File: 5CCGuide-Tree-LibraryFormat.PDF
Format: PDF
First release: June 1st, 2008
Last updated: February 3rd, 2012
Author: Mafi; closecombat2@claranet.de; http://closecombat2.fortunecity.com/
Actual version: 2.00

Close Combat series of games

5CC – Guide

Tree-Library File Format

(a description for my CC map editor for PC- & Mac)

It became necessary

Goal: storing CC map cutouts into a library to make it easier to share such libraries among map makers and modders. CC map cutouts will contain both, graphics (none, 1 image, or perhaps up to three images) and as a minimum terrain data definitions arranged in a rectangle. In addition there should be some kind of hot-point defining the logical "center" of the cutout = the terrain data element which will be placed at the user's cursor position when putting the cutout back to a map in the map editor 5CC. The images must not be of identical size, and their size must not fit to the terrain datas rectangle. An image can be omitted as well. The logical hot-point between images and terrain data rectangle is the upper left corner of terrain data rectangle and images.

The intended file format should be logically compatible to 5CC's existing Preferences file format, which already contains some sort of (graphics free) tree terrain patch cutout storing.

Furthermore the tree-library format should be expandable for future extensions without becoming incompatible to older versions. This requires to store a version-ID, the number of entries in a "to be created" directory and informations about directory entry size, which might vary between different versions of this file format but must be indentical for all entries of one file.

To save memory and time on saving/loading, the image graphics should be stored as usual in CC series of games as 16-bit integer per pixel, stored uncompressed from left to right and from top to bottom similiar to TARGA file format. Accordingly furthermore to TARGA format the byte sequence should be Little Endian (like in CC3 or newer). To keep it simple, all other datas should be stored also in Little Endian.

So I will setup the following file format definition for 5CC's TREE-Library as v1.0:

```
// This is VERSION 1.0 of 5CC's Tree-Library file-format
// all data encoded in Little Endian
// header, always fixed size 16 bytes
String4     "EERT"     // 4 bytes, reverted "TREE" file type indicator
LongInt     1          // version-ID, 4 bytes
LongInt     295        // fixed size in bytes of each directory entry, 4 bytes
LongInt     iiii       // number of entries in directory, 4 bytes

// comment string
SignedInt   1000       // chapter-ID, 2 bytes, negative value means "read-only"
String255              // 255 bytes for author's remarks/instruction or copyright, ASCIIZ

// library menu color, 16-bit color info TARGA-like
SignedInt   6000       // chapter-ID, 2 bytes, negative value means "read-only"
LongInt     iiii       // number of color "pixels" to come, each "pixel" 2 bytes long
ShortInt    ii         // 2 bytes for 16-bit color information for menu display

// cover picture, always 200x200 pixels
SignedInt   7000       // chapter-ID, 2 bytes, negative value means "read-only"
LongInt     iiii       // cover pict width, counted in pixels
LongInt     iiii       // cover pict height, counted in pixels
Data                   // up to 80000 bytes cover image, 16-bit graphics, 2 bytes per pixel,
                       // picture should not be greater than 200x200 pixels.

// start of directory
SignedInt   8000       // directory-ID, 2 bytes, negative value means "read-only"
For every directory entry
            String255          // 255 bytes for an ASCIIZ string, name of cutout
            LongInt    iiii    // terrain cutout width, counted in terrain elements
            LongInt    iiii    // terrain cutout height, counted in terrain elements
            LongInt    iiii    // terrain cutout hot-point X, counted in terrain elements
            LongInt    iiii    // terrain cutout hot-point Y, counted in terrain elements
            LongInt    iiii    // terrain data offset, counted from top-of-file
            LongInt    iiii    // number of images, can be 0 or 1 for 5CC v1.08
            LongInt    iiii    // image width, counted in pixels
            LongInt    iiii    // image height, counted in pixels
            LongInt    iiii    // image opacity, value ranging between 0 and 100
            LongInt    iiii    // image data offset, counted from top-of-file
Next directory entry

// start of user data
SignedInt   9000       // user-data area-ID, 2 bytes, negative value means "read-only"
data-terrain 0              // terrain data, each data element a signed int = 2 bytes per element
data-image 0                // image data, 2 bytes per pixel
data-terrain 1
data image 1
data-terrain 2
data-image 2
etc. ...

// eof – no indicator, appending bytes will be ignored, may contain comments or copyright
```

Icon for a 5CC-Tree-Library file:  .
Filetype extension: ".5TL".

Next step in 2012 was the expansion of this file format to version 2.0, inluding the support of

- 16bit and 24bit graphics to store,
- varying scale of the terrain elements, from CC2-CCLSA's 10x10 pixel size to 16x16 pixel size,
- backward compatibility to v1.0.
- supported patch graphics size extended to 240x240 pixels.

```
// This is VERSION 2.0 of 5CC's Tree-Library file-format
// all data encoded in Little Endian
// header, always fixed size 16 bytes
String4     "EERT"     // 4 bytes, reverted "TREE" file type indicator
LongInt     2          // version-ID, 4 bytes  ---> changed
LongInt     307        // fixed size in bytes of each directory entry, 4 bytes  ---> changed
LongInt     iiii       // number of entries in directory, 4 bytes

// comment string
SignedInt   1000       // chapter-ID, 2 bytes, negative value means "read-only"
String255              // 255 bytes for author's remarks/instruction or copyright, ASCIIZ

// resolution  ---> new
SignedInt   2001       // chapter-ID, 2 bytes, negative value means "read-only"
ShortInt    ii         // element width (in CC2 .. CC:LSA: 10 pixel), 2 bytes
ShortInt    ii         // element height (in CC2 .. CC:LSA: 10 pixel), 2 bytes
                       // for v2.0: these values will override the individual settings
                       // of the patches

// color-depth for all cutouts  ---> new
SignedInt   2002       // chapter-ID, 2 bytes, negative value means "read-only"
ShortInt    ii         // color-depth (in CC2 .. CC:LSA: 16), 2 bytes
                       // for v2.0: this value will override the individual setting
                       // of the patches

// library menu color, 16-bit color info TARGA-like
SignedInt   6000       // chapter-ID, 2 bytes, negative value means "read-only"
LongInt     iiii       // number of color "pixels" to come, each "pixel" 2 bytes long
ShortInt    ii         // 2 bytes for 16-bit color information for menu display
```

```
// cover picture, always 200x200 pixels
SignedInt   7000        // chapter-ID, 2 bytes, negative value means "read-only"
LongInt     iiii        // cover pict width, counted in pixels
LongInt     iiii        // cover pict height, counted in pixels
Data                    // up to 80000 bytes cover image, 16-bit graphics, 2 bytes per pixel,
                        // picture should not be greater than 240x240 pixels.

// start of directory
SignedInt   8000        // directory-ID, 2 bytes, negative value means "read-only"
For every directory entry
            String255           // 255 bytes for an ASCIIZ string, name of cutout
            LongInt     iiii    // terrain cutout width, counted in terrain elements
            LongInt     iiii    // terrain cutout height, counted in terrain elements
            LongInt     iiii    // terrain cutout hot-point X, counted in terrain elements
            LongInt     iiii    // terrain cutout hot-point Y, counted in terrain elements
            LongInt     iiii    // terrain element width in pixels ---> new
            LongInt     iiii    // terrain element height in pixels ---> new
            LongInt     iiii    // terrain data offset, counted from top-of-file
            LongInt     iiii    // number of images, can be 0 or 1 for 5CC v1.08
            LongInt     iiii    // image width, counted in pixels
            LongInt     iiii    // image height, counted in pixels
            LongInt     iiii    // image opacity, value ranging between 0 and 100
            LongInt     iiii    // image color-depth, counted in bits ---> new
            LongInt     iiii    // image data offset, counted from top-of-file
Next directory entry

// start of user data
SignedInt   9000        // user-data area-ID, 2 bytes, negative value means "read-only"
data-terrain 0              // terrain data, each data element a signed int = 2 bytes per element
data-image 0                // image data, 2 bytes per pixel or 3 bytes per pixel, depending on
data-terrain 1              // the entry in the new section "1002"; in case of 3 bytes per pixel:
data image 1                // color byte sequence: Blue + Green + Red (like a BMP-file).
data-terrain 2
data-image 2
etc. ...

// eof – no indicator, appending bytes will be ignored, may contain comments or copyright
```

16-bit graphics will always occupy 2 bytes per pixel. 24-bit graphics will always occupy 3 bytes per pixel. For the 24-bit graphics of v2.0: the color byte sequence for every pixel is Blue + Green + Red (like in a BMP-file). Line orientation for 16-bit and 24-bit graphics is top-down (what differs from BMP-file format).

Mafi
cs2xh@web.de

http://cc2revival.npage.de/
http://closecombat2.npage.de/

formerly:

http://closecombat2.fortunecity.com/
http://www.closecombat2.claranet.de/
http://www.ftf.claranet.de/
http://www.geocities.com/cc2revival/