Ver. 1.2

Format plików gry Dispel

Pliki .GTL i .BTL:

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Size [Byte]** |
| Tile pixels (RGB16\_565) | short[] | 1k \* 2 |
| ... |  |  |

Tile:

0

1

2

3

4

5

6

7

8

...

Width = 64

Height = 32

pos = 0;

for (int y = 0; y < tile.Height; y++)

{

var n = y < tile.Height / 2 ? y : tile.Height - 1 - y;

var r = 1 + 2 \* n;

for (int x = tile.Width / 2 - r; x < tile.Width / 2 + r; x++)

{

var byte0 = pixels[pos++];

var byte1 = pixels[pos++];

int red = byte1 & 0xF8;

int green = byte1 << 5 & 0xFF | byte0 >> 3 & 0xFC;

int blue = byte0 << 3 & 0xFF;

int alpha = red + green + blue > 0 ? 0xFF : 0;

tile[x, y] = alpha << 24 | red << 16 | green << 8 | blue;

}

}

Pliki .MAP:

Mapy z kafelków tiles są zawsze w postaci rombu o rozmiarach 2 \* HexMapSize:

Castles: 124

Dungeons: 249

Maps: 499

Final: 199

2 \* HexMapSize

MapSize

GridOffset

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Size [Byte]** |
| (HexMap.Width + 1) / 25 (HexMap.Width = HexMapSize) | int | 4 |
| (HexMap.Height + 1) / 25 (HexMap.Height = HexMapSize) | int | 4 |
| Tiles props count TPC (= 2) | int | 4 |
| GTiles.Count | int | 4 |
| GTiles props block | int[] | GTiles.Count \* TPC \* 4 |
| BTiles.Count | int | 4 |
| BTiles props block | int | BTiles.Count \* TPC |
| Sprites.Count | int | 4 |
| for (int n = 0; n < Sprites.Count; n++)  sprite = Sprites.Add() | | |
| sprite.Stamp | int | 4 |
| sprite header | byte[] | 264 |
| Magic number (= 80) | int[2] | 2 \* 4 |
| sprite.Frames.Count | int | 4 |
| UNK | int | 4 |
| for (int i = 0; i < sprite.Frames.Count; i++)  frame = sprite.Frames.Add() | | |
| frame.Info | int | 6 \* 4 |
| Origin.X | int | 4 |
| Origin.Y | int | 4 |
| width | int | 4 |
| height | int | 4 |
| size | int | 4 |
| Frame pixels | short[] | width \* height \* 2 |
| endfor | | |
| UNK | byte[] | sprite.Stamp \* 364 - 280 |
| endfor | | |
| spritesInfoCount | int | 4 |
| for (int n = 0; n < spritesInfoCount; n++) | | |
| spriteID | int | 4 |
| UNK | int | 4 |
| UNK | int | 4 |
| Right | int | 4 |
| Bottom | int | 4 |
| Left | int | 4 |
| Top | int | 4 |
| sprite.Frames.Info block | int[] | (sprite.Frames.Count – 1) \* 6 \* 4 |
| endfor | | |
| tileObjects.Count | int | 4 |
| Block Count (=1) | int | 4 |
| for (int i = 0; i < tileObjects.Count; i++)  tileObject = tileObjects.Add() | | |
| tileObject header (=[0]) | byte[] | 260 |
| tileObject ID | int | 4 |
| Magic number (=8010) | int[] | 4 \* 4 |
| Left | int | 4 |
| Top | int | 4 |
| Right | int | 4 |
| Bottom | int | 4 |
| X (=Left) | int | 4 |
| Y (=Top) | int | 4 |
| X’ | int | 4 |
| Y’ | int | 4 |
| Counter1 | int | 4 |
| Counter2 | int | 4 |
| Counter3 | int | 4 |
| for (int n = 0; n < Counter3; n++) | | |
| tileObject.Cells[n].Piece.ImageIndex | short | 2 |
| endfor | | |
| UNK (=[0]) | byte[] | 84 |
| UNK (=[0,1]) | int[] | (Counter1 + Counter2 + Counter3) \* 4 |
| endfor | | |
| Event block   |  |  |  | | --- | --- | --- | | EventID | short | 2 | | EventFlags (=0, 0x40(64), 0x80(128), 0xC0(192), 0x140(320), 0x280(640), 0x380(896), 0xF40(3904)) | short | 2 | | short[] | (HexMapSize^2 – 1) \* 2 \* 2 |
| GTiles block | int[] | HexMapSize^2 \* 4 |
| Roof BTiles block | int[] | HexMapSize^2 \* 4 |