MUIBuilder

version 2.2
Documentation

Eric Totel

Copyright © 1993-1994 Eric Totel

This is the second edition of the MUIBuilder documentation, and is consistent with version 2.2 of MUIBuilder. Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies. Permission is granted to copy and distribute translations of this manual into another language, under the above conditions, except that this translation may be approved by the author.

1 Introduction

Thanks for trying MUI-Builder !!!

MUI-Builder is a tool that I hope you find to be user-friendly. It is, however, far from perfection.

Please feel free to send me all your ideas and opinions about this software so it may evolve in the way you'd like.

With MUI-Builder, you'll be able to write `MUI` applications without writing lines of source code nor knowing anything about `MUI` functions' syntax (even though MUI is simple in itself).

MUI-Builder's aim is to let you create a graphic interface without technical problems, and with no more effort than thinking about your final goal.

At the beginnning, I wrote this program only for my own needs and for learning how to use MUI. MUI is a wonderful tool written by Stefan Stuntz

I hope you find this program as useful as I've already found it.

1.1 Advantages of MUIBuilder

Many people will wonder about the usefulness of this program, as MUI is simple to program in the first place.

The advantages that I and all the beta-testers have found in using MUI-Builder follow:

1. MUI-Builder is a simple way to quickly learn the principles of MUI programing by reading the generated source code.
2. MUI-Builder is a sort of MUI interpreter. You are able to test the look of your application's GUI and see how the result will appear in real time.
3. MUI-Builder offers the user a great flexibility in source code generation. You will be able to select precisely which object code you want to generate, with or without the declarations, initializations, localization ...
4. You can finally build the online help of your application directly from the application builder.
5. It is the very first builder that provides for easy localization !

6. In this new version appear fantastic :) new functions, that you have never seen before in similar tools on Amiga. It is now possible to build the events notifications directly from the interface builder AND TO TEST THEM.

2 Principles

MUI-Builder works on the same principles as MUI, and the two programs are intricately interwoven. My first advice is to read MUI documentation to better understand how it works.

MUI-Builder allows you to create the complete interface of an application. For example, all the windows of this application were created with MUI-Builder.

Each application is made up of a hierarchy of interface objects. As an example, the following diagram depicts a possible application.

```
            Application
                |
                |
               / \
              /   \
             /     \
            /       \
      window 1     window 2
       |               |
       |               |
      / \             / \
     /   \           /   \
    /     \         /     \
objet1   objet2  objet3  objet4
```

MUI-Builder is aimed at letting you create this tree simply by using a simple user-friendly graphic interface (at least, I hope so).

3 Distribution

This distribution consists in a lharc archive. This archive must not be modified in any case. This program must not and can't in any way be distributed on another form, without the written permission of the author.

Moreover, this program may be freely distributed, as long as no charges more than reasonable copying and handling fees are collected. (must not be more than 4$-US, 15FF, 4.DM). For any other type of distribution, you must have the permission of the author. This program may be included in freeware collections, providing that the previous conditions are respected.

MUIBuilder is **Giftware**.

This means you are absolutely NOT forced to send me money to go on using this program. But, if you really like this software or find it really useful, then you are may send about 15 $US or 50FF to the following address:

```
Eric Totel
26 route de Montsuzain
10150 Voue
France
E-Mail  : Eric.Totel@ramses.fdn.org
Fidonet : 2:320/104.56
```

If you didn't enjoy MUI-Builder enough to send me some money, please send me a postcard, a letter, an email to encourage me, give me some ideas or remarks ... and so on ... Some people sent me some programs they made : what an excellent idea !! It seems to me it is one of the best gifts you can send to me ! If you use MUIBuilder, i would be happpy if you write it in your documentation ...

Don't hesitate at all to write me : I absolutely want to know if further developments of MUI-Builder will be really useful and appreciated.

Please don't send files at this e-mail address. Just write a message, and i will tell you where you can send files to me !! Thanks in advance !

4 Disclaimer

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDER AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (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 THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

5 Tutorial

To help beginners to understand quickly how this program must be used, i've included a little tutorial example.

We will build step by step a little GUI, whose goal will be to display a simple text when we click on some buttons.

- First of all we have to create a window (`New Window` button in the main window).
- Select the only group in the list containing the GUI objects, and click on the `Add Object` button.
- Choose a Text object, define its parameters (you don't need to modify anything at this time) and click on `Ok`.
- Your Text object is now inserted in your GUI tree.
- Now add a group (choose an horizontal group by checking the `Horizontal` attribute to TRUE).
- If everything goes well, you'll see the group name just under the text object : select this group, and add three buttons in it. Just define their titles '1st button', '2nd button' and '3rd button' and their shorcuts '1', '2' and '3'.
- Your GUI is now completed ! Now we will define the notifications on these objects.
- Select the first button, and click on the `Notify` button.
- In the window which appears, choose the `Release button` event, then the text object you previously created, and at last the `Put a constant value` action. Add this notification simply by double-clicking on the action name, or click on the `Add notify`. MUIBuilder will ask for a string for you to enter. Enter '1st button selected'. Once added, quit the notification window.
- Repeat the previous operation on the two other buttons : enter '2nd button selected' and '3rd button selected' as constant strings.
- Once you've returned to the creation window, just press the `Test` button, and click on the buttons of your GUI.

This is of course a really litlle GUI, but i hope it will always be as easy to do bigger ones !! MUIBuilder should be able to help to build some tricky interfaces now ... or at least i hope so !!! :-)

6 Use of MUIBuilder

The principles on which MUI-Builder is used can be classified in the following headings.

6.1 MUIBuilder Preferences

This configuration Panel offers many choices for parameters for the program. All choices will be saved in an environment variable (`ENV:MUIBuider.env`) which is creating when you are installing MUIB.

There are the following choices :

1. Ghost windows : to see ONLY one window of MUIBuilder at a time. (very useful for slow computers, and for people who dislike having too many windows on the screen !)
2. Icons : select it if you want an icon created with the save file
3. Requests : select it if you want all the warning requesters
4. Test : if you select this option, you will see the result for your work during the building you your GUI.
5. Code : chose your generator module
6. Editor : You must specify here the name of your favorite editor. This editor should be run in a synchronous way ! (i.e. not as a background task). the following editors should work without any problems :
   - Vim (v2.0 and latter) : with the '-x' option
   - CygnusEd : with the '-keepio' option
   - GoldEd : with the 'STICKY' option
   - TurboText : with the 'WAIT' option
   - Ed : default editor
7. GetString : If you enter a string here, it will be the default name of the 'GetString' function which is used to localize your program. If this String is EMPTY, then MUIBuilder will create a name using the application name ('GetApplicationNameString').
8. Depth : allows you to define the minimal depth used when displaying the tree hierarchy in the creation window.
9. Character : this character is inserted in the tree display to make hierarchy more readable.
10. Label : if you set this option, the names of the objects will appear to make the hotkay visible.
11. Columns : number of columns used to display the image objects.

12. Images Location : The directory where you can find the objects images.

Once you have selected these options, you can save them by selecting the corresponding button. When you save the preferences, it saves your working directory too (in the environnment variable of course).

6.2 Application save file

The SAVE and LOAD buttons will respectively save to and load from a file the interface of the application you're creating. The Asl requester will open on your working directory (defined in the `MUIBuilder.env` variable). Using these buttons, you can continue with a previously unfinished application, or modify an already existing application.

Note that the MUIBuilder main window is an AppWindow. You can therefore load an MUIBuilder save file by dropping its respective icon on the window.

Moreover, most of you will see that the save files is an ascii file. So it is easy to edit it using a text editor. Please don't modify them in this way : there are a lot of chances that you corrupt the file . . .

6.3 Code generation

When you click on the 'Code' button of the MUI-Builder main window, you start the generation of the source code.

MUI-Builder will verify that you haven't used the same label for two different objects twice.

If everything goes well, you'll see two lists :

- one containing the names of all objects in you application - one containing the names of all labels that will be generated in your source code.

You will be able to control very precisely the way MUI-Builder will generate the labels in your code. If you don't care about this option, MUI-Builder will do everything automatically for you!

Before generating the source code, you'll have to define the options. (see Section 6.3.1 [Options], page 13).

The software actually creates generic code, which is more a description of the program than a real compilable source.

After this generic code is created (temporary file in 'T:'), MUIBuilder executes one of the code-generation modules (located in the 'modules' directory) which uses the temporary file. Actually most languages are available.

If you feel good enough (!!!) to program a module for you favorite language : feel free to contact me !!! I'll explain to you how to use the generic code to create the source of the alien language !!! The muibuilder.library provides some functions to help you to create the code in your language.

6.3.1 Code generation : Options

Five options are available :

- `Declarations`: to obtain the declarations and initializations in your source
- `Environnement`:select it if you want to generate code for includes, event loop, procedure declaration ...
- `Code`:select it if you want to generate the MUI code.
- `Locale`:to generate localize code (see Section 6.4 [Locale], page 14).
- `Notifications`:to generate notifications source code.

Note that each option may be selected independently from the others. You can also create exactly the part of the code you desire ...

Example:

You've created a window and you want to add a button in your code. Select only the 'Code' option and create it will create the button code! After inserting this text in the program, you'll need the declaration of the object button in your source. Select 'Declaration' ... and insert the generated text directly where you want in your source !

6.3.2 Application code

This button enables you to generate the source code for the whole application. The generated source will depend on the options you previously choose.

6.3.3 Object code

This button enables you to generate the source code of an object you previously chose in the 'Objects labels' list. This operation can really be of interest, if you want to be able to create stand-alone MUI objects : for example, you could be able to generate each window independantly, or include an object in more than one window by calling the function creating this object.

6.3.4 Remove a label

By selecting this button, you can deactivate the code generation for the object selected in the 'Generated Labels' list.

MUI-Builder automatically knows if it has to generate the label of each object. For example it is often useless to keep a pointer variable to a MUI-Group, unless you want to dynamically add objects to this group during the execution of the program.

This button, and the 'Add label' button (see Section 6.3.5 [Add a label], page 14) allow you to change MUI-Builder standard definitions on objects.

6.3.5 Ajouter un label

By clicking on this button, you tell MUI-Builder to generate the label of the selected object (see Section 6.3.4 [Remove a label], page 14).

6.4 Localisation

When you select the 'Locale' Option in the code-generation window, you chose to generate localized source code. In fact, all strings and shortcuts will be replaced by a call to a 'GetString' function (which should return a locale string from your catalog file).

A lot of program are able to generate this function for you (Catcomp and Flexcat for examples).

The name of the 'GetString' function can be chosen either in the preferences window, or in the code-generation window.

If you don't know how to use these features, please take a look at the examples included in the MUIBuilder archive.

6.4.1 Catalog

The name of the '.cd' file must be given before generating the Catalog.

Here is a summarize of what is necessary to know about the catalog description file ('xxxx.cd' file)

You'll find in this file all the strings of your program in the default language.

Thanks to that file, you will be able to create automatically (with Catcomp or Flexcat) :

- the translation file ('xxxxx.ct' file) in which you'll find the strings in a foreign language.
- the program source file corresponding to your favorite language.

To all beginners who still don't know how to localize a program, i suggest to use the now well-known Flexcat program (by Jochen Wiedmann) which will provide you the possibiliy to generate code in most languages.

6.5 In line help generation

You will be able to make your online help directly from your favorite interface builder (MUI-Builder of course!!!).

You are able to attach a help text to each MUI-object created by MUI-Builder.

Then MUI-Builder will automatically create a hypertext documentation in AmigaGuide Format (that you will be able to view by clicking the 'View' button).

As with code generation, you create only one part of the documentation for insertion directly into your previously written documentation.

You can edit the text of the documentation :

- the main text with the `App Node` button
- a window text with the `Window Node` button
- an object text with the `Object Node` button

You are able to create the documentation :

- for the application with the `Generate whole Doc` Button
- for a window with `Generate win Doc`
- for an object with `Generate obj Doc`

By double-clicking on an object or a window name, you can edit the title of the associated help-text, and view it. (same as the `Edit` buttons).

When you create the source code, MUIBuilder will generated a MUIA_Help_Node attribute to each object whose text is non-empty and which you declared as generated. The generated file is designed to be an online-help ... not really a real documentation ...

6.6 Some details to know about

There are a lot of details (tips and hints !!) that you will find in this section, and could be (in my humble opinion) really useful ...

6.6.1 Special Chars

In EVERY text of your interface, you can insert the following special characters :

`\n`	newline
`\r`	carriage return
`\t`	tabulation
`\e`	escape
`\\`	backslash character `\`
`\"`	a double quote
`\xNN`	the character of ascii code NN (in hexadecimal)
`\nnn`	the character of ascii code nnn (in octal)
`\c`	c if c is any other caractere

Here are some examples :

`\033b`	to print a bold text
`\033n`	to come back to normal text
`\0338`	to display a white text
`\033c`	to center the text
`\033l`	to justify left
`\033r`	to justify right

Note that you have to type '`\"`' instead of '`"`' when you want double-quote. Otherwise, you'll get errors when compiling. Moreover '\033' is the same as '\e'. Don't forget to enter '\033' instead of '\33' !!!

6.6.2 Objects lines

To line up some objects, you just have to include them in a multi column (vertical lining) or a multi-lines (horizontal lining) group (most of the time multi column)

Some objects (Slider, CheckMark, String) have a title proposed by MUIBuilder. These titles don't exist in MUI: to give you this possibility, MUIBuilder include the object and its title in an horizontal group. So you won't be able to line up these objects, if you use this feature : just use some labels `+` objects without title in some multicolumn groups.

6.6.3 Resizable interface

One of the frequently asked questions is : `"after i added some objects in my window, the GUI is no longer resizable"` : only one answer to give : use `Space` objects and add them in your GUI !

7 Objects

MUIBuilder enables you to create every MUI objects that you would like to include in you application. These objects can be created or modified in the main creation window (after you pressed the `New Window` button in the main window).

7.1 Area Object

Every MUI objects that inherits from the Area object include in their GUI a register group. The second page contains some attributes of the Area object that could be set at initialization.

- `Hide` : The object won't appear in the GUI, except if it is alone in the window.
- `Disable` : the object gets a ghost pattern and doesn't respond to user input.
- `Input Mode` : enable to turn an object to a button.
- `Phantom Frame` : The frame will not appear.
- `Frame` : Defines a frame for the current object.
- `Background` : Defines the background you want in your object.
- `Control char` : the character needed to activate a given object.
- `Title Frame` : a title added in the top of the frame of the object

Some attributes are not provided for some object : this is not a bug ... it's a feature !!! I don't want the user to be able to do something wrong in his GUI (such as making a button with a String frame !!!!).

7.2 Application

Your application is the root node of the dependency tree representing your complete graphic interface. By clicking the 'Appli' button in the main window of MUI-Builder, you can set:

- The 'Base' of the application i.e. the name of the AREXX server of your application
- the author's name
- the application's name

- the version
- the copyright text
- a description of your application

Moreover, you'll find in this window :

- A menu button, that will open the menu creation window, that will enable you to create the application Menu (see Section 7.24 [Menus], page 31). This menu can't be tested from MUIBuilder, since it must be attached to an application ! And i can't create a new application in the MUIBuilder application itself !!!!
- The possibility to add notifications on the application (see Chapter 8 [Notification], page 33).

It can have only one sort of child object: the Window (see Section 7.3 [Window], page 20).

7.3 Window

You all already know what a window is! But the problem is learning how to make one with MUI-Builder...

The MUI-Builder 'Window attributes' window has two distinct pages, one named 'Creation' and the other called 'Attributes'.

1. The `Creation` page contains 2 lists:

   In the fist one, you build your GUI itself. The display offers a hierarchical tree. To fold or unfold a node of the tree, you just have to double-click on the little arrow at the left of the object. Thanks to the buttons between the two lists, you can edit/copy/move the objects of your GUI. Moreover, the menus offer two items `Fold` and `Unfold` which will enable you to fold or unfold completely the whole tree. In the two most left columns appear two letters : `H` (for Help) and `G` (for generated) : they enable the control respectively from the node generation in the guide documenation, and for the label generation in the source code. If you double-click on one of this letter, it toggles the generation either of the AmigaGuide node, or of the label in the source.

   - The goal of the right-most list is to store temporarily some objects. (see Section 7.3.1 [Liste temporaire], page 21).

Groups (see Section 7.4 [Group], page 21) are the basics elements to which all other objects (see Chapter 7 [Objects], page 19) will be placed. Every time you create an object you will have to declare it as a child of a specific group. You may specify this group by selecting it in the list gadget.

2. In the 'Attributes' page, you can find :
   - Window label and title.
   - The window attributes themself :
     - `AppWindow`: if you want your window to be a appwindow
     - `Borderless`: if you don't want a border
     - `Depth gagdet`: if you want a depth gadget
     - `Size gadget`: if you want a size gadget
     - `BackDrop`: to have a window in the background
     - `Close gadget`: to put a close gadget in the window
     - `Dragbar`: enables the window to be dragged
     - `Open at Init`: (IMPORTANT) this specify if you want your window to be opened when your application will be created.

7.3.1 Temporary List

In this list you can put objects you want to move from a group to another one, or from a window to another one. The list content won't be deleted when you delete your application, or when you load a new one. In this way, you will be able to transfer an object from an application to another one ...

7.4 Group

When you create a new window, it already has a group default child. This child object is the root group. You define a window by attaching children (see Chapter 7 [Objects], page 19) to this root group. Note that other groups may also be children to the root group. You must define the attributes for each group:

- `Horizontal`: The objects of the group will be placed horizontally.
- `Register`: The group will show only one of its children at a time. After selecting this attribute, you must go to the 'Register' entries page to chose the names of the pages of the register.
- `Same Height`: The all chidlren of the group will have the same height.

- `Same Width`: The all children of the group will have the same width.
- `Same Size`: The all children of the group will have the same size.
- `Virtual`: The group will be virtual.
- `Columns`: Format the group in columns. Enter the number of columns in the string gadget.
- `Lines`: Format the group in rows. Enter the number of rows in the string gadget.
- `Spacing`
  - Horizontal : enables you to control horizontal spacing between the objects.
  - Vertical : enables you to control vertical spacing between the objects.

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.5 List

To completely define a list, you must specify :

- its label
- if 'double click' is enable or not
- if the multiselection is enable
- If the list is in input mode or not
- If the vertical size must be adjusted at creation or not
- If the horizontal size must be adjusted at creation or not
- The part of the list you want to show to user (activate):
  - No special positionning
  - Show the first entry
  - Show the last entry
- the type of the list
  - a standard list
  - a floattext list, that allows to show texts. You can enter a text in the corresponding string gadget `Float Text`)
  - a list of volumes (volumes `+` assigns)
- The function hooks can be defined too, but they are not absolutely necessary to make the list usable (see MUI-Autodocs!) :
  - Construction : called when you insert an object in the list

  - Destruction : called when you delete an object in the list
  - Compare : called when you sort the list
  - Display : called when you display an object
  - Multitest : called when you perform multiselection
- The format of the list enables to define multicolumn lists (not tested directly in MUIBuilder).
- The list title.

Note that there is an other type of list called a DirList (see Section 7.6 [DirList], page 23) This type of list can be created by clicking on its button in the 'Object Choice' window.

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.6 DirList

The Directory list shows the files and directories in the selected directory. The possible options are :

- `Drawers onlyt`: shows ONLY directories
- `File Only`: shows ONLY files
- `MultiSelection`: enables a multiselection of the files in the list
- `Reject Icons`: don't show the '.info' files
- `Sort High-Low`: reverse sorting order
- `Sort Type`: choice of the sort key (Name, Date, Size)
- `Sort Dirs` : Three possibilities :
  - Directories appear at the head of the list
  - Directories appear at the tail of the list
  - Directories are mixed with file (following `Sort Type` option)
- The directory to read
- The pattern to accept
- The pattern to reject

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.7 String

To completely define a string gadget, you must specify :

- Its title (that you can replace with your own)
- its label
- the initial content of the string gadget
- a string containing all the letters accepted by the gadget
- a string containing all the letters refused by the gadget
- the maximal length of the string

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.8 Button

To completely define a button, you must specify:

- its label
- The text that will appear in it

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.9 Label

To completely define a label gadget, you must specify :

- its label
- the text that appears on screen

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.10 Cycle Gadget

To completely define a cycle gadget, you must specify :

- the list of its entries
- its label

If you specify a control char for a cycle gadget, please add a label with the same control char, so that the user could see which character he has to use !

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.11 Radio Buttons

To completely define a radio gadget, you must specify :

- the buttons' list : to add, delete, rename the entries of this list, just use the buttons at the right of the list.
- its label

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.12 CheckMark

You can specify wether you want a Title or not (like the strings and sliders) before the CheckMark. If you want a title, you can specify it in the 'title' string gadget Don't forget to specify a label.

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.13 Image

To define an image for this gadget, you must select the image that you wish to be displayed.

Possible choices are :

- `Free Vertical`: the image will resize vertically.
- `Free Horizontal`: the image will resize horizontally.
- `Input Mode`: the user will be able to select the image.
- `Fix Height`: set the image height to a constant specified in the associated string gadget. This makes `Free horizontal` useless.
- `Fix Width`: set the image width to a constant specified in the associated string gadget. This makes `Free vertical` useless.
- As with other objects, you must specify the label.

Two notes about the image support in this version 2.2 :

- the popasl gadget provide the ability to enter an image file (which will be displayed using datatypes)
- It seems there is a problem with dynamically added images in MUI : they seem to not be displayed right. So, when you'll test your application, sometimes the image won't appear. Despite of this, the code should be correctly generated.

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.14 Slider

You can specify:

- if the current level must be displayed ('Slider Quiet')
- if the slider must be displayed in the reverse mode
- if the slider must have a title or not

Then you must specify:

- the maximum value
- the minimum value
- the initial value

You can also set:

- the title of the object
- . . . and its label !

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.15 Gauge

To define a gauge, you must set:

- its orientation (horizontal or vertical)
- if you want to set the height of the gauge to a constant value it must be specified in the associated String Gadget. This is especially useful with horizontal gauges.
- if you want to set the width of the gauge to a constant value it must be specified in the associated String Gadget. This is especially used with vertical gauges.
- if its value must be 'divided' (then give the associated value)
- its maximum value
- the info test of the gauge : this little text will be displayed inside the gauge itself. To include in this text the current level of the gauge, just put a `%ld` in the info text.
- its label

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.16 Text

To define a Text Gadget, you must specify :

- `Text_SetMax`: the maximum size of the gadget will be its initial size.
- `Text_SetMin`: the minimum size of the gadget will be its initial size.
- `Label` : Its label in the generated source code

The text must be written in the associated string gadget and can contain every Special character (see Section 6.6.1 [Caract˙res sp˙ciaux], page 16) you can find in a C source code.

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.17 Scale

A scale gadget must be used with a Gauge gadget to display a graduation beside it.

You only have to specify its orientation. Note that only the horizontal orientation is available at this time in MUI.

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.18 Proportionnal Gadget

Set the following options :

- `Horizontal`: the gadget must be horizontal
- `Fix Width`: set the width to a constant specified in the associated string gadget.
- `Fix Height`: set the height to a constant specified in the associated string gadget.

Then you should specify :

- the number of entries
- the number of the first entry
- the number of visible entries

As always : don't forget the label !

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.19 Space

This object allows you to insert a space between two other objects so the window can be resized. It can be horizontal, vertical or vertical and horizontal. You can enter the spacing value for both the horizontal and vertical space.

7.20 PopAsl

This object enables the inclusion of standard ASL requesters in your GUI. The attributes you can set are:

- The image you prefer
- The type of the requester which could be :
  - File requester
  - Font requester
  - Screen Mode requester
- There are some hooks you can set:
  - Start hook : called at the requester creation
  - Stop hook : called when you close the requester
- the label of the requester

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.21 PopObject

The Popobject provides a list and an image associated with any of the MUI object you can create with MUIBuilder. This object will pop up when you'll click on the popup image. It will appear in the tree hierarchy display as any other object of the GUI, and can be edited in the usual way ... even if it is a group which contains many other objects.

To define this object, MUIBuilder provides :

- A list with the possible images
- Some attributes :
  - `Follow` : The Pop object follows the window
  - `Light` : The Pop Object is displayed in a borderless window
  - `Volatile` : The object disappear when you click on the pop image
- The hooks provided here are :
  - Open Hook : called at the opening of the pop window
  - Close Hook : called at the close of the pop window
- The label of the object

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.22 Rectangle

This object is little bit complex, because he offers in fact more than only one object. It enables the creation of :

- A rectangle area in a window
- A horizontal bar, with or without a title to separate objects in a GUI
- A vertical bar to separate objects in a GUI

You can define the height and width of these objects too.

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.23 Color Field

This simple object is designed to show a colour on screen. You just have to define :

- If you want to fix its height or not
- If you want to fix its width or not
- the colour of the field

This object inherits from the Area object (see Section 7.1 [Area Object], page 19).

7.24 Menus

A menu can be attached to a window or the application object. It is made of several titles, sub menus and items. When built, it is similar to a tree hierarchy, and is displayed in this way in the menu creation window.

The only limitation is the depth of the tree which is limited to 2. (In fact : a sub menu can't have a sub menu)

Moreover:

- The titles, submenus and items can be activated or not using the `Enable` checkmark
- The items have acces to some more attributes :
  - It can be Checkable
  - If checkable, you can set its initial state
  - You can define a shortcut for any menu item

8 Notifications

The events notifications is to my mind the best new feature provided in MUIBuilder 2.0. It's the first time in the amiga history :-) that such a feature is included in an interface builder.

The goal is to provide the possibility to define links between events and actions. The events come from MUI Object : you will define for a given event the action that will be performed, and on which object it will be performed.

Once a link event-action is builded, it can be tested directly by MUIBuilder, by clicking on the `Test` button on the creation window. Every action which don't use the application object or/and external program variables can be tested without any problem in MUIBuilder.

The notification window offers :

- The list of the possible events generated by the object whose name appears at the top of the window
- The list of the objects on which an action can be performed. To help you to find a given object, the list entries are organized in a hierarchical order. The window containing the object you are editing appears at first position in this list, then comes the application object, and at least all other windows in the application.
- Once you have selected a destination object, the third list contains all actions which can be performed on this object. Some of these actions need some arguments . . . MUIBuilder will ask for them if needed.

To add a notification on an object you are currently editing, you need either to double-click on an element of the actions list, or to select the `Add Notify` button.

The application object collects all actions to make on your program itself (for example external variables modifications, functions call . . .)

When you validate an action, it is possible that it needs a parameter : if so the program will open a request window. The parameter can be of the following types :

The choice of an object to which you want to add a notification can be performed in two different ways :

- For a window : by clicking on the `Notify` button of the `Attributes` page
- For an object: by selecting the given object in the creation window and clicking on the `Notify` button.

8.1 Functions

If an action parameter is a function `Hook`, MUIBuilder ask you for the function name. It opens a window where all the existing functions appear. You can either chose one of the existing functions or create a new one. When you generate the source code, MUIBuilder will generate what is needed to reference these functions.

8.2 Variables

Some actions need a variable as parameter. To get this variable name, MUIBuilder opens a window, in which you can find the list of existing variables. You can either chose one of the existing function or create a new one.

8.3 Constants

A constant can have several types:

- A boolean : 2 possibles values True/False
- A string
- An integer
- A character

Depending on the type of the constant, one of the gadgets of the request window will be activated. You'll have to enter the needed value in this gadget.

9 Future improvements

Many features need to be implemented later to make of this program of the best ever seen !!! :-)))) Most of them should have been included in this version ! But unfortunately i had no more time to improve it :-(... And as i'm not sure to find enough time to add these features in the next months : I'll release it now !

Here are the enhancements i'd like to be able to see in the future versions :

- Some objects to improve (such as Space, PopAsl, Image ...)
- Some objects to add (such as palette)
- Arexx support (with some functions to have access to MUIBuilder internals)
- Probably a support for MUI custom objects, if nobody releases something similar
- I want to improve the notification support, to be able to build more complex notifications, keeping the same GUI . . .
- Improvement of the external code generators : add a GUI and many many options !
- Add preferences (local and global) to generators
- To have an interface between MUIBuilder and text editors would be great
- I'm waiting for Drag&Drop in MUI to improve the GUI construction
- The object buttons will probably become images
- The save files will be improved to enable to edit them easily
- The catalog generation will be improved : the strings added by hand won't be deleted anymore

As you can see, i have some more months of work before i can begin a new project !!! I'm always waiting for some suggestions !!! :-)

Greetings

MUIBuilder is **dedicated to the memory of Pierre Carrette**, one of the best amiga programmers I knew ...

I'd like to thank here every people who helped me and contributed to MUIBuilder (no special order) :

- **Lionel Vintenat** for the idea of the generic code and the implementation of the 'E language' module.
- **Albert Weinert** for the implementation of the OberonII module.
- **Stefan Schulz** for the Modula2 module.
- **Stefan Sommerfeld** for the assembly generator.
- **Dirk-Michael Brosig** for his GCC-only module.
- **Bernd Noll** for his great help when debugging MUIBuilder, and his really BIG :-) bug reports !!
- **Andreas Jung** for his fantastic help and bug reports !
- **Pierre Carrette** for all his ideas :-) (and BrowserII !!!!)
- **Gael Marziou**, **Pascal Pensa**, **Christian Brandel**, **Daniel Murrel** for their carefull tests and suggestions.
- **Christian Brandel** for the german documentation, and the german catalog.
- **Pascal Rabier**, **Mike Manzaro**, **Daniel Murrell** for their corrections on the english documentation !
- **Tobias Ferber** for his superb MagicWB icon and the objects images.
- **Jérôme Chesnot** for his objects images.
- **Marco Musso** for the italian translation of the catalog.
- **Marcin Orlowski** (Mr Soft/W.F.M.H.) for the polish catalog.
- **Stanislav Kneifl** for the Czech catalog.
- **Soeren Berg Glasius** for the Danish catalog.

I have to thank every people who sent me some bug reports and ideas : without all of you, this program would not be what it is . . .

Index

A

Application Object 19
AppWindow 12
Author 5

B

Button Object 24

C

Catalog 14, 15
CheckMark Object 25
Code generation 11, 12
Colord Field Object 31
Configuration 11
Constants 34
Cycle Object 25

D

DirList object 23
Distribution 5

E

Events 33
External generator 13

F

Flexcat 14, 15
Function Hooks 34

G

Gauge Object 27
General Principles 3
Group Object 21

I

Icons 11
Image Object 26

L

Label Object 24
Labels 14
List Object 22
Locale 11

M

Menus 31

N

Notifications 33

O

Objects 19
Objects lines 17
Objet PopAsl 29
On line help 15
Options (code generation) 13

P

PopObject 30
Proportionnal Object 28

R

Radio Object 25
Rectangle Object 30
Requests 11
Resizable interface 18

S

Save 12
Scale Object 28
Slider Object 26
Space Object 29
Special Chars 16
String Object 24

T

Temporary files 13

Temporary List 21
Test Window 33
Text editors 11
Text Object 28
Tree display 11

V

Variables 34

W

Window Configuration 11
Window object 20

Short Contents

1 Introduction 1
2 Principles 3
3 Distribution 5
4 Disclaimer 7
5 Tutorial 9
6 Use of MUIBuilder 11
7 Objects 19
8 Notifications 33
9 Future improvements 35
Greetings 37
Index 39

Table of Contents

1 Introduction 1

1.1 Advantages of MUIBuilder 1

2 Principles 3

3 Distribution 5

4 Disclaimer 7

5 Tutorial 9

6 Use of MUIBuilder 11

6.1 MUIBuilder Preferences 11
6.2 Application save file 12
6.3 Code generation 12
    6.3.1 Code generation : Options 13
    6.3.2 Application code 14
    6.3.3 Object code 14
    6.3.4 Remove a label 14
    6.3.5 Ajouter un label 14
6.4 Localisation 14
    6.4.1 Catalog 15
6.5 In line help generation 15
6.6 Some details to know about 16
    6.6.1 Special Chars 16
    6.6.2 Objects lines 17
    6.6.3 Resizable interface 18

7 Objects 19

7.1 Area Object 19
7.2 Application 19
7.3 Window 20
    7.3.1 Temporary List 21
7.4 Group 21
7.5 List 22
7.6 DirList 23

7.7 String 24
7.8 Button 24
7.9 Label 24
7.10 Cycle Gadget 25
7.11 Radio Buttons 25
7.12 CheckMark 25
7.13 Image 26
7.14 Slider 26
7.15 Gauge 27
7.16 Text 28
7.17 Scale 28
7.18 Proportionnal Gadget 28
7.19 Space 29
7.20 PopAsl 29
7.21 PopObject 30
7.22 Rectangle 30
7.23 Color Field 31
7.24 Menus 31

8 Notifications 33

8.1 Functions 34
8.2 Variables 34
8.3 Constants 34

9 Future improvements 35

Greetings 37

Index 39