GenCodeC

version 2.00
English Documentation

Eric Totel

1 Introduction

MUIBuilder does not produce directly usable code, but a description of the final code which can be adapt to the destination language (not so easy, but possible !!) . . . at least those supported by MUI !!

To make this code compilable, it must be parsed with an external code generator. GenCodeC is the C language external generator.

GenCodeC is **FREEWARE**. BUT it is copyrighted by me (Eric Totel). You can distribute it, modify its sources, modify it as long as my name remain in the code and as long as this documentation is distributed with the new GenCodeC you created. Moreover, if you created such a new generator, you MUST send it to me (Eric Totel) BEFORE you release it !!!

See more about the author in the MUIBuilder documentation.

2 How to write GenCodeC ?

In this archive you could find the sources of GenCodeC : feel free to modify it, to write enhance it . . . or what you want !

The goal of this program is to produce some C code with the temporary files produced by **MUIBuilder** in T:. It uses the functions available in the muibuilder.library : each function is described in the autodocs.

3 Use

You must know tow different things : how to use GenCodeC itself and how to use the source generated by GenCodeC. I will try to explain both here :

3.0.1 Use of GenCodeC

`GenCodeC` is a CLI only program. It is launched by **MUIBuilder** (using a `SystemTags` call) when you request some C source code from **MUIBuilder** [just select the corresponding language in the preferences pannel].

If you run it from a shell when **MUIBuilder** is not active, it will probably create no file, because there is probably no temporary files generated by MUIBuilder in T:. GenCodeC needs no parameters : all of them are included in the temporary files.

3.0.2 Generated Code

`GenCodeC` generates (for a given `foo` object of an `App` application) :

- a header file in which you'll find the foo object definition
- a .c file (whose name is defined in **MUIBuilder**)[1] in which you can find a `Createfoo` function which returns the object foo.
- a file `fooExtern.h` : it contains all external references to some variable or functions of your own.

Two configurations files called `H-Header` and `C-Header` allow to customize the generated code : GenCodeC copies them at the head of the generated code.

The creation procedure creates both the objects and the notifications. It adds the windows opening too. The example which you can find in this documentation is really usable in most of the cases.

3.0.3 Source Example

```
#include <libraries/mui.h>

/* protos */
#include <clib/muimaster_protos.h>
#include <clib/alib_protos.h>
#include <clib/dos_protos.h>
#include <clib/exec_protos.h>

/*  Pragmas  */
#include <pragmas/muimaster_pragmas.h>
#include <pragmas/exec_pragmas.h>
```

[1] if you don't give the '.c' extension it will be automatically added by `GenCodeC`

```
/*  Ansi  */
#include <stdlib.h>
#include <stdio.h>

/* MUIBuilder */
#include "NONE.h"

struct Library * MUIMasterBase;

/* Init function */
static void init( void )
{
if (!(MUIMasterBase = OpenLibrary(MUIMASTER_NAME,MUIMASTER_VMIN)))
{
printf("Can't Open MUIMaster Library");
exit(20);
}
}

/* main function */
main()
{
struct ObjApp * App = NULL; /* Application object */
BOOL running = TRUE;
ULONG signal;

/* Program initialisation ( you need to write it yourself) */
init();

/* Create Application : generated by MUIBuilder */
App = CreateApp();

while (running)
        {
                switch (DoMethod(App->App,MUIM_Application_Input,&signal))
                {
                case MUIV_Application_ReturnID_Quit:
                        running = FALSE;
                        break;
                }
if (running && signal) Wait(signal);
        }
DisposeApp(App);
CloseLibrary(MUIMasterBase);
exit(0);
}
```

3.0.4 How to insert objects.

I'm sure some people will miss some objects in MUIBuilder !! So i must say it's not a problem :-)))

MUI provides the ability to dynamically add objects in a GUI : so you have to create your own function to build your object, and use it as follow :

```
extern APTR CreateMyObject();

/* MUIBuilder source code */
App = CreateApp();

/* Dynamically add your own object */
DoMethod(App->EmptyGroup, OM_ADDMEMBER, CreateMyObject());
```

The group EmptyGroup must be built with MUIBuilder and IS one of the objects of the GUI.

4 Tips and Hints !

You must know some features of the generated code before using it :

1. The structure defined in the header file contains only the labels of the objects that you specified as generated in **MUIBuilder**. (you can see the 'G' letter at the right of their name in the main creation window).
2. The file fooExtern.h contains external references to variables or functions of your own : these definitions are standard, so you must modify them to your needs. Once they are in this file : GenCodeC won't change them, so you will modify their definition ONLY once !!! GenCodeC will NOT erase these modifications the next time you wil generate the same object !!!
3. The header file used to provide locale support (this file should be generated either by CatComp or Flexcat) MUST be called foo_cat.h.
4. Modify immediatly `C-Header` and `H-Header` to fit your needs : they are really important !!! They are the only way you have to customize the source code generated by GenCodeC !

Short Contents

Table of Contents