Entis CSX: Cotopha Image file

By Proger\_XP, Version 1  
*(Wanko to Lily, system \ wanko.csx, 3,813,713 bytes)*

**Code:**

The general format of CSX-file:   
.00 .. 40 - CSX file header   
.41 + - Sections   
  
CSX file header:   
.00: [S] "Entis" # $ 1A # 00 # 00   
...   
.0 C: [DW] Should be zero   
.10: [S] "Cotopha Image file"   
...   
.38: [DW] Contents size   
...

where:

* Should be zero - the field should be zero, otherwise the engine will report an incorrect file header. Maybe this version, zarezirvirovanoe box, the code expansion of the format or something else.
* Contents size - the size of the entire file, except for its title (in other words, the value of + 40 must be exactly 1 byte larger than the entire file).

CSX-file consists of sections. There are five types:

* image - the actual command script and text. The largest section.
* function - a function table, and apparently the entry points or something else. Just something where you have to change shifts.
* global - did not go into details, does not affect the possibility of translation.
* data - the same thing here.
* linkinf - do not specifically know what it is, because in the game, this section was empty ...

The Section has the following format:

**Code:**

.00: [S8] ID   
.08: [DW] Size   
.0 C: [DW] Unknown (usually 0)

where:

* ID - the form of section (one out of five). Extra space is padded with spaces (20, not zero).
* Size - size of the section without a heading.

*Some sections still Read more [DW] or more.*   
  
  
Unicode string   
The file uses a common format for all rows, which, incidentally, all Unicode (except ID sections, which are ASCII). Here I refer to it [US]:

**Code:**

.00: [DW] Character count   
.04: [S] Unicode string

where:

* Character count - the number of **characters** (not bytes). To get the length of the string must multiply it by 2.
* Unicode string - for example, 0030 BF7D ...

Linkinf section   
Since it is not interested in this section, I can only say that she has a title (in addition to a sectional):

**Code:**

.00: [DW] Count 1   
.04: [DW] Unknonwn   
.08: [DW] Count 2   
.0 C: [DW] Unknown

Linkinf has two parts, each of which has a number of "something."   
  
  
Data section   
Is also not the most interesting section.

**Code:**

Has a header of 4 bytes   
.00: [DW] Records count   
  
Record:   
.00: [US] Name 1   
+00: [DW] Unknown (usually 0)   
.04: [DW] Unknown (usually 80)   
.0 C: [US] Name 2

Examples of (Name 1 => Name 2):

**Code:**

input => InputFilter   
screen => Window   
frameskin => ResourceManager

Global section   
This section is, again, not of particular interest, since it does not affect the translation. However, this is what we know about her:

**Code:**

Has a header of 4 bytes   
.00: [DW] Records count   
  
Record:   
.00: [US] Name   
+00: [DW] Unknown   
.04: [DW] Unknown 2

It is very possible that Unknown 2 is the length of another Unicode-string.   
  
  
Function section   
But this is the most interesting place. Here, pointers are stored inside the section image, so here we need a fixed shifts when you change the script.   
  
Section consists of two parts. The first store just a shift in the second - the table with the names and offsets.   
*All addresses in this section are relative to the section image.* That is to get the absolute address should be added to address them early image (typically 50).   
  
**The first part**

**Code:**

Has a header of 4 bytes   
.00: [DW] Records count   
  
Record:   
.00: [DW] Relative offset

where:

* Relative offset - otnositelnyy address in the section image.

**The second part**

**Code:**

Has a header of 8 bytes   
.00: [DW] Unknown (usually 0)   
.04: [DW] Records count   
  
Record:   
.00: Relative offset   
.04: [US] Name

where:

* Relative offset - otnositelnyy address in the section image.

Image section   
Messages are two functions: Talk and Mess. Function HitretNewPage waiting user response and displays the following message (on a new screen).   
  
The peculiarity of Talk and Mess is that Talk (apparently) is always used for displaying messages, not speech, *but* not always Ondo message appears only a function of Talk.Usually the message is divided so that Talk displays the first 26 characters (the maximum that can fit on the screen in one line), and the rest of the output of the Mess (apparently also one function on one line).   
Interestingly, that any long string of fine displays even a single function. Not entirely clear why the script they are broken in this way.   
  
At the same time, the Mess is used for speech output (without the initial call Talk). Split one message is possible by means of catching HitretNewPage.   
  
  
**The procedure for the parser for the script** (RTF of this screen enclosure): 

**Attachment:**

http://honyaku-subs.ru/forums/styles/subsilver2/imageset/icon_topic_attach.gif [Cotopha parser jumps.rar](http://translate.googleusercontent.com/translate_c?hl=zh-CN&ie=UTF-8&sl=auto&tl=en&u=http://honyaku-subs.ru/forums/download/file.php%3Fid%3D107%26sid%3D4686e5c11c1628da8a33a1541e081e9a&prev=_t&rurl=translate.google.com&twu=1&usg=ALkJrhiP6DIOhNv2-bOekMpoFmoOa2dQLA) [03.03 KB]   
Downloads: 147

Now a few words about each function.   
  
**Talk**

**Code:**

.00: [US] "Talk"   
+00: [DW] Unknown (usually February 1, 2000 06)   
.04: [US] Message

**Mess**

**Code:**

.00: [US] "Mess"   
+00: [DW] Unknown (usually February 1, 2000 06)   
.04: [US] Message

**HitretNewPage**

**Code:**

.00: [US] "HitretNewPage"

This is enough to change rows.   
  
  
Where and what to change?   
When you change the line in the section image to change the following places in the CSX-file:

* @ 38 - the size of CSX-Failla without a title.
* "Image" section:
* - @ 08 - the size of the section;
* - The length of the Unicode-string in the script.
* - To correct short-distance jumps, see below.
* "Function" section:
* - Change the address in the first part of the section;
* - Do the same in the second part.

Short hops   
  
The script can also meet the function jumps. Usually there is relative jump distance (ie how many bytes to skip). Because of this it is necessary not only to change the address in the offset table (functions, see above), but also keep track of these jumps in the script and change the distance in them if necessary.   
  
For example, a function *IsGameClear,* in which the distance is recorded here: [us] *IsGameClear* [w] [dw] *distance* (ie, 2 unknown bytes after the function name, and then shift the size DWord). The same syntax to have similar features: *ChkFlagOn, ChkSelect* and *OnFlag.*   
  
Function names, as usual, of Unicode (for the English names of the symbol is simply the code + 0x00 for 2-th byte).   
  
**One more thing.** Options jumps can sometimes refer to the code that is in this context is also a leap. This code - one byte = *0x06* followed the 4-byte-range jump dyal: [b.]*0x06* [dw] *distance.* So that such "jumps" are also needed to monitor and tinker with.   
  
Generally, scripts are replete with jumps - for example, *Yosuga no Sora* only 4 choices, but the script is already 22 jump.   
  
  
In conclusion ...   
The good news is that the engine always uses Unicode, so the problems with any language (including our favorite Cyrillic) no. Very curious fact that if you disable hard returns in the Japanese style through a small hack, the engine itself will carry Russian and English words and even [add](http://www.solelo.com/p4s/img/wanko1.png) hyphens in them.   
By the way, the engine is running and without the established Japanese locale.   
  
That's all.