

MUI-AmigaE

COLLABORATORS

	<i>TITLE :</i> MUI-AmigaE	
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>
WRITTEN BY		February 12, 2023
		<i>SIGNATURE</i>

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	MUI-AmigaE	1
1.1	MUI-AmigaE.guide	1
1.2	Introduction	1
1.3	Changes	2
1.4	The Files	3
1.5	Files: Modules/muimaster.m	3
1.6	Files: Modules/libraries/mui.m	3
1.7	Files: Modules/libraries/mui.e	4
1.8	Files: Modules/libraries/muip.m	4
1.9	Files: Modules/libraries/muip.e	5
1.10	Files: Modules/mui/muicustomclass.m	5
1.11	Files: Modules/mui/muicustomclass.e	5
1.12	Files: Examples/#?.e	6
1.13	Files: Tools/brush2e	6
1.14	Files: Tools/brush2e.e	7
1.15	Problems with MUI and E	7
1.16	Problems: TRUE=1 or TRUE=-1	8
1.17	Problems: Problem with SetAttrsA()	8
1.18	Problems: TAG_IGNORE,0	9
1.19	Author	9
1.20	Future	9
1.21	Disclaimer	10

Chapter 1

MUI-AmigaE

1.1 MUI-AmigaE.guide

MUI 3.7 DEVELOPER FILES FOR AMIGA E

~~~~~

Introduction  
First words.

Changes  
What's new ?

Files  
Description of the files.

Problems  
Problems with MUI and AmigaE.

Future  
New MUI -> new files ?

Author  
My address.

Disclaimer  
No warranty!

### 1.2 Introduction

Introduction

~~~~~

This archiv contains all files which are needed to write MUI-programs

in E, and some additional information.

If you use this files, please send
me
a postcard or an email. I'd
like to know, how many people are using this files!

And if you have any questions about MUI-programming with E, just send
me an email and I'll try to help you.

1.3 Changes

CHANGES

~~~~~

Changes since "mui36dev-E.lha":

- MUI 3.6 to MUI 3.7 update.

Changes since "mui33Edev.lha":

- MUI 3.3 to MUI 3.6 update.
- Some more
  - example-sources  
(thanks to Klaus Becker)
- The set() and nnset() macros in  
mui.m  
are now PROCs. Read  
here  
to  
see why.
- Moved  
muicustomclass.m  
from "Modules/tools/" to "Modules/mui/".

Changes since "mui32Edev.lha":

- Little MUI 3.2 to MUI 3.3 update.
- Some more
  - example-sources  
(thanks to Sven Steiniger)

Changes since "mui31Edev.lha":

- MUI 3.1 to MUI 3.2 update.
- New  
brush2e  
tool.

Changes since the files from the mui31dev.lha-archive:

- Complete and already compiled modules.
- New module  
muicustomclass.m  
.
- New example-source "Class1.e".

Changes since "mui23Edev.lha":

- E versions <3.1a are no longer supported to keep the total size of this

Amiga-E files reasonable. If you still use E v2.1b than update now or use "mui23Edev.lha".

## 1.4 The Files

```

                                THE FILES
                                ~~~~~~

Modules/

 muimaster.m
 Modules/libraries/
 mui.m

 mui.e

 muip.m

 muip.e
 Modules/mui/
 muicustomclass.m

 muicustomclass.e
 Examples/
 example-sources
 Tools/
 brush2e

 brush2e.e

```

## 1.5 Files: Modules/muimaster.m

```

"Modules/muimaster.m"
~~~~~

```

This file defines the functions of the muimaster.library. Copy it to EMODULES: and to use it write in your source:

```
MODULE 'muimaster'
```

## 1.6 Files: Modules/libraries/mui.m

```

"Modules/libraries/mui.m"
~~~~~

```

This module contains a complete translation of the original C mui.h file with only one exception: The muip\_... OBJECTs are moved to the file

```
Modules/libraries/muip.m
```

```
.
```

The #defines (and CONSTs) are written exactly the same way as in the mui.h file (with one exception: The macro String() is renamed to StringMUI() to avoid conflicts with the E function String()). The identifiers of the OBJECTs are all written totally lowercase!!

Copy this file to "EMODULES:libraries/mui.m". To use it in your source write:

```
MODULE 'libraries/mui'
```

Don't forget to add "OPT PREPROCESS" to your source, if you want to use the macros!

See also:

```
Modules/libraries/mui.e
and
Modules/libraries/muip.m
```

## 1.7 Files: Modules/libraries/mui.e

```
"Modules/libraries/mui.e"
```

```
~~~~~
```

This is the source-file of  
mui.m

. Use it to find out what the macros  
do and maybe read the comments.

## 1.8 Files: Modules/libraries/muip.m

```
"Modules/libraries/muip.m"
```

```
~~~~~
```

This module contains all the MUIP\_...-structs of the original C mui.h file as OBJECT's. I think, this OBJECT's are not very usefull in E, that's why I didn't include them in the first releases. But to have a complete AmigaE-MUI-interface...(and to use customclasses...)

Copy this module to "EMODULES:libraries/muip.m". To use this module, write in your source:

```
MODULE 'libraries/muip.m'
```

See also: Modules/libraries/muip.e and "Examples/Class1.e"

## 1.9 Files: Modules/libraries/muip.e

```
"Modules/libraries/muip.e"
~~~~~
```

```
This is the source-file of
    muip.m
.
```

## 1.10 Files: Modules/mui/muicustomclass.m

```
"Modules/mui/muicustomclass.m"
~~~~~
```

If you want to use custom-classes in you programmes - and the new philosophy of MUI-programming is, to use as many custom-classes as possible (have a look at the comments of psi.c !) - you'll get a problem.

The dispatcher-function of the custom-class is called as a hook-funktion. Normally you would use the installhook()-function from the installhook-module to install the hook-structure of a hook-function.

But with custom-classes, MUI will install the hook-structure for you and uses the userdata-field of this structure (which is normally used by installhook to store the contents of A4) itself. So you can't use installhook() with a custom-class-dispatcher.

To solve this problem, I wrote the "muicustomclass.m"-module. It contains a function called eMui\_CreateCustomClass(), which should be used as a replacement for the original Mui\_CreateCustomClass()-function. Have a look at the source of this module and at some of the example-

```
sources
to see, how to use it and how it works.
```

Note: The userdata-filed of the iclass-structure your dispatcher gets, is used by this module to point to some important data. But the first LONG of this data is unused and could be used as a new userdata-field. Simply write 'iclass.userdata[]' instead of 'iclass.userdata'.

```
See also:
 muicustomclass.e
.
```

## 1.11 Files: Modules/mui/muicustomclass.e

```
"Modules/mui/muicustomclass.e"
~~~~~
```



This is the source of  
     muicustomclass.m  
     . Have a look at the comments  
 to see, whats going on exactly.

See also the  
     example-sources

## 1.12 Files: Examples/#?.e

"Example-sources"  
 ~~~~~

The "Examples"-directory contains some example-sources. They are all translations of original C-sources which came with the original MUI-developper-archiv.

file: written/translated by:
 ~~~~~

|                |                    |
|----------------|--------------------|
| Appwindow.e    | Klaus Becker       |
| Balancing.e    | Sven Steiniger     |
| Boopdoor.e     | Klaus Becker       |
| Class1.e       | Jan Hendrik Schulz |
| Class2.e       | Sven Steiniger     |
| Class3.e       | Sven Steiniger     |
| Dragndrop.e    | Klaus Becker       |
| Envbrowser.e   | Klaus Becker       |
| Inputhandler.e | Klaus Becker       |
| Layout.e       | Klaus Becker       |
| MUI-Demo.e     | Jan Hendrik Schulz |
| Pages.e        | Klaus Becker       |
| Settings.e     | Klaus Becker       |
| ShowHide.e     | Klaus Becker       |
| Showimg.e      | Klaus Becker       |
| Slidorama.e    | Sven Steiniger     |
| Subtask.e      | Klaus Becker       |
| Virtual.e      | Klaus Becker       |

## 1.13 Files: Tools/brush2e

"Tools/brush2e"  
 ~~~~~

This little tool is based on "brush2c" which came with MUI. Use brush2e if you need a Bodychunk-object. With brush2e you could build an emodule from an IFF-brush.

Usage: brush2e ILBMFILE/A,EMODSRC/A,NOHEADER/S

ILBMFILE = The IFF-brush
 EMODSRC = The name of the emodule-source that will be created.
 NOHEADER = Don't create the imgXxxHeader() PROC (see later).

The created emodule contains the following CONSTs and PROCs: (The XXX or Xxx is the name of the brush-file without path and extensions)

CONST: IMG_XXX_WIDTH
 IMG_XXX_HEIGHT
 IMG_XXX_DEPTH
 IMG_XXX_COMPRESSION
 IMG_XXX_MASKING

PROC: imgXxxBody() - Returns a pointer to the BODY-datas
 imgXxxColors() - Returns a pointer to the colors (CMAP-chunk)
 imgXxxHeader() - Returns a bitmapheader-OBJECT. As this PROC
 is normally not needed, you could tell brush2e
 to not create it (with NOHEADER).
 imgXxxObject() - This PROC creates a new MUI-Bodychunk-object
 by using the above PROCs and CONSTs.

See also:

brush2e.e

1.14 Files: Tools/brush2e.e

"Tools/brush2e.e"

~~~~~

This is the source of  
 brush2e  
 . It is based on the brush2c-source  
 that came with MUI, but brush2e is much more powerfull.

See also:

brush2e

## 1.15 Problems with MUI and E

PROBLEMS WITH MUI AND E

~~~~~

MUI_TRUE
 Why a CONST MUI_TRUE?

SetAttrsA()
 Problems with set()

TAG_IGNORE

Why "TAG_IGNORE,0," in some macros?

See also

muicustomclass.m
to avoid problems with custom-classes!

1.16 Problems: TRUE=1 or TRUE=-1

TRUE=1 or TRUE=-1

~~~~~

In C TRUE has the value 1 but in E TRUE has the value -1. That can cause problems, therefore a CONST "MUI\_TRUE" with value 1 is defined in

Modules/libraries/mui.m  
. Use this one instead of TRUE whenever you want  
to give TRUE to MUI. See  
example-sources  
.

## 1.17 Problems: Problem with SetAttrsA()

Problem with SetAttrsA()

~~~~~

Whenever you try to set an attribut to the value it allready has, MUI overwrites this attribut in the taglist with TAG_IGNORE to make shure that notify-class don't react on it. This is nessesary to prevent endless-notification-loops.

As long as you are a C programmer and use SetAttr() that is no problem, because MUI then changes only a *copy* of the original datas of the SetAttrs()-call. But if you use SetAttrsA(), as we must do because SetAttrs() is a function of amiga.lib, it is a problem!

You only give a PTR to the original-datas with SetAttrsA() and now MUI changes this *original* datas. If you then uses this datas again at the next execution of the same SetAttrsA()-call, the datas are still changed and nothing (TAG_IGNORE) will happen :-)

One possible way to avoid this problem is, to make the attribut a non-constant data. Instead of

```
SetAttrsA(obj, [ MUIA_..., value, ..., TAG_DONE])
```

write

```
SetAttrsA(obj, [Eval(`(MUIA_...)), value, ..., TAG_DONE])
```

~~~~~

Now everytime this SetAttrsA()-call is executed, the attribute is again

"evaluated" and stored in the list.

To avoid the SetAttrs()-problem the macros set() (and nnset()) in the file

```
Modules/libraries/mui.m
where defined like this:
```

```
#define set(obj,attr,value) SetAttrsA(obj,[Eval(`(attr)),value,TAG_DONE])
```

But, unfortunately, MUI (since MUI 3.0 ?) changes the "value" in some cases too. (Thanks to Sven Steiniger for telling me this!) One possible solution to avoid this problem, is to use the Eval()-trick on the value too, but a better way (faster and \*much\* smaller) is to use a PROC for set() and nnset(). So I changed these macros in

```
Modules/libraries/mui.m
into PROCs.
```

## 1.18 Problems: TAG\_IGNORE,0

```
TAG_IGNORE,0
```

```
~~~~~
```

Some of the macros in

```
Modules/libraries/mui.e
```

are ending with

"[TAG\_IGNORE,0,". That seems to be superfluous but it is needed, because in AmigaE v2.1b if you want to split a statement over several lines, you can't end a line with "[" but with a comma. And with AmigaE v3.1 it's needed to allow the same use of the macros.

## 1.19 Author

```
AUTHOR
```

```
~~~~~
```

Snail mail: Jan Hendrik Schulz

Elsässer Str. 19

22049 Hamburg

Germany

e-mail: schulz\_j@informatik.fh-hamburg.de

or

schulzjan@dame.shnet.org

## 1.20 Future

---

## FUTURE

~~~~~

If there comes a new MUI release, I will create a new archive with AmigaE-files like this. You'll find it on Aminet in dev/mui.

If you have some ideas to make things better, please write
me
.

1.21 Disclaimer

DISCLAIMER

~~~~~

This files are provided "AS IS" without warrenty of any kind, expressed or implied! I'm NOT liable to you for damages or problems, including any general, special, incidental or consequential damages or problems arising out of the use or inhability to use of the files. Including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of a program build with this files.

(I hope you know what I want to say! My english is not good enough to write such legal stuff!)

---