :     UNK_BUF_
↪NAME          [BYTE] * LENGTH_DVF_FILENAME_2
+0x07 + LENGTH_DVF_FILENAME + LENGTH_OBJECT_NAME + LENGTH_DVF_FILENAME_2:     LENGTH_
↪OBJECT_NAME_2  [WORD]
+0x09 + LENGTH_DVF_FILENAME + LENGTH_OBJECT_NAME                         :     UNK_2_
↪BUF_NAME        [BYTE] * LENGTH_OBJECT_NAME_2
    end
end
```

- If ALT_PROFILE_PRESENT is set to TRUE, it will open an altprofile DVF file

Struct sub_58B8C0

if value == 0x01

```
if value == 0x01
+0x00:          UNK_WORD_00       [WORD]
+0x02:          UNK_WORD_01       [WORD]
+0x04:          HEIGHT            [WORD]    // from top to bottom
end
```

Computed info come from the associated DVF fileinfo [[DVF_File_Format#Object|DVF object info]]:

```
POINT_X = UNK_WORD_00 + UNKNOWN0 (FROM DVF)
POINT_Y = UNK_WORD_01 + UNKNOWN1 (FROM DVF) + HEIGHT
```

if value == 0x00

```
if value == 0x00
+0x00:          PATHFINDER_INDEX                [BYTE]
+0x01:          UNK_WORD_01                     [WORD]    // Y TOP LEFT coordinate BOX ?
+0x03:          UNK_WORD_02                     [WORD]    // X TOP LEFT coordinate BOX ?
+0x05:          UNK_WORD_03                     [WORD]    // Y DOWN RIGHT coordinate␣
↪BOX ?
+0x07:          UNK_WORD_04                     [WORD]    // X TOP LEFT coordinate BOX ?
+0x09:          PATHFINDER_ALTERNATE_INDEX      [BYTE]
+0x0A:          UNK_WORD_05                     [WORD]    // Y TOP LEFT coordinate BOX ?
+0x0C:          UNK_WORD_06                     [WORD]    // X TOP LEFT coordinate BOX ?
+0x0E:          UNK_WORD_07                     [WORD]    // Y DOWN RIGHT coordinate␣
↪BOX ?
+0x10:          UNK_WORD_08                     [WORD]    // X TOP LEFT coordinate BOX ?
+0x12:          UNK_WORD_09                     [WORD]
+0x14:          UNK_WORD_10                     [WORD]
+0x16:          UNK_WORD_11                     [WORD]
+0x18:          UNK_WORD_12                     [WORD]
+0x1A:          UNK_WORD_13                     [WORD]
+0x1C:          UNK_BYTE_00                     [BYTE]
+0x1D:          UNK_BYTE_01                     [BYTE]
end
```

if value == 0x02

```
if value == 0x02
TODO
end
```

Type 0x0001 || 0x0002 || 0x0003 || 0x0004 || 0x0005 || 0x0006

- Read method virtual address: 0x0048A7C0

```
+0x00:   Struct sub_58AFF0   [sizeof (Struct sub_58AFF0)] // value == 0x00
+0x00:   UNK_WORD_00         [WORD]
+0x00:   UNK_WORD_01         [WORD]
```

- 0x0001: John Cooper

- 0x0002: Doc Mc Coy
- 0x0003: Sam
- 0x0004: Kate O'Hara
- 0x0005: Pablo Sanchez
- 0x0006: Mia Jung

Type 0x0007 || 0x0210 || 0x0211 || 0x0212 || 0x0213 || 0x0214 || 0x0215

- "'DVElementActorAnimal"'
- Read method virtual address: 0x00463E30

```
+0x00:   Struct sub_58AFF0   [sizeof (Struct sub_58AFF0)] // value == 0x00
+0x00:   LENGTH_CLASSNAME    [WORD]
+0x00:   CLASSNAME           [BYTE] * LENGTH_CLASSNAME
+0x00:   IGNORED_WORD        [WORD]
```

Type 0x0101

- "'DVElementActorNPC"'
- Read method virtual address: 0x0047EAA0
- Second virtual method: 0x00438BA0

```
+0x00:   Struct sub_58AFF0   [sizeof (Struct sub_58AFF0)] // value == 0x00
+0x00:   LENGTH_CLASSNAME    [WORD]
+0x00:   CLASSNAME           [BYTE] * LENGTH_CLASSNAME
+0x00:   UNK_DWORD_00        [DWORD]
+0x00:   UNK_WORD_00         [WORD]
+0x00:   UNK_BYTE_00         [BYTE]
+0x00:   UNK_BYTE_01         [BYTE]
+0x00:   UNK_WORD_01         [WORD]
+0x00:   UNK_WORD_02         [WORD]
```

Type 0x0102

- "'DVElementActorNPC"'
- Read method virtual address: 0x0047EAA0
- Second virtual method: 0x00406760

```
+0x00:   Struct sub_58AFF0   [sizeof (Struct sub_58AFF0)] // value == 0x00
+0x00:   LENGTH_CLASSNAME    [WORD]
+0x00:   CLASSNAME           [BYTE] * LENGTH_CLASSNAME
+0x00:   UNK_WORD_00         [WORD]
+0x00:   UNK_BYTE_00         [BYTE]
+0x00:   UNK_BYTE_01         [BYTE]
+0x00:   UNK_WORD_01         [WORD]
+0x00:   UNK_WORD_02         [WORD]
```

Type 0x0201

- "'DVElementActorHorse"'
- Read method virtual address: 0x00467DA0

```
+0x00:   Struct sub_58AFF0   [sizeof (Struct sub_58AFF0)] // value == 0x00
+0x00:   LENGTH_CLASSNAME    [WORD]
+0x00:   CLASSNAME           [BYTE] * LENGTH_CLASSNAME
```

Type 0x0800

- "'DVElementTarget"'
- Read method virtual address: 0x004A9110

```
+0x00:   Struct sub_492650 [sizeof (Struct sub_492650)]
+0x00:   UNK_WORD_00       [WORD]
+0x00:   UNK_WORD_01       [WORD]
+0x00:   UNK_WORD_02       [WORD]
+0x00:   UNK_WORD_03       [WORD]
+0x00:   UNK_WORD_04       [WORD]
+0x00:   UNK_DWORD_00      [DWORD]
+0x00:   LENGTH_CLASSNAME  [WORD]
+0x00:   CLASSNAME         [BYTE] * LENGTH_CLASSNAME
```

Type 0x1001

- "'DVElementFX"'
- Read method virtual address: 0x00492650

```
+0x00:   Struct sub_492650
```

Example (Level_01.dvd)

```
type = 0x1001
[+] name (DVF FileName) = Level01_Acrobate
[+] name (Object Name) = Acrobate
[+] val == 1: unk_word_00 = 0x0216
[+] val == 1: unk_word_01 = 0x0189
[+] val == 1: unk_word_02 = 0x0005
```

```
[+] unk_word_00  = 0x0001
[+] unk_word_01  = 0x0001
[+] width        = 0x001E
[+] height       = 0x0061
[+] unk_dword_00 = 0x000000A0
[+] unk_dword_01 = 0x00000078
```

- Y = 0x0216 + 0x000000A0 = 0x2B6
- X = 0x0189 + 0x00000078 + 0x0005 = 0x206

Type 0x0301 || 0x1101 || 0x1102 || 0x1103 || 0x1106 || 0x1105 || 0x1104 || 0x1107 || 0x1108 || 0x1109 || 0x110D || 0x110E || 0x110B || 0x110A || 0x1110 || 0x1111 || 0x1112 || 0x1113 || 0x1114 || 0x1115 || 0x1116 || 0x1117

- "'DVElementObject"'
- Read method virtual address: 0x004A0970

```
+0x00:   Struct sub_58AFF0   [sizeof (Struct sub_58AFF0)] // value == 0x02
+0x00:   UNK_WORD_00         [WORD]
```

Type 0x110C

- "'DVElementObject"'
- Read method virtual address: 0x004A0970

```
+0x00:   Struct sub_58AFF0   [sizeof (Struct sub_58AFF0)] // value == 0x02
+0x00:   UNK_WORD_00         [WORD]
+0x00:   UNK_WORD_01         [WORD]
+0x00:   UNK_WORD_02         [WORD]
```

extract_elm_info.rb script

Result:

```
ba => block_address
so => size_object
c_va => constructor virtual address
v_va => vtable virtual address
rd => read method virtual address
v => value (type of object)
v = 0x00000001 ; rd: 0x0048A7C0 ; ba: 0x004ACE86 ; so: 0x02D0 ; c_va: 0x0044A180 ; v_
↪va: 0x00652648 ;
v = 0x00000002 ; rd: 0x0048A7C0 ; ba: 0x004ACEBD ; so: 0x02D4 ; c_va: 0x00453EA0 ; v_
↪va: 0x006527E8 ;
v = 0x00000003 ; rd: 0x0048A7C0 ; ba: 0x004ACEF4 ; so: 0x02D4 ; c_va: 0x0059D350 ; v_
↪va: 0x00654394 ;
v = 0x00000004 ; rd: 0x0048A7C0 ; ba: 0x004ACF37 ; so: 0x02D0 ; c_va: 0x0050EED0 ; v_
↪va: 0x006539C4 ;
v = 0x00000005 ; rd: 0x0048A7C0 ; ba: 0x004ACF6E ; so: 0x02D0 ; c_va: 0x00553D90 ; v_
↪va: 0x006540D8 ;
v = 0x00000006 ; rd: 0x0048A7C0 ; ba: 0x004ACFA5 ; so: 0x02D8 ; c_va: 0x005114C0 ; v_
↪va: 0x00653ACC ;
v = 0x00000007 ; rd: 0x00463E30 ; ba: 0x004ACFDC ; so: 0x026C ; c_va: 0x00463830 ; v_
↪va: 0x00652A68 ;
v = 0x00000101 ; rd: 0x0047EAA0 ; ba: 0x004AD00F ; so: 0x0AF0 ; c_va: 0x004858B0 ; v_
↪va: 0x00653124 ;
v = 0x00000201 ; rd: 0x00467DA0 ; ba: 0x004AD094 ; so: 0x029C ; c_va: 0x00467860 ; v_
↪va: 0x00652B20 ;
v = 0x00000102 ; rd: 0x0047EAA0 ; ba: 0x004AD0C5 ; so: 0x0878 ; c_va: 0x00485590 ; v_
↪va: 0x00652F90 ;
v = 0x00000210 ; rd: 0x00463E30 ; ba: 0x004AD0F8 ; so: 0x026C ; c_va: 0x00463830 ; v_
↪va: 0x00652A68 ;
v = 0x00000211 ; rd: 0x00463E30 ; ba: 0x004AD136 ; so: 0x026C ; c_va: 0x00463830 ; v_
↪va: 0x00652A68 ;
```

(continues on next page)

(continued from previous page)

```
v = 0x00000212 ; rd: 0x00463E30 ; ba: 0x004AD169 ; so: 0x026C ; c_va: 0x00463830 ; v_
↪va: 0x00652A68 ;
v = 0x00000301 ; rd: 0x004A0970 ; ba: 0x004AD1BB ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00000213 ; rd: 0x00463E30 ; ba: 0x004AD1EC ; so: 0x026C ; c_va: 0x00463830 ; v_
↪va: 0x00652A68 ;
v = 0x00000214 ; rd: 0x00463E30 ; ba: 0x004AD21F ; so: 0x026C ; c_va: 0x00463830 ; v_
↪va: 0x00652A68 ;
v = 0x00000215 ; rd: 0x00463E30 ; ba: 0x004AD252 ; so: 0x026C ; c_va: 0x00463830 ; v_
↪va: 0x00652A68 ;
v = 0x00000800 ; rd: 0x004A9110 ; ba: 0x004AD285 ; so: 0x0148 ; c_va: 0x004A8630 ; v_
↪va: 0x006535CC ;
v = 0x00001001 ; rd: 0x00492650 ; ba: 0x004AD2E3 ; so: 0x00B8 ; c_va: 0x00490EF0 ; v_
↪va: 0x00653368 ;
v = 0x00001101 ; rd: 0x004A0970 ; ba: 0x004AD31D ; so: 0x01B4 ; c_va: 0x004A2D90 ; v_
↪va: 0x00653530 ;
v = 0x00001102 ; rd: 0x004A0970 ; ba: 0x004AD362 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001103 ; rd: 0x004A0970 ; ba: 0x004AD39C ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001106 ; rd: 0x004A0970 ; ba: 0x004AD3E7 ; so: 0x01B4 ; c_va: 0x004A2D90 ; v_
↪va: 0x00653530 ;
v = 0x00001105 ; rd: 0x004A0970 ; ba: 0x004AD421 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001104 ; rd: 0x004A0970 ; ba: 0x004AD45B ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001107 ; rd: 0x004A0970 ; ba: 0x004AD495 ; so: 0x01B4 ; c_va: 0x004A2D90 ; v_
↪va: 0x00653530 ;
v = 0x00001108 ; rd: 0x004A0970 ; ba: 0x004AD4F5 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001109 ; rd: 0x004A0970 ; ba: 0x004AD52F ; so: 0x01B4 ; c_va: 0x004A2D90 ; v_
↪va: 0x00653530 ;
v = 0x0000110D ; rd: 0x004A0970 ; ba: 0x004AD569 ; so: 0x01B4 ; c_va: 0x004A2D90 ; v_
↪va: 0x00653530 ;
v = 0x0000110E ; rd: 0x004A0970 ; ba: 0x004AD5A3 ; so: 0x01B4 ; c_va: 0x004A2D90 ; v_
↪va: 0x00653530 ;
v = 0x0000110B ; rd: 0x004A0970 ; ba: 0x004AD5DD ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x0000110C ; rd: 0x004A0970 ; ba: 0x004AD617 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x0000110A ; rd: 0x004A0970 ; ba: 0x004AD651 ; so: 0x01B4 ; c_va: 0x004A2D90 ; v_
↪va: 0x00653530 ;
v = 0x00001110 ; rd: 0x004A0970 ; ba: 0x004AD68B ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001111 ; rd: 0x004A0970 ; ba: 0x004AD6C5 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001112 ; rd: 0x004A0970 ; ba: 0x004AD6FF ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001113 ; rd: 0x004A0970 ; ba: 0x004AD739 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001114 ; rd: 0x004A0970 ; ba: 0x004AD773 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001115 ; rd: 0x004A0970 ; ba: 0x004AD7AD ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
v = 0x00001116 ; rd: 0x004A0970 ; ba: 0x004AD7E7 ; so: 0x012C ; c_va: 0x0049A7D0 ; v_
↪va: 0x006534C0 ;
```

2.2.8 FXBK

General

Load FXs, Sounds, Exclamations files

- VA: 0x57FF70

Specifications

.code-block:: text

    struct file_header struct FXs struct Sounds struct Exclamations

File Header

.code-block:: text

    +0x00: VERSION [DWORD]

- Version must be equal to 0x03.

It's followed by a list of FXs, Sounds and Exclamations to load

struct FXs

.code-block:: text

    +0x00: NB_FXS [WORD] +0x02: ID_FXS [WORD] * NB_FXS

ID is an entry in file `\Data\Sounds\desperados.fxg`.

struct Sounds

.code-block:: text

    +0x00: NB_SOUNDS [DWORD] +0x04: ID_SOUND [DWORD] * NB_SOUNDS

ID is an entry in file `\Data\Sounds\desperados.sbk`.

struct Exclamations

.code-block:: text

    +0x00: NB_XET [WORD] +0x02: ID_XET [WORD] * NB_XET +0xXX: NB_XCT [WORD] +0xXX: ID_XCT [WORD] * NB_XCT +0xXX: NB_XPT [WORD] +0xXX: ID_XPT [WORD] * NB_XPT

`XET%02d.dat`, `XPT%02d.dat`, `XCT%02d.dat` stored in `\Data\Sounds\Expressions`

2.2.9 JUMP

General

Jump coordinate informations:

If you are standing on a roof or balcony of a house and you see a horse directly underneath, you can jump into the saddle and ride away.

VA: 0x004EFF80

Specifications

```
struct file_header
for (file_header.nb_jumps) {
    Struct JUMP
    if (JUMP.unk_dword_cc > 1) {
        for (JUMP.unk_dword_cc - 1) {
            STRUCT_INFO_ROOF_BALCONY
        }
    }
}
struct Sounds
struct Exclamations
```

File Header

```
+0x00:   VERSION    [DWORD]
+0x04:   NB_JUMPS   [WORD]
```

- Version must be equal to 0x01.

Struct JUMP

```
+0x00:   unk_dword_aa               [WORD]
+0x02:   unk_dword_bb               [WORD]
+0x04:   unk_dword_cc               [WORD]
+0x06:   pos_y                      [WORD]
+0x08:   pos_x                      [WORD]
+0x0A:   unk_dword_ff               [WORD]
+0x0C:   unk_dword_gg               [WORD]
+0x0E:   unk_dword_hh               [WORD]
+0x10:   STRUCT_INFO_ROOF_BALCONY   [] * (unk_dword_cc - 1)
+0xXX:   unk_dword_ii               [WORD]
+0xXX:   unk_dword_jj               [WORD]
+0xXX:   STRUCT_COORDS              []
```

STRUCT_INFO_ROOF_BALCONY

```
+0x00:   pos_y                    [WORD]
+0x02:   pos_x                    [WORD]
+0x04:   unk_dword_cc             [WORD]
+0x06:   unk_dword_dd             [WORD]
+0x08:   unk_dword_ee             [WORD]
```

STRUCT_COORDS

```
+0x00:   nb                       [WORD]
+0x02:   STRUCT_COORD             [] * nb
```

STRUCT_COORD

```
+0x00:   y                        [WORD]
+0x02:   x                        [WORD]
```

Example (Level_02.dvd)

2.2.10 LIFT

General

Lift coordinate.

VA: 0x4E1920

Specifications

File Header

```
+0x00:   VERSION    [DWORD]
+0x04:   NB_LIFTS   [WORD]
```

- Version must be equal to 0x02.

Struct LIFT

```
+0x00:   sector_id                [WORD]
+0x02:   type                     [BYTE]
+0x03:   unk_word_01              [WORD]
+0x05:   unk_word_02              [WORD]
+0x07:   unk_word_03              [WORD]
+0x09:   unk_word_04              [WORD]
+0x0B:   STRUCT_COORDS            []
+0xXX:   StructCOORD              [0x06]
+0xXX:   unk_word_04              [WORD]
```

STRUCT_COORDS is present when it is a wall that you can climb.

StructCOORD is the position of the character during the climbing of wall, ladder, stair, etc ...

type

- 0x00: ????
- 0x01: stair
- 0x02: ladder
- 0x03: wall

STRUCT_COORDS

```
+0x00:   nb                          [WORD]
+0x02:   STRUCT_COORD                [] * nb
```

STRUCT_COORD

```
+0x00:   y                           [WORD]
+0x02:   x                           [WORD]
```

Example (Level_01.dvd)

2.2.11 MASK

2.2.12 MAT

Defines the material for an area?

Structure

```
struct mat_header
for (mat_header.nb_entry) {
    struct Entries_00
    for (Entries_00.nb_entry) {
        byte type
        if type == 0x08 {
            struct Entries_01_type_1
        }
        else {
            struct Entries_02_type_2
        }
    }
}
```

Specification

```
+0x00 :      VERSION      [DWORD]
+0x04 :      NB_ENTRY     [DWORD]
```

- Version must be equal to 0x04

Entries_00

```
+0x00 :      UNK_WORD_00    [WORD]
+0x02 :      NB_ENTRY       [WORD]
```

Entries_01

```
+0x00 :      NS_TYPE            [BYTE]
if NS_TYPE == 0x08:
    +0x01 :      UNK_WORD_00    [WORD]
    +0x03 :      NB_COORD       [WORD]
    +0x05 :      [Y, X]         [WORD] * NB_COORD
else:
    +0x01 :      UNK_DWORD_00   [DWORD]
    +0x05 :      UNK_DWORD_01   [DWORD]
    +0x09 :      UNK_BYTE_00    [BYTE]
    +0x0A :      UNK_DWORD_02   [DWORD]
    +0x0E :      NB_COORD       [WORD]
    +0x10 :      [Y, X]         [WORD] * NB_COORD
```

2.2.13 MISC

2.2.14 MOVE

2.2.15 MSIC

General

Contains filename of “Quiet”, “alert”, “fight” music loop

VA: 0x0057FF20

Specifications

```
struct file_header
List <Struct MSIC Quiet>
List <Struct MSIC Alert>
List <Struct MSIC Fight>
```

File Header

```
+0x00:   VERSION    [DWORD]
```

- Version must be equal to 0x01.

Struct MSIC Quiet

```
+0x00:   NB_ELEMS       [WORD]
+0x02:   PASCAL_STRING  [] * NB_ELEMS
```

Struct MSIC Alert

```
+0x00:   NB_ELEMS       [WORD]
+0x02:   PASCAL_STRING  [] * NB_ELEMS
```

Struct MSIC Fight

```
+0x00:   NB_ELEMS       [WORD]
+0x02:   PASCAL_STRING  [] * NB_ELEMS
```

PASCAL_STRING

```
+0x00:   LENGTH_STR     [WORD]
+0x02:   STR            [BYTE] * LENGTH_STR
```

Example (Level_01.dvd)

```
Container:
    version = 1
    StructMSICQuiet = Container:
        nb_elems = 1
        name = [
            'green01.wav'
        ]
    StructMSICAlert = Container:
        nb_elems = 1
        name = [
            'orange02.wav'
        ]
    StructMSICFight = Container:
        nb_elems = 1
        name = [
            'red00.wav'
        ]
```

2.2.16 PAT

2.2.17 SCRP

..[[DVD_File_Format#Type_Signature|SCRP]] (script) entries in the [[DVD File Format]].

Contains scripts for a level?

Specification

Header

```
+0x00 :   VERSION       [DWORD]
+0x04 :   NB_ENTRY      [WORD]
```

- Version must be equal to 0x01

Entry

Depending of the first WORD value read, entry are totally different.

If first DWORD equal 0x01:

```
+0x00 :   FIRST_WORD   [WORD] // Equal to 1 in this case
+0x02 :   UNK_WORD_01  [WORD]
+0x04 :   UNK_WORD_02  [WORD]
+0x06 :   UNK_WORD_03  [WORD]
+0x08 :   UNK_WORD_04  [WORD]
+0x0A :   UNK_BYTE_00  [BYTE]
```

If first WORD is different of 0x02:

```
+0x00                       :   FIRST_WORD / NB_COORDINATES [WORD]
+0x02                       :   Y / X                       [(WORD, WORD)] * NB_
↪COORDINATES
+0x02 + NB_COORDINATES * 4 :   UNK_WORD_01                  [WORD]
+0x04 + NB_COORDINATES * 4 :   UNK_WORD_02                  [WORD]
+0x06 + NB_COORDINATES * 4 :   CLASSNAME_PRESENT            [BYTE]
+0x07 + NB_COORDINATES * 4 :   CLASSNAME_LENGTH             [WORD] // If CLASSNAME_
↪PRESENT equal to 0x01
+0x09 + NB_COORDINATES * 4 :   CLASSNAME                    [BYTE] * CLASSNAME_LENGTH
```

Example (Level1.DVD)

```
[+] nb coordinate = 0x0006
[(920, 806), (900, 826), (922, 842), (883, 872), (846, 809), (882, 785)]
[+] unk_word_01 = 0x0000
[+] unk_word_02 = 0x0000
[+] classname_present = 0x01
[+] classename = "Zone_Boxeur_Ko_54709ec"
```

“Zone_Boxeur_Ko” means “boxer area”.

If we draw pink crosshair on those coordinate, they are near the boxer area.

Looks like those coordinate are here to "delimit" script execution.

2.2.18 SGHT

2.2.19 SND

2.2.20 WAYS

General

Specifications

```
struct ways_header
for ways_header.nb_entry {
    short nb_waypoints
    for nb_waypoints {
        struct waypoint
    }
}
```

Struct WAYS

```
+0x00:    VERSION      [DWORD]
+0x04:    NB_ENTRY     [WORD]
```

- Version must be equal to 0x01

Struct waypoint

```
+0x00:    Y_COORD              [WORD]
+0x02:    X_COORD              [WORD]
+0x04:    UNK_WORD_00          [WORD]
+0x06:    UNK_WORD_01          [WORD]
+0x08:    CLASS_NAME_PRESENT   [BYTE]
+0x0A:    LENGTH_DATA          [WORD]
+0x0C:    DATA                 [BYTE] * LENGTH_DATA
```

- If CLASS_NAME_PRESENT is equal t0 0x01, DATA is a classname that we can find in the associate .scb file

2.2.21 General

The DVD files contain data about a level. For every DVD file, there is a corresponding *DVM*, *SCB*, *STF* file with the same name.

2.2.22 Specification

The file contains a list of entries.

Type Entry

```
+ 0x00:    ENTRY_TYPE                  [DWORD]
+ 0x04:    SIZE                        [DWORD]
```

Type Signature

- 0x20204941 / ' IA' => *AI*
- 0x444E4742 / 'DNGB' => *BGND*
- 0x444E4F42 / 'DNOB' => *BOND*
- 0x4C495542 / 'LIUB' => *BUIL*
- 0x54524143 / 'TRAC' => *CART*
- 0x53474C44 / 'SGLD' => *DLGS*
- 0x4D454C45 / 'MELE' => *ELEM*
- 0x4B425846 / 'KBXF' => *FXBK*
- 0x504D554A / 'PMUJ' => *JUMP*
- 0x5446494C / 'TFIL' => *LIFT*
- 0x4B53414D / 'KSAM' => *MASK*
- 0x2054414D / ' TAM' => *MAT*
- 0x4353494D / 'CSIM' => *MISC*
- 0x45564F4D / 'EVOM' => *MOVE*
- 0x4349534D / 'CISM' => *MSIC*
- 0x20544150 / ' TAP' => *PAT*
- 0x50524353 / 'PRCS' => *SCRP*
- 0x54484753 / 'THGS' => *SGHT*
- 0x20444E53 / ' DNS' => *SND*
- 0x53594157 / 'SYAW' => *WAYS*

2.3 DVF

This type of file contains sprites and animation data.

2.3.1 File list

- `.\Game\Data\Animations\*`
- `.\Game\Data\Characters\*`

2.3.2 Specifications

Structure

```
struct file_header
for (file_header.nb_sprites) {
    struct sprite_header
    data   sprite
}
struct profiles_header
for (profiles_header.nb_profiles) {
    struct profile
    for (profile.nb_animations * profile.nb_perspective) {
            struct animation
            for (animation.nb_frames) {
                struct frame
            }
        }
    }
}
```

File Header

```
+ 0x00:    VERSION     [WORD]
+ 0x02:    NB_SPRITES  [WORD]      // number of sprites
+ 0x04:    PADDING     [WORD]      // NOT USED
+ 0x06:    MAX_WIDTH   [WORD]      // max width of all sprites
+ 0x08:    MAX_HEIGHT  [WORD]      // max height of all sprites
+ 0x0a:    PADDING     [BYTE] * 20 // NOT USED
```

- Size of `Header` : *0x1E*
- `VERSION` must be equal to *0x200*

Sprite Header

```
+ 0x00:    SIZE        [DWORD]  // offset to next sprite
+ 0x04:    WIDTH       [WORD]   // width of the sprite (pixel)
+ 0x06:    HEIGHT      [WORD]   // height of the sprite (pixel)
+ 0x08:    PADDING     [WORD]   // NOT USED
+ 0x0a:    SPRITE_DATA [BYTE] * SIZE
```

> **Warning:** The size of each row must be rounded up to a multiple of 4 bytes (a 32-bit DWORD) by padding

Sprite Data

For every row (height) in the sprite:

```
+ 0x00:    NB_TRANSPARENT_PIXELS  [WORD]
+ 0x02:    NB_TOTAL_PIXELS        [WORD]
```

If NB_TOTAL_PIXELS (int16_t) is -1, the whole line is transparent (color for transparent is *0x7C0* or *0x1F*). If NB_TOTAL_PIXELS > -1, fill the first NB_TRANSPARENT_PIXELS pixels with transparent. Read 2 byte long pixels (R5G6B5) for NB_TOTAL_PIXELS times. Fill the remaining pixels (according to the sprite header's width) with transparent.

Profiles Header

```
+ 0x00:    NB_PROFILES [WORD]   // number of profiles
```

Profile

```
+ 0x00:    NAME              [BYTE] * 32  // profile name
+ 0x20:    NB_PERSPECTIVES   [WORD]       // number of perspectives the profile can␣
↪be shown in
+ 0x22:    PADDING           [BYTE] * 32
+ 0x42:    NB_ANIMATIONS     [WORD]       // number of animations available for the␣
↪profile
+ 0x44:    PADDING           [BYTE] * 16
+ 0x54:    MAX_WIDTH         [WORD]
+ 0x56:    MAX_HEIGHT        [WORD]
+ 0x58:    COORDINATE_X      [DWORD]
+ 0x5C:    COORDINATE_Y      [DWORD]
+ 0x60:    PADDING           [BYTE] * 20
```

- Size of `Profile` : *0x74*

Animation

```
+ 0x00:    PADDING           [BYTE] * 4
+ 0x04:    NB_FRAMES         [WORD]
+ 0x06:    UNKNOWN0          [WORD]  // latest frame?
+ 0x08:    PADDING           [WORD]
+ 0x0a:    COORDINATE_X      [DWORD]
+ 0x0e:    COORDINATE_Y      [DWORD]
+ 0x12:    PERSPECTIVE_ID    [WORD]
+ 0x14:    ANIMATION_ID      [WORD]
+ 0x16:    ANIMATION_NAME    [BYTE] * 32
```

- Size of `Animation` : *0x36*

UNKNOWN0, COORDINATE_X, COORDINATE_Y don't seem to have any impact.

Warning: First byte of ANIMATION_NAME seems to always be *0x20* (SPACE) and can be ignored (old storage of the max length?). Please note that animations are referenced by their *ID* not their name.

Frame

```
+ 0x00:    SPRITE_ID         [WORD]
+ 0x02:    DURATION          [WORD]
+ 0x04:    DISTANCE          [WORD]
```

(continues on next page)

(continued from previous page)

```
+ 0x06:    COORDINATE_X      [WORD]
+ 0x08:    COORDINATE_Y      [WORD]
+ 0x0a:    SOUND_EFFECT      [WORD]
+ 0x0c:    PADDING           [WORD]
```

- Size of `Frame` : *0x0E*

`DURATION` is the time until the next frame is shown. This seems to be a multiple of 1/30 seconds.

`DISTANCE` is the distance the object covers while the animation displays the frame, 0 being the lowest distance, scaling upwards linear (might be a pixel unit or something arbitrary).

`COORDINATE_X` and `COORDINATE_Y` is the relative position between the object and the frame. Different frames from one animation have different sizes so to keep the object in the frames centered the position must be adjusted by `COORDINATE_X` in X direction and `COORDINATE_Y` in Y direction.

When the frame is displayed and the `SOUND_EFFECT` contains a valid sound effect ID (id is not null), the sound effect will be played. The sound can be found in the folder `data\\sounds\\fx_XXXX.wav`, where `XXXX` is the decimal value of `SOUND_EFFECT`.

2.3.3 Example

The file from the *demo* version `demo\data\animations\level00_rodeo.dvf` is used as an example.

- Extract all frames to BMP format in the output direcetory `/tmp/bmp_out/`:

```
$ mkdir /tmp/bmp_out
$ odv_files_test -t dvf -e -o /tmp/bmp_out/ /tmp/level00_rodeo.dvf
[+] odv_dvf_open = /tmp/level00_rodeo.dvf
$
```

- Display all extracted informations to the standard output:

```
$ odv_files_test -t dvf -i /tmp/bmp_out/ /tmp/level00_rodeo.dvf
[+] odv_dvf_open = /tmp/level00_rodeo.dvf
    [Profile #0 information]
    + name            : L00 Rodeo
    + nb_perspectives : 0x0001
    + nb_animations   : 0x0003
    + max_width       : 0x006D (109)
    + max_height      : 0x0066 (102)
    + coordinate_x    : 0.000000
    + coordinate_y    : 0.000000
        [Animation #0 information]
        + nb_frames      : 0x0001
        [...]
        [Animation #2 information]
        + nb_frames      : 0x0024
        + unk_word_00    : 0x0023
        + unk_word_01    : 0x0000
        + coordinate_x   : 0.000000
        + coordinate_y   : 0.000000
        + perspective_id : 0x0000
        + animation_id   : 0x00A1
        + animation_name : Ejection
        [...]
            [Frame #14 information]
```

(continues on next page)

(continued from previous page)

```
            + sprite_id    : 0x003C
            + duration     : 0x0001
            + distance     : 0x0000
            + coordinate_x : 86.000000
            + coordinate_y : 63.000000
            + sound_id     : 0x0599 (1433)
            + unused_00    : 0x0000
        [...]
$
```

Extracted BMP file associated with the *profile 0 (L00 Rodeo)*, *animation 2 (Ejection)*, *Frame 14*:

$$\text{Rodeo}_{E}jection_{0}014.bmp$$

Fig. 1: `L00 Rodeo_Ejection_0014.bmp`

The sound file (`sound_id`) associated with this frame (*Frame 14* of *animation 2 (Ejection)*) is `demo\data\sounds\fx_1433.wav`:

- Generate an animated GIF image for the *profile 0 (L00 Rodeo)*, *animation 1 (Rodeo 01)*:

```
$ convert -delay 1x60 -loop 0 'L00 Rodeo_Rodeo 01_*.bmp' Rodeo.gif
```

Fig. 2: `Rodeo.gif`

2.4 DVM

2.4.1 General

Files with extension `*.dvm` contains the map stored in R5G6B5 bzip2 compressed image.

2.4.2 Specifications

Structure

```
struct sbpicture
```

SBPICTURE

See *SBPicture*

Example (Level_01.dvm)

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  40 08 40 04 02 00 00 00 80 55 2C 00 42 5A 68 39  @.@......U,.BZh9
00000010  31 41 59 26 53 59 FC AE 63 97 05 A8 2F 7F FF FF  1AY&SY..c.../...
00000020  FF FF FE 7E 7C 7E 7C 7F FF FF FF FF 7F 7F 7F 7F  ...~|~|.........
```

Parse header and uncompressed data:

```
[+] width = 0x0840
[+] height = 0x0440
[+] type_picture = 0x00000002
[+] length = 0x002C5580
[+] length_u = 0x00462000
0000  ca a3 49 93 48 93 89 ab ca a3 8b bc cd b4 ce b4   ..I.H...........
0010  8c ac 8d b4 89 9b 08 83 4c a4 cb 9b 07 8b 85 7a   ........L......z
```

2.5 FNT

2.5.1 General

Files with extension `*.fnt` contains a list of wide character 16-bit (2 bytes) with their 2D associated representation stored in embedded pictures (normal and bold type).

2.5.2 Specifications

```
struct fnt_header
for fnt_header.nb_entry{
    struct char_entry
}
struct sbpicture
struct sbpicture
```

2.5.3 FNT Header

```
+0x00:    SIGNATURE     [BYTE] * 6
+0x06:    VERSION       [DWORD]
+0x0A:    FONT_NAME     [BYTE] * 36
+0x2E:    TYPE          [DWORD]
+0x32:    HEIGHT        [DWORD]
+0x36:    UNK_DWORD_00  [DWORD]    // Width rectangle letter
+0x3A:    UNK_DWORD_01  [DWORD]    // Maximum width letter
+0x3E:    NB_ENTRY      [DWORD]
if VERSION >= 0x200:
    +0x42:  UNK_DWORD_02  [DWORD]    // COORDINATE_Y + UNK_DWORD_05 ?
end
```

- `SIGNATURE` must be equal to "SBFONT"
- **`TYPE:`**
  - *0x00*: letter with a border between them
  - *0x02*: letter without a border between them

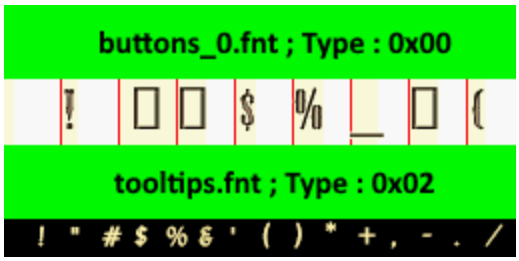

Fig. 3: Truncated extracted image extracted from `buttons_0.fnt` and `tooltips.fnt`

2.5.4 Char Entry

```
+0x00:    CHAR_VALUE     [WORD]
+0x02:    COORDINATE_Y   [DWORD]
+0x06:    WIDTH_LETTER   [DWORD]
+0x0A:    PRE_SPACING    [DWORD]
+0x0E:    POST_SPACING   [DWORD]
```

- The "real" witdh is computed: `WIDTH_LETTER` + `PRE_SPACING` + `POST_SPACING`

2.5.5 SBPICTURE

See *SBPicture*

Example (buttons_0.fnt and tooltips.fnt)

```
[+] fnt_filename = .\buttons_0.fnt
[+] first_unk_dword_00 = 0x00000200
[+] name = PlugCapsSSK
[+] unk_word_00 = 0000 (0)  // TYPE SOMETHING ?
[+] unk_word_01 = 0000 (0)
[+] unk_dword_02 = 00000021 (33)  // HEIGHT ?
[+] unk_dword_03 = 0000001C (28)  // Width square letter
[+] unk_dword_04 = 00000017 (23)  // Max width letter
0000  74 00 00 00                                        t...
[+] nb_entry = 0x00000074
[+] unk_dword_05 = FFFFFFFD (4294967293)
[+] num = 0000
[+] char_value = 0x0020 (" ")
[+] Y_COORD = 0x00000000
[+] WIDTH_LETTER = 0x00000005
[+] unk_dword_00 = 0x00000000
[+] unk_dword_01 = 0x00000002
[+] num = 0001
[+] char_value = 0x0021 ("!")
[+] Y_COORD = 0x0000001D
[+] WIDTH_LETTER = 0x00000008
[+] unk_dword_00 = 0x00000001
[+] unk_dword_01 = 0x00000001

--------------------
```

(continues on next page)

(continued from previous page)

```
[+] fnt_filename = .\tooltips.fnt
[+] first_unk_dword_00 = 0x00000200
[+] name = Lydian
[+] unk_word_00 = 0002 (2)  // TYPE SOMETHING ?
[+] unk_word_01 = 0000 (0)
[+] unk_dword_02 = 00000013 (19)  // HEIGHT ?
[+] unk_dword_03 = 0000000F (15)  // Width square letter
[+] unk_dword_04 = 0000000C (12)  // Max width letter
0000  a2 00 00 00                                          ....
[+] nb_entry = 0x000000A2
[+] unk_dword_05 = FFFFFFFE (4294967294)
[+] num = 0000
[+] char_value = 0x0020 (" ")
[+] Y_COORD = 0x00000000
[+] WIDTH_LETTER = 0x00000005
[+] unk_dword_00 = 0x00000000
[+] unk_dword_01 = 0x00000002
[+] num = 0001
[+] char_value = 0x0021 ("!")
[+] Y_COORD = 0x00000010
[+] WIDTH_LETTER = 0x00000006
[+] unk_dword_00 = 0x00000001
[+] unk_dword_01 = 0x00000001
```

The first image is of type 0x00 (we can guess type 0x00 == letter with a border between them)

The second image is of type 0x02 (we can guess type 0x02 == letter without a border between them)

2.6 FXG

2.6.1 General

Sound bank file

2.6.2 File List

- `.\Game\Data\Sounds\desperados.fxg`
- `.\Game\Data\Sounds\menu.fxg`

2.6.3 Specifications

File header

```
+0x00:   SIGNATURE    [DWORD]
+0x04:   UNK_DWORD_00 [DWORD]
+0x08:   UNK_DWORD_01 [DWORD]
+0x0C:   NB_ENTRY     [DWORD]
```

...

```
+0x00:   TYPE_BANK     [DWORD]
+0x04:   FX_ID         [DWORD]
+0x08:   UNK_WORD_00   [WORD]
```

- Signature must be equal to 0x4B425846

2.7 LOG

2.7.1 File list

- `.\Data\Configuration\debug.log`

2.7.2 Description

bzip2 archive containing exception information.

2.7.3 Example

```
00000000  42 5A 68 39 31 41 59 26 53 59 AC C9 E4 8E 00 00  BZh91AY&SY......
00000010  18 5F 80 00 12 60 65 4A 72 57 87 4D 84 3F E7 DE  ._...`eJrW.M.?..
00000020  E0 30 00 A6 C1 A9 A3 49 A9 EA 00 34 7A 8F 51 A3  .0.....I...4z.Q.
00000030  40 D1 E9 A9 8C 90 AA 79 A9 E5 4C 0D 35 30 00 9A  @......y..L.50..
00000040  60 00 09 88 A6 10 68 C8 03 40 00 00 00 24 0B CF  `.....h..@...$..
00000050  5E A2 57 98 C2 3A 0E 0B 4C 19 45 D7 AD 58 C9 5B  ^.W..:..L.E..X.[
00000060  B6 F2 93 5C 8D 2F 43 98 F7 FA ED CC 54 00 F6 20  ...\./C.....T..
00000070  2B 68 59 58 60 D5 3F C9 31 42 4A BE 04 FB 1E 78  +hYX`.?.1BJ....x
00000080  3B A5 51 60 B5 78 5A D9 9C E5 8D D4 95 20 57 36  ;.Q`.xZ...... W6
00000090  76 EF 8E 1D BA 4D B3 6B 75 99 20 9E 38 21 12 FB  v....M.ku. .8!..
000000A0  EA 4C 31 2A A4 26 62 24 9C 85 10 15 B0 18 D4 5D  .L1*.&b$.......]
000000B0  73 3C 06 29 81 D9 DE F9 02 5C 39 90 FE FF C5 B2  s<.).....\9.....
000000C0  49 C1 48 75 40 D0 A0 4E 2D A0 BD 09 06 1E D8 39  I.Hu@..N-......9
000000D0  78 2E E4 8A 70 A1 21 59 93 C9 1C                 x...p.!Y...
```

- 0x5A42 = ‘BZ’ signature/magic number
- 0x68 = version (‘h’ for Bzip2 (‘H’uffman coding), ‘0’ for Bzip1 (deprecated))
- 0x39 = ‘1’..’9’ block-size 100 kB-900 kB (uncompressed)
- … etc …

2.7.4 Extracted

```
================================================================
Fatal Exception in File P:\Death Valley\SBLibNG\SBDrawManager.cpp, Line 883:
The surface you try to delete does not exist (859)!
================================================================
Writing error information to E:\Game\Desperados\Game\crash000.dat
```

2.8 MAP

2.8.1 General

Files with extension `*.map` contains background map during the menu stored in R5G6B5 bzip2 compressed image.

2.8.2 File List

Filename	md5sum
data/interface/maps/louisiana.map	c32568e4871bc8875ada61ea02463552
data/interface/maps/new mexico.map	1929980087102120b2fe391d85331193
demo/data/interface/maps/demo_k.map	b5b46505480cd34ec8a32aab35f68494
localisation/russian/data/interface/maps/louisiana.map	bcbc490d35ba57bcc186127ad1754158
localisation/russian/data/interface/maps/new mexico.map	d2561061ffcd1d218785e984cd67a375
localisation_demo/french/data/interface/maps/notsure_demo_k.map	3d370ab0723e0ce55f1aa5b80d1e4a41
localisation_demo/german/data/interface/maps/notsure_demo_k.map	6aade25ee42efa97e32884a7a2264f28
localisation_demo/spanish/data/interface/maps/notsure_demo_k.map	c407ea3903c03a30339bae53f9ffb37c

2.8.3 Specifications

Structure

```
struct sbpicture
```

SBPICTURE

See *SBPicture*

2.8.4 Example

The file from the *demo* version `demo/data/interface/maps/demo_k.map` is used as an example.

- Extract *sbpciture* to BMP format in the output direcetory `/tmp/bmp_out/`:

```
$ mkdir /tmp/bmp_out
$ odv_files_test -t map -e -o /tmp/bmp_out/ /tmp/demo_k.map
[+] odv_dvf_open = /tmp/demo_k.map
$
```

- Display all extracted informations to the standard output:

```
$ ./bin/odv_files_test -t map -i /tmp/demo_k.map
[+] odv_map_open = /tmp/demo_k.map
[- ODV MAP information -]
[- ODV IMAGEMAP information -]
image_count : 0x00000001
[- ODV IMAGE information -]
width : 0x0400 (1024)
height: 0x0200 (512)
```

(continues on next page)

(continued from previous page)

```
type_compression: 0x00000002
data_size: 0x00100000
[-------------------------]
[-------------------------]
[---------------------------------]
```

Extracted BMP file:

Fig. 4: `demo_k_map.bmp` (converted to png)

2.9 PAK

2.9.1 General

Files with extension `*.pak` contains list of file stored in R5G6B5 bzip2 compressed image

2.9.2 File List

- `.\Game\Data\Interface\Loading.pak`

But binary contains string:

- `Data\Interface\WaitScreen.pak`
- `Data\Music\music.pak`
- `Data\Sounds\expressions.pak`
- `Data\Sounds\fxs.pak`
- `Data\Interface\Dialogs\dialogs.pak`

2.9.3 Specifications

```
while (data_availabe in file) {
    struct sbpicture
}
```

SBPICTURE

See *SBPicture*

2.10 RES

2.10.1 General

Files with extension `*.res` contains dialogs, buttons, cursors, etc... stored in different ways.

2.10.2 File list

Filename	md5sum
data/sounds/expressions/expressions.res	dc45e9538a6ac4683e2c66b071fe6fd7
demo/data/interface/default.res	864dba4dc53ddac85220917e2ebac55b
localisation/dutch/data/interface/default.res	64301d08b0070910b9a5f5f8a1f1ff06
localisation/dutch/data/interface/texts.res	896d15c6d65f135b53405074de0065aa
localisation/english/data/interface/default.res	3c18108ac7ba9576cd09d82c340a4fe6
localisation/english/data/interface/texts.res	db21118e7be568573a8fd377082daafe
localisation/french/data/interface/default.res	ff8bbdf8ac39f472e92d1392a405f375
localisation/french/data/interface/texts.res	c5debbfee1600aa8c907bedbf1956328
localisation/german/data/interface/default.res	e919510da87df778777dec8949221aff
localisation/german/data/interface/texts.res	8a26a38d24821ec6b9ac52b15b7f79b7
localisation/german/data/sounds/expressions/expressions.res	71a0b5c0dcb497725e20442717333b93
localisation/italian/data/interface/default.res	60243c5fb1cea1bf45353905d6be0b53
localisation/italian/data/interface/texts.res	0ca98c08de8ee12813b02bc75ca6e994
localisation/italian/data/sounds/expressions/expressions.res	dc45e9538a6ac4683e2c66b071fe6fd7
localisation/russian/data/interface/default.res	cf450d0dec042decbf5bdec481d65781
localisation/russian/data/interface/texts.res	5194c687f948f84abdc0a00206727670
localisation/russian/data/sounds/expressions/expressions.res	dc45e9538a6ac4683e2c66b071fe6fd7
localisation/spanish/data/interface/default.res	80d9111473d8495eb53aef3177e1b54e
localisation/spanish/data/interface/texts.res	b366f0cc3ad42403708a9ba48a864e22
localisation/spanish/data/sounds/expressions/expressions.res	dc45e9538a6ac4683e2c66b071fe6fd7
localisation_demo/french/data/interface/default.res	ea0851031814436d4c80f1bc681c00e8
localisation_demo/french/data/interface/texts.res	f3d43779f1c895ba0c84f11d6b12c44e
localisation_demo/german/data/interface/default.res	158e699a55746727f8b5cd7dda7df3df
localisation_demo/german/data/interface/texts.res	66c475f0f2b9e0ab2b734802dca44144
localisation_demo/spanish/data/interface/default.res	c4238b4eb99186d543451bdfe033dcea
localisation_demo/spanish/data/interface/texts.res	76905fad96f264a9d8e6947245477869

2.10.3 Texts.res

This file contains only 'TEXT' & 'WAVE'

2.10.4 Expressions.res

This file contains only 'WAVE'

2.10.5 DEFAULT.RES

This file contains 'SLID', 'TEXT', 'NPTF', 'BTTN', 'WAVE', 'TOGL', 'CUR ', 'RDO ', 'PIC ', 'PICC'

2.10.6 Specifications

Structure

```
struct file_header
for (file_header.nb_type_entry) {
    struct resource_type_entry
}
```

File Header

```
+ 0x00:    SIGNATURE                  [DWORD]
+ 0x04:    VERSION                    [DWORD]
+ 0x08:    NB_TYPE_ENTRY              [DWORD]
```

The header is followed by different type entries

- Size of `Header` : *0x0C*
- `SIGNATURE` must be equal to "SERS" (*0x53455253*)
- `VERSION` must be equal to *0x100*

Resource Type Entry

```
+ 0x00:    SIGNATURE_RESOURCE_TYPE    [DWORD]
+ 0x04:    ID_RESOURCE                [DWORD]
```

Resource Type Signature

Signature	Description	Type
0x45564157 ('EVAW')	Wave table resource	*'WAVE'*
0x43434950 ('CCIP')	Picture collection resource	*'PICC'*
0x44494C53 ('DILS')	Slider resource	*'SLID'*
0x20525543 (' RUC')	Mouse resource	*'CUR '*
0x20434950 (' CIP')	Picture resource	*'PIC '*
0x204F4452 (' ODR')	Radio resource	*'RDO '*
0x4E545442 ('NTTB')	Button resource	*'BTTN'*
0x4654504E ('FTPN')	Input field resource	*'NPTF'*
0x4C474F54 ('LGOT')	Toggle button resource	*'TOGL'*
0x54584554 ('TXET')	String table resource	*'TEXT'*

'WAVE'

Resource Header

```
+ 0x00:    PADDING_00       [DWORD] // unused
+ 0x04:    NB_ENTRY         [WORD]
```

Entries are path to *WAV* file in this format:

```
+ 0x00:    LENGTH           [WORD]
+ 0x02:    PATH_STR         [BYTE] * LENGTH
```

Example (Expressions.res)

```
00000000  53 52 45 53 00 01 00 00 33 00 00 00 57 41 56 45  SRES....3...WAVE
00000010  00 00 00 04 01 00 00 00 0C 00 14 00 5C 44 56 49  ............\DVI
00000020  5F 58 43 54 30 31 45 30 30 56 30 31 2E 77 61 76  _XCT01E00V01.wav
00000030  14 00 5C 44 56 49 5F 58 43 54 30 31 45 30 31 56  ..\DVI_XCT01E01V
```

- Signature Resource Type = 0x45564157
- Index = 0x4000000 (67108864)
- UNK_DWORD_00 = 0x01
- Number of entry = 0x000C (12)
- First Entry length = 0x0014 (20)
- First Entry path = "\DVI_XCT01E00V01.wav"
- Second Entry length = 0x0014 (20)
- Second Entry path = "\DVI_XCT01E01V……"

'PICC'

Resource Header

```
+ 0x00:    FLAG            [DWORD]
+ 0x04:    NB_ENTRY        [DWORD]
```

Entries are image (*SBPicture*) stored in RAW or ZLIB format

'SLID'

Resource Header

```
+ 0x00:    FLAG            [DWORD]
+ 0x04:    NB_ENTRY_BIT    [DWORD]
```

Nb entry of image is the number of bit (max 6 bits) to 1 stored in NB_ENTRY_BIT.

```
nbentry = 0;
for (i = 0; i < 6; i++) {
    if ((1 << i) & NB_ENTRY_BIT)
        nbentry++;
}
```

Entries are image (*SBPicture*) stored in RAW or ZLIB format

'CUR '

Contains all CURSOR

Resource Header

```
+ 0x00:    PADDING_00                      [DWORD]   // unused
+ 0x04:    TICK_COUNT_REDRAW               [WORD]
+ 0x06:    HIT_POINT_Y                     [WORD]
+ 0x08:    HIT_POINT_X                     [WORD]
+ 0x0A:    TICK_COUNT_REDRAW_PER_IMAGE     [WORD]   /* IF NB_ENTRY > 1 */
+ 0x0C:    NB_ENTRY                        [DWORD]
```

Entries are image (*SBPicture*) stored in RAW or ZLIB format

'PIC '

Resource Header

```
+ 0x00:    FLAG         [DWORD]
```

One entry is an image (*SBPicture*) stored in RAW or ZLIB format

'RDO '

Ressource for RADIO button img (it's used in Sound menu option for volume choice).

Resource Header

```
+ 0x00:    FLAG            [DWORD]
+ 0x04:    NB_ENTRY_BIT    [DWORD]
```

Nb entry of image is the number of bit (max 7 bits) to 1 stored in NB_ENTRY_BIT.

```
nbentry = 0;
for (i = 0; i < 7; i++) {
    if ((1 << i) & NB_ENTRY_BIT)
        nbentry++;
}
```

Entries are image (*SBPicture*) stored in RAW or ZLIB format

'BTTN'

Resource Header

```
+ 0x00:    FLAG            [DWORD]
+ 0x04:    NB_ENTRY_BIT    [DWORD]
```

Nb entry of image is the number of bit (max 4 bits) to 1 stored in NB_ENTRY_BIT.

```
nbentry = 0;
for (i = 0; i < 4; i++) {
    if ((1 << i) & NB_ENTRY_BIT)
        nbentry++;
}
```

Entries are image (*SBPicture*) stored in RAW or ZLIB format

'NPTF'

Note: Similar to 'SLID'

```
+ 0x00:    FLAG            [DWORD]
+ 0x04:    NB_ENTRY_BIT    [DWORD]
```

Nb entry of image is the number of bit (max 6 bits) to 1 stored in NB_ENTRY_BIT.

```
nbentry = 0;
for (i = 0; i < 6; i++) {
    if ((1 << i) & NB_ENTRY_BIT)
        nbentry++;
}
```

Entries are image (*SBPicture*) stored in RAW or ZLIB format

'TOGL'

Resource Header

```
+ 0x00:    FLAG            [DWORD]
+ 0x04:    NB_ENTRY_BIT    [DWORD]
```

Nb entry of image is the number of bit (max 5 bits) to 1 stored in NB_ENTRY_BIT.

```
nbentry = 0;
for (i = 0; i < 5; i++) {
    if ((1 << i) & NB_ENTRY_BIT)
        nbentry++;
}
```

Entries are image (*SBPicture*) stored in RAW or ZLIB format

'TEXT'

Resource Header

```
+ 0x00:    PADDING_00      [DWORD] // unused
+ 0x04:    NB_ENTRY        [WORD]
```

Entries are in-game dialog stored in wide character string:

```
+ 0x00:    LENGTH          [WORD]
+ 0x02:    PATH_STR        [BYTE] * LENGTH * 2
```

Example (Texts.res)

```
00000000  53 52 45 53 00 01 00 00 6F 00 00 00 54 45 58 54  SRES....o...TEXT
00000010  00 00 00 01 01 00 00 00 22 00 4C 00 51 00 75 00  ........".L.Q.u.
```

- SIGNATURE = 0x53455253
- Version = 0x100
- Number of TYPE entry = 0x0000006F (111)

Example (Texts.res)

```
00000000  53 52 45 53 00 01 00 00 6F 00 00 00 54 45 58 54  SRES....o...TEXT
00000010  00 00 00 01 01 00 00 00 22 00 4C 00 51 00 75 00  ........".L.Q.u.
00000020  27 00 65 00 73 00 74 00 2D 00 63 00 65 00 20 00  '.e.s.t.-.c.e. .
00000030  71 00 75 00 65 00 20 00 76 00 6F 00 75 00 73 00  q.u.e. .v.o.u.s.
00000040  20 00 66 00 61 00 69 00 74 00 65 00 73 00 20 00   .f.a.i.t.e.s. .
00000050  69 00 63 00 69 00 20 00 3F 00 20                 i.c.i. .?.
```

- Signature Resource Type = 0x54584554
- Index = 0x1000000 (16777216)

- UNK_DWORD_00 = 0x01
- Number of entry = 0x0022 (34)
- First Entry length = 0x004C (76)
- First Entry dialog = “Qu’est-ce que vous faites ici ? C’est ma ville ! Je suis arrive le premier !”
- . . . etc . . .

2.11 SBK

SBK files map .wav files to an ID.

2.11.1 Specification

The file contains a header followed by an arbitrary amount of entries.

Header

```
+ 0x00:    SIGNATURE       [DWORD]   /* 0x4b425344 / 'DSBK' */
+ 0x04:    VERSION?        [DWORD]   /* Must be 0x1 */
+ 0x08:    UNKNOWN         [DWORD]   /* 0x78 */
```

Entry

```
+ 0x00:    ID                [DWORD]
+ 0x04:    FILE_NAME_LENGTH  [WORD]
+ 0x06:    FILE_NAME         [BYTE] * FILE_NAME_LENGTH /* wav file in the same␣
↪directory as the sbk file */
```

2.12 SCB

2.12.1 File List

- .\Game\Data\Levels\Level_01.scb
- .\Game\Data\Levels\Level_02.scb
- .\Game\Data\Levels\Level_03.scb
- .\Game\Data\Levels\Level_04.scb
- .\Game\Data\Levels\Level_05.scb
- .\Game\Data\Levels\Level_06.scb
- .\Game\Data\Levels\Level_07.scb
- .\Game\Data\Levels\Level_08.scb
- .\Game\Data\Levels\Level_09.scb
- .\Game\Data\Levels\Level_10.scb

- .\Game\Data\Levels\Level_11.scb
- .\Game\Data\Levels\Level_12.scb
- .\Game\Data\Levels\Level_13.scb
- .\Game\Data\Levels\Level_14.scb
- .\Game\Data\Levels\Level_15.scb
- .\Game\Data\Levels\Level_16.scb
- .\Game\Data\Levels\Level_17.scb
- .\Game\Data\Levels\Level_18.scb
- .\Game\Data\Levels\Level_19.scb
- .\Game\Data\Levels\Level_20.scb
- .\Game\Data\Levels\Level_21.scb
- .\Game\Data\Levels\Level_22.scb
- .\Game\Data\Levels\Level_23.scb
- .\Game\Data\Levels\Level_24.scb
- .\Game\Data\Levels\Level_25.scb

2.12.2 Specifications

```
struct file_header
for (file_header.nbOfClasses) {
    struct class_header
    for (class_header.nboffunctions {
        struct function_header
    }
    data bytecode
}
```

sscanf format for parsing all header.

File Header

```
"version %f, debug %d\n"
"nbOfClasses %d\n"
```

- Version must be 1.0
- debug are set to 0 (FALSE)

Class Header

```
"fileName %1023s , className %1023s\n"
"nbOfVariables %d, sizeOfVariables %d\n"
"nbOfFunctions %d\n"
// Here we have all the functions header
```

(continues on next page)

(continued from previous page)

```
"nbOfQuads %d\n"
// Here we have all the bytecode of every function belonging to the class
```

Function Header

```
"functionName %1023s , address %d, nbOfParams %d, sizeOfRetVal %d, sizeOfParams %d\n"
"functionParameters\n"
"\n"
" sizeOfVolatile %d, sizeOfTempor %d\n"
```

ByteCode

In the game they call that "quad".

bytecode in file are stored in this way:

```
+ 0x00:    OPCODE    [BYTE]
+ 0x01:    OPERANDS  [QWORD]
+ 0x09:    PADDING   [BYTE]
```

2.12.3 Virtual Machine

IDA func: sub_618050

Operand flag

- 0x0000: UNKNOW ?
- 0x4000: CLASS ATTRIBUTES (class_var)
- 0x8000: VOLATILE VARIABLE (vol_var)
- 0xC000: TEMPORARY VARIABLE (temp_var)

ByteCode

Warning: /!bytecode in game memory are not stored in the same way as in the file /!

```
+ 0x00:    OPCODE    [DWORD]
+ 0x04:    OPERANDS  [QWORD]
```

Opcode

Max Opcode: 0x2C (44)

0x00

- Opcode: 0x00 (0)
- Nb of operands: 0

```
"VMCore::Empty: Operation not allowed.\n"
```

This opcode is not allowed and will set the VM PC to 0xFFFFFFFF

0x01

- Opcode: 0x01 (1)
- Nb of operands: 0

This opcode do nothing, it will only increment VM PC by 1

0x02

- Opcode: 0x02 (2)
- Nb of operands: 1
- Operand 0x01: Displacement (Size: WORD)

Copy variable from Function arguments / Volatile variable / temporary variable to field 0x0C of VM

0x03

- Opcode: 0x03 (3)
- Nb of operands: 2
- Operand 0x01: SIZE OF VOLATILE VARIABLE (Size: WORD)
- Operand 0x02: SIZE OF TEMPORARY VARIABLE (Size: WORD)

Entry point for every function in order to allocate enough place for the different variables

0x04

- Opcode: 0x04 (4)
- Nb of operands: 0

```
"VMCore::EndFunction: Operation not allowed (add a Return !!!).\n"
```

This opcode will increment VM PC by 1

0x05

- Opcode: 0x05 (5)
- Nb of operands: 1
- Operand 0x01: Address function (Size: DWORD)

TODO: Looks like a CALL to another function of the current class

0x06

- Opcode: 0x06 (6)

TODO

0x07

- Opcode: 0x07 (7)

TODO

0x08

- Opcode: 0x08 (8)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: DWORD)

```
MOV op1, op2
```

0x09

- Opcode: 0x09 (9)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: DWORD)

```
MOV op2, op1
```

0x0A

- Opcode: 0x0A (10)
- Nb of operands: 1
- Operand 0x01: Displacement (Size: WORD)

TODO

0x0B

- Opcode: 0x0B (11)
- Nb of operands: 1
- Operand 0x01: Displacement (Size: WORD)

Push argument for external CALL

```
PUSH op1
```

0x0C

- Opcode: 0x0C (12)
- Nb of operands: 0x01
- Operand 0x01: Index (Size: Immediate32)

This opcode will call a function stored in a table of address function at dword_6938A4. (see [[Script Function]])

```
.text:00613CB3 8B 46 40                mov     eax, [esi+40h]       // [esi+CVMScript.
↪VM_pc]     ; Actual PC
.text:00613CB6 8B 50 04                mov     edx, [eax+4]         // Operand 1
.text:00613CB9 A1 A4 38 69 00          mov     eax, dword_6938A4    // Table of␣
↪function (table initialized by sub_5E3BF0: size 0x320 (800) == 200 functions!)
.text:00613CBE 57                      push    edi
.text:00613CBF 8B 7E 3C                mov     edi, [esi+3Ch]
.text:00613CC2 8D 4E 48                lea     ecx, [esi+48h]        //␣
↪[esi+CVMScript.field_48] ; Function arguments
.text:00613CC5 FF 14 90                call    dword ptr [eax+edx*4] // Operand 1 is␣
↪used as an index
```

```
CALL [Index]
```

0x0D

- Opcode: 0x0D (13)
- Nb of operands: 1
- Operand 0x01: Displacement (Size: WORD)

Save return value from external function called before

0x0E

- Opcode: 0x0E (14)
- Nb of operands: 1
- Operand 0x01: Address (Size: DWORD)

Jmp to desired address

```
JMP Imm32
```

0x0F

- Opcode: 0x0F (15)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Address (Size: DWORD)

Conditional jump if value is TRUE.

```
TEST op1, op1 ; JZ Imm32
```

0x10

- Opcode: 0x10 (16)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Address (Size: DWORD)

Conditional jump if value is FALSE.

```
TEST op1, op1 ; JNZ Imm32
```

0x11

- Opcode: 0x11 (17)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)

mov from op2 to op1

```
mov op1, op2
```

0x12

- Opcode: 0x12 (18)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)

mov from op2 to op1

```
mov op1, op2
```

TODO: check why same as 0x11

0x13

- Opcode: 0x13 (19)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Immediate (Size: DWORD)

mov Immediate to op1

```
mov op1, Imm32
```

0x14

- Opcode: 0x14 (20)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Immediate (Size: DWORD)

mov Immediate to op1

```
mov op1, Imm32
```

TODO: check why same as 0x13

0x15

- Opcode: 0x15 (21)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)

```
sub op1, op2
```

0x16

- Opcode: 0x16 (22)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)

```
sub (float)op1, (float)op2
```

0x17

- Opcode: 0x17 (23)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)

```
mov (float)op1, (float)op2
```

0x18

- Opcode: 0x18 (24)
- Nb of operands: 2
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)

```
mov (float)op1, (double)op2
```

0x19

- Opcode: 0x19 (25)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 + op3)
```

0x1A

- Opcode: 0x1A (26)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 - op3)
```

0x1B

- Opcode: 0x1B (27)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 * op3)
```

0x1C

- Opcode: 0x1C (28)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 / op3).. code-block:: text
```

0x1D

- Opcode: 0x1D (29)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 + (float)op3)
```

0x1E

- Opcode: 0x1E (30)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 - (float)op3)
```

0x1F

- Opcode: 0x1F (31)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 * (float)op3)
```

0x20

- Opcode: 0x20 (32)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 / (float)op3)
```

0x21

- Opcode: 0x21 (33)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 <= op3)
```

0x22

- Opcode: 0x22 (34)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 < op3)
```

0x23

- Opcode: 0x23 (35)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 >= op3)
```

0x24

- Opcode: 0x24 (36)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 > op3)
```

0x25

- Opcode: 0x25 (37)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 != op3)
```

0x26

- Opcode: 0x26 (38)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov op1, (op2 == op3)
```

0x27

- Opcode: 0x27 (39)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 > (double)op3)
```

0x28

- Opcode: 0x28 (40)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 >= (double)op3)
```

0x29

- Opcode: 0x29 (41)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 < (double)op3)
```

0x2A

- Opcode: 0x2A (42)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 <= (double)op3)
```

0x2B

- Opcode: 0x2B (43)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 == (double)op3)
```

0x2C

- Opcode: 0x2C (44)
- Nb of operands: 3
- Operand 0x01: Displacement (Size: WORD)
- Operand 0x02: Displacement (Size: WORD)
- Operand 0x03: Displacement (Size: WORD)

```
mov (float)op1, ((float)op2 != (double)op3)
```

2.13 STF

2.14 SXT

2.14.1 General

Sprite/Image displayed on the screen when you win or loose a mission

2.14.2 File List

- `.\Game\Data\Interface\Debriefing\GameOver1.sxt`
- `.\Game\Data\Interface\Debriefing\GameOver2.sxt`
- `.\Game\Data\Interface\Debriefing\Victory.sxt`

2.14.3 Specifications

Structure

```
struct sbpicture
```

SBPICTURE

See *SBPicture*

Extension	Description	Link
.cpg	campaign configuration	*CPG*
.dvd	level data	*DVD*
.dvf	sprites and animation data	*DVF*
.dvm	level map	*DVM*
.fnt	font	*FNT*
.fxg	Sound bank	*FXG*
.log	exception information	*LOG*
.map		*MAP*
.pak		*PAK*
.res	resources	*RES*
.sbk	sound table	*SBK*
.scb	level scripts	*SCB*
.stf	link to a briefing file	*STF*
.sxt	sprite end mission	*SXT*
.tmp		
.dat		
.cfg		
.wav	wave sound file	

CHAPTER 3

Miscellaneous

3.1 SBPicture

All graphics in the game are stored in the R5G6B5 pixel format (5 bits for red, 6 bits for green and 5 bits for blue).

Two colors are used for transparency in the game:

- *0x07C0*: *R5G6B5(0, 62, 0)* / *R8G8B8(0, 248, 0)*
- *0x001F*: *R5G6B5(0, 0, 31)* / *R8G8B8(0, 0, 248)*

3.1.1 Specifications

```
struct picture_header
unsigned char pixel_data[picture_header.size_compressed]
```

PICTURE HEADER

```
+0x00:    WIDTH             [WORD]
+0x02:    HEIGHT            [WORD]
+0x04:    TYPE_COMPRESSION  [DWORD]
+0x08:    SIZE_COMPRESSED   [DWORD]
```

TYPE_COMPRESSION

- *0x00* : raw
- *0x01* : *zlib* compression
- *0x02* : *bz2* compression

3.1.2 Convertion (*R5G6B5* to *R8G8B8*)

```
unsigned short r5g6b5;
unsigned char red8, green8, blue8;

/* ... */

red8 = (r5g6b5 >> 11 << 8) / 32;
green8 = (((r5g6b5 >> 5) & 0x3F) << 8) / 64;
blue8 = ((r5g6b5 & 0x1F) << 8) / 32;
```