# E code generation

| COLLABORATORS | | | |
|---|---|---|---|
| | *TITLE* :<br><br>E code generation | | |
| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
| WRITTEN BY | | July 26, 2024 | |

| REVISION HISTORY | | | |
|---|---|---|---|
| NUMBER | DATE | DESCRIPTION | NAME |
| | | | |

# Contents

# Chapter 1

# E code generation

## 1.1 GenCodeE v1.4

```
*****************************************************************************

                              GenCodeE (v1.4)
                    E code generation module for MUIBuilder
                            Update of 18 april 1994
                    © Copyright 1993, 1994, Lionel Vintenat

*****************************************************************************

 Introduction
 The generated files
 The generated code
 The examples
 Bugs
 History
 Future
 Distribution
 The author
```

## 1.2 Introduction

Warning, this new version of GenCodeE (1.4) doesn't replace the previous one (1.3) : GenCodeE v1.4 does the same thing than GenCodeE v1.3, but it does it differently !

Indeed, GenCodeE v1.3 and GenCodeE v1.4 are both designed to work with MUIBuilder v1.1 and generate EXACTLY the same code. Howerver, GenCodeE v1.3 directly modify your source files in environment mode, whereas GenCodeE v1.4 always generate 2 separated files which must be joined with EPP, the E preprocessor of Barry Wills.

If you aren't an EPP user, GenCodeE v1.4 is completly useless for you, so keep GenCodeE v1.3 in this case, but otherwise...

Note that EPP is available in its last version (v1.4d) on aminet in dev/e. Try it, it's really a good tool for Amiga E v2.1b.

## 1.3 The generated files

GenCodeE v1.4 always generates 2 files. If inside the 'Code' string gadget of the code generation window of MUIBuilder, you have written (for instance) "Prog" or "Prog.e", GenCodeE v1.4 will generate the files "Prog_defs.e" and "Prog_procs.e".

If you don't activate the 'Environment' checkmark in this window, you will obtain :
- in "Prog_defs.e" some strings like "var_name : LONG"
- in "Prog_procs.e" the "raw" code of the application or of an object according to the case

This is only interesting for the unlikely case where you will have to do some cut/paste directly in your code.

But all the interest of GenCodeE v1.4 appears in environment mode. Indeed, in this mode, you can directly use the 2 generated files with EPP in adding somewhere in your main source file the line "PMODULE 'Prog_defs', 'Prog_procs'". EPP will intelligently join all the pieces.

In this mode, "Prog_defs.e" contains :
- some calls to external E modules with the "MODULE" instruction : these modules are necessary to use MUI
- the MUI_TRUE constant definition (see the generated code )
- the definition of an E object whose name is "obj_app" if you have requested the code generation of the whole application, or "obj_ObjectMUI" if you have requested the code generation of the "ObjectMUI" object, inside MUIbuilder; the fields of this E object are all the generated variables

In this mode, "Prog_procs.e" contains :
- a procedure whose name is "create_app" if you have requested the code generation of the whole application, or "create_ObjectMUI" if you have requested the code generation of the "ObjectMUI" object, inside MUIbuilder; this procedure contains all the creation code of the application or of the object according to the case
- a procedure whose name is "dispose_app" if you have requested the code generation of the whole application, or "dispose_ObjectMUI" if you have requested the code generation of the "ObjectMUI" object, inside MUIbuilder; this procedure contains all the deleting code of the application or of the object according to the case
- the "doMethod" procedure for E language (its source was given to me by Wouter van Oortmerssen, the Amiga E author) : this one is essential to use MUI

After, the using of these 2 files is quite simple. As the variables aren't generated alone, but inside an E object, you can declare as much variables of type this "obj_ObjectMUI" object as you need copies of this object in your application. Then, you assign these variables by a call to "create_ObjectMUI" when you need it. Finally, to delete all these object copies, you use each time the "dispose_ObjectMUI" procedure. It's all ! You can like that easily create number of times the same object in your GUI.

For the application itself, although it is possible, you will of course create only one copy of it.

Even if all that sounds a bit confusing for you now, make some tries, all that is very intuitive (and above all very powerful).

## 1.4 The generated code

As MUI-Builder generates generic code which is later interpreted by modules, E generated code is identical in its content to C generated code. So, refer to language C paragraph for all that is related to generated variables, notification and localization. The only important difference is the MUI_TRUE constant which must be defined to the value 1 and which is used instead of TRUE (which equals -1) for MUI, because MUI doesn't always recognize value 1 like true.

The only difficult point is related to the strings you enter using MUI-Builder. The general rule is to enter them as if you want to use them with C language, with one exception: unlike what was stated before, the character '"' can be entered without a '\' (it was a problem only C).
The E code generation module will translate it correctly. More precisely, this module recognizes:

- \r replaced by \b
- \n, \t et \e leaved unchanged
- \0oo where oo is an octal number : be careful, the \ must be followed by a 0, itself followed by 2 figures among 0 and 8
- \xhh where hh is an hexadecimal number : be careful, the \ must be followed by a x (case sensitive), itself followed by 2 correct symbols (figure or letter among a and f, or A and F)
- ' replaced by \a

Be careful - if the syntax of \0oo or the syntax of \xhh isn't correct, the E generation code module won't print any error but the result may not be the one you expect!

If E code generation module meets \033 or \x1B (decimal value 27, namely the ESC code), it will replace it by \e. If on the other hand it meets \0oo or \xhh with a decimal value different from 27, it will produce a string not under the form 'string', but under the form [ "s", "t", "r", "i", "n", "g", 0 ]:CHAR because there isn't equivalent things in E language to \0oo and to \xhh of the C language. But the 2 forms are totally equivalent.

Finally, the generated source file is full of macros identical to the C language ones. To use those macros, you will need an annex preprocessor because Amiga E doesn't support macro replacment for the moment. So, MUI-Builder is provided with an extra archive, Mac2E.lha (available alone on aminet too). You will find in this archive all that you need to use generated source files, especially with a preprocessor specialized in macro replacment designed for Amiga E. Refer to this archive for more details.

## 1.5 The examples

Nothing special to say except that I used FlexCat for the localized examples. For each one, I provided, in addition to the source files and to the executable file , the files xxxx.cd, xxxx.ct (French version), xxxx.catalog (French version) and xxxx_cat.e (the code generated by FlexCat for this catalog).

## 1.6 Bugs

None known.

## 1.7 History

```
Version 1.0 :  - 1st distributed version with MUIBuilder v1.0
Version 1.1 :  - a little bug fixed (GenCodeE sometimes forgot to indent the
               begining of the code)
               - version never distributed
Version 1.2 :  - localization support added
               - version never distributed
Version 1.3 :  - environment mode added
               - 2nd version distributed with MUIBuilder v1.1
Version 1.4 :  - 3rd distributed version, external to the MUIBuilder archive
               - must be used with MUIBuilder v1.1
```

## 1.8 Future

As I think that separated compilation is an essential aspect to every "serious" compiler, all the following versions of GenCodeE will use this feature, through EPP while waiting for Amiga E v3.0, then after through Amiga E v3.0 when this one will be released.

## 1.9 Distribution



## 1.10 The author

You can reach me by mail at:

- my student address, up to and including July 1994

  Lionel Vintenat
  appartement 21

11 rue François Oulié
31500 TOULOUSE
FRANCE

- my family address:

Lionel Vintenat
3 impasse Boileau
Lotissement Les Termes
87270 COUZEIX
FRANCE

Write to me at my student address until July 1994 as I'm there much more often than at my family address.

You can also reach me on the INTERNET. My e-mail address is vintenat@irit.fr. I would prefer that you contact me by e-mail rather than mail. I will reply to all questions that are sent to me by e-mail, but don't expect a reply by mail (I am very lazy when it comes to picking up a pen...).