MUI - MagicUserInterface

A system to create and maintain graphical user interfaces

- User Documentation -

Version 1.4
19-Oct-1993

Stefan Stuntz

1 Introduction

1.1 The concepts behind MUI

MagicUserInterface (also known as MUI) is a complete system to create and maintain graphical user interfaces. The creating GUIs has been a big problem for a very long time. Mainly because the programmer got only a minuscule amount of support from the operating system. Beginning with Kickstart 2.0, the '`gadtools library`' was a step in the right direction, however, even using this library to generate complex and flexible interfaces remained difficult and still required a great deal of patience.

Today there are tools available that make the use of '`gadtools library`' much more simplified, but even these alternatives are not often satisfying.

The largest problem in existing tools for the creation of user interfaces is the inflexible output. Most of the programs are still using built-in fonts and window sizes, thus making the use of high resolution graphics hardware adaptors nearly unbearable. Its been said that Amiga users have had to live with such similar shortcomings all along. Even the preference programs on the Workbench are still only using the default font, topaz/8!

MUI corrects all these disadvantages! The central scheme behind MUI assumes that only the user (and not the programmer) of an application knows how the program he is using best fits his personal needs, and that of his computer system. Because MUI applications don't contain any absolute values for sizes or positions, the programmer instead only defines objects and groups of objects. Such objects are defined on run time by MUI according to the users settings.

Consequently, an MUI application gives the user **many** more important advantages:

- Font sensitivity

  In MUI Its possible for the font to be set in every application. No more times where the A2024 users had to suffer from the programs that only used the tiny topaz/8 font. Even better, MUI gives the user no restrictions on which fonts he may use, especially proportional fonts! The proportional fonts make a program much more appealing and even reduces the space a program's window uses.
- Changeable window sizes

  All MUI windows have a sizing gadget which allows users to change the window size until it suits their needs. The smaller a window becomes, the closer control items within the window come

together. The larger the window, the more space that will be used for displaying information (e.g. in list-views). The size and position of every window can be saved, thus giving you your favorite setting every time you start the program.

– Flexibility

  Almost all elements can be changed by the user regarding their own personal tastes. The user can define the thickness of borders, how the scroll-bars look, mhich images have to be used, and how much space should be inserted between the lines of a list-view. MUI gives the user a lot of options to change the look and feel of an MUI based application. See Section 3.1 [PRF˙INTRO], page 7 for details.

– Controlling by keyboard

  Most of the time it is expected that graphical user interfaces (GUIs) and of course MUI applications as well, are controlled by the mouse. However, many users prefer the use of the keyboard for faster execution of operations, and because it can be more comfortable. Because of this, all MUI objects (e.g. string gadgets, radio buttons, or list-views) can be controlled by the keyboard as well as by the mouse! You can even put away your mouse completely if you wish! Its no longer needed!

– System integration

  MUI applications cooperate with the operating system in many ways. Every program can be iconified and uniconified by pushing a gadget or by using the Commodities exchange program. Furthermore, every MUI application has an ARexx-Port that allows you full remote control (and more) over the user interface.

– Adjusting to its environment

  It doesn't matter which screen or screen size your MUI applications run on. Workbench or public, 640x200 or 1280x1024 pixels, 4 or 256 colors, it doesn't matter! Every application can be made to open on any screen, and adapts itself to it's environment.

All of the MUI settings listed above (and more) can be changed by the user via the MUI preferences program. This can be set for every program in one setting (global) or can be set for each and every single application.

1.2 System Requirements

MUI requires version 2.0 of the Amiga operating system or higher.

Kickstart 1.3 is *not* supported; this operating system has been considered to be obsolete.

The use of MUI on a harddisk is *highly* recommended, although floppy disk usage is still possible. Due to the modular concept behind MUI the first application startup may last "a little" longer.

MUI does not require a special processor, but of course the faster CPUs make life easier. Many complex calculations are needed for the management and layout of the objects, so a "base" 68000 based machine could be a tad slow.

MUI applications can run on machines only equipped with 512K of free RAM, but could become obsolete quite soon. One Megabyte (1024K) RAM should be sufficient even for the most complex MUI applications.

1.3 Installation

MUI is distributed together with the Section 4.8 [POL'INSTALLER], page 26 program from Commodore. Therefore making installation a breeze! Just double-click on the '`MUI-Install`' icon and the installation procedure is on it's way.

2 Using MUI Applications

2.1 Windows

As mentioned previously, nearly all MUI windows are resizeable. This allows the user to determine if he wants his control items (i.e. buttons, list-views, a.s.o.) small and space saving, or bigger and easy to use. It would be very annoying to adjust the size and position every time an application appears, all MUI windows remember their size and positions and uses these values when the application appears again. This is true for the "normal" window position and size as well as for the values of the window in a "zoomed" state. (after hitting the zoom gadget)

After rebooting the data for the windows are usually lost, if you have not saved them by hitting the '`Save`' button in the preferences window. By doing so, all data regarding the windows are saved and are available even after a reset.

In addition to the depth and soom gadget there is a third button in the title-bar of every MUI window. This button is used to iconify the whole application. All windows (and screens if available) in the application are closed and a little appicon appears on the Workbench or default screen. Double-clicking on this appicon makes the program open its window(s) (and screen) again.

2.2 Keyboard control

All control items in an MUI window can be controlled completely by the keyboard. All the known keyboard shortcuts (marked by an uderscore character) are supported. However, this method is limited if used with list-views or cycle gadgets.

Because of this the `TAB` cycling (up until now only used for string gadgets) has been made available for MUI applications. You can activate every object (not only string gadgets) by hitting the `TAB` key. As soon as you activate an object it can then be controlled by the keyboard.

– Button gadgets

  `Return` is the equivalent to clicking the gadget with the moust button. Pressing and releasing a button is handled in a different way. If you push a knob and then wish you cancel this action, you may do so by additionally pushing `Shift` before releasing the `Return` key.

– Checkmark gadgets

  An active checkmark gadget can be controlled by `Space` or `Return`. The value of this gadget is toggled every time you press the corresponding key.

– Slider gadgets

  The value of the horizontal or vertical slider gadgets can be changed by using the four cusor keys. Using qualifier keys additionally allows you to change the increase/decrease amount of the sliders.

– Cycle gadgets

  An active cycle gadget can be switched by using the cursor keys. The `Return` key pops up a popup menu (as long as you did not disable this feature in the MUI preferences program).

– Radio-Buttons

  Radio buttons are also controlled by the cursor keys.

– List-Views

  In an activated list-view the cursor can be moved line by line using the cursor keys and together with the corresponding qualifier keys pagewise or even to the top or to the bottom. The `Return` key simulates a double-click.

  If multiselecting is allowed in a list-view, you can select the different items by using the `Space` key.

– Windows

  For applications opening several windows simultaneously you can switch from one window to another by using the `Alt-Tab` key or the `Shift-Alt-Tab` key respectively. If the window has a close gadget , you can hit the `ESC` key to close the window.

All information refers to the default settings. When using the preferences program, you can change all the keyboard combinations until the suit your needs (see Section 3.10 [PRF˙KEYBOARD], page 17).

2.3 Cycle gadgets

Besides the MUI cycle gadgets supporting their “normal” function (next entry by clicking on them, previous entry by holding `Shift` additionally), offers a menu feature. This popup menu appears as soon as the text section of the cycle gadget is hit, then allows a quick and easy selection of one of the listed entries.

The behavior of how the popup menus function can be changed in the Section 3.4 [PRF˙LISTVIEWS], page 9 section of the MUI preferences program.

2.4 Commodities interface

Every MUI application ties itself in the system as a commodity. This is nice in that the user can control any MUI application via the '`Commodities Exchange`' program, i.e. MUI Applications can be iconified or canceled.

2.5 built-in ARexx Port

Every MUI application is able to receive commands via the built-in ARexx port. Here are some default commands which are understood by every program:

- QUIT
  Ends the application.
- HIDE
  Hides (iconifies) the application
- SHOW
  Shows (pops up) an iconified application.
- INFO ITEM/A
  According to the given parameter the result string is filled with the following contents:
  - "title" Title of the application
  - "author" Author of the application
  - "copyright" Copyright message
  - "description" Short description
  - "version" Version string
  - "base" Name of the ARexx port
- HELP FILE/A
  A list of all ARexx commands available for the application is written into the given file. In addition to the default commands an MUI application can (and of course should) support many application specific commands. The help list will contain these commands as well.

Some example scripts can be found in the '`Rexx`' drawer on the main directory of the distribution.

3 Preferences-Programm

3.1 Introduction

With traditional applications, the user usually has no or only very limited possibilities to influence the look and feel of an user interface. With MUI, interfaces are a lot more flexible. The programmer only specifies very few things about the outfit of particular gadgets, what actually is displayed on screen depends on the users preferences setting.

To adjust these settings, MUI comes with a preferences program called '`MUI`'. After installation, this tool can be found in the system's preferences drawer.

3.2 Main Window

The main window of the preferences program consists of three single regions; an application list on the left side, some configuration pages right beneath, and at the botton a group with four action buttons.

The application list allows adjustment of different preference settings for different MUI programs. It contains the names of all previously started MUI applications. The active entry determines which applications settings you wish to edit. Since almost all settings are usually adjusted globally (for all applications at once), the application list contains one special entry called '`-Global`' at the top. These global settings act as defaults for all MUI applications, but can be overrode by an applications local settings.

Usually you will configure a lot of global settings and only very few application specific changes, e.g. different public screens or iconify icon positions. The '`Default`' button can be used to discard an applications local settings.

Another (rarely used) function of the application list is to display some information. As long as the program is currently running, double- clicking on it's name (or a click on the '`Info`' button) will bring up a requester containing the authors name and the title of the application's ARexx port.

The right side of the window contains a lot of gadgets for all the possible configuration switches. These gadgets are divide into several groups, changing these groups is possible with the cycle

gadget at the top or with some menu entries. Detailed explanation for all groups follows in the next chapters.

At the lower part window border are the '`Save`', '`Use`' and '`Cancel`' buttons, already common in several of the system preference programs. Additionally you'll find a '`Test`' gadget and a '`Test`' menu entry, and can be considered to be the most useful function in the preferences program. It will be handy in the beginning to use the '`Test`' options to play around with the different settings until you have found the configuration that best fits your needs. Pushing the `Test` button makes all currently running applications adopt their parameters from the new values. Thus making it possible to change the settings of a running application and immediately notice the consequences of your actions.

The '`Use`'-Gadget saves the changes to the '`ENV:`' drawer and then ends the preferences program. Please notice that the '`ENV:`' directory usually resides in the '`RAM DISK:`' and a reset discards all the settings made. If you want your settings to be permanent, please use the '`Save`'-Button. In addition to global and application specific preferences settings, window positions of all the applications will also be saved.

If you hit '`Cancel`', all changes will be discarded. Applications that have already adjusted themselves to the new values because you hit '`Test`' automatically return to their previous settings. According to the system's preference programs, no safety requestors appear.

3.3 Spacing Page

The spacing page contains some sliders to adjust the spacing between several basic objects. More sophisticated spacing adjustments can be found on the Section 3.5 [PRF˙FRAMES], page 9.

The first four values identify the distance between a windows border and its contents. Usually the values are identical for '`Left`' and '`Right`' as well as for '`Top`' and '`Bottom`'. Thus the reason for these gadgets being connected. As soon as you move the '`Left`' or '`Top`' slider, the '`Right`' or '`Bottom`' sliders are moving automatically. Nevertheless, to allow different values, this connection is single-sided. If you move '`Right/Bottom`' the value for '`Left/Top`' remains unchanged.

'`Group Horizontal`' and '`Group Vertical`' determine how many spacing pixels are inserted between horizontal/vertical groups.

'`Radio`' slider gadgets influence the look of the (rarely used) radio button objects.

3.4 List-View Page

The '`Leading`' value determines the number of additional pixels that are inserted between the lines of a listview to improve its readability. According to the font used and your personal taste, it could make sense to set higher values, especially if you use small fonts such as '`topaz/8`';

'`Smoothing`' enables smoothing of list-views. This setting effects the position of the list in that it doesn't follow the scroll-bar moves immediately, but instead is delayed for some short amount of time. The result is smoother scrolling! Selecting zero (0) prevents the list-views from doing any smoothing at all.

In multi select list-views the user can select either '`shifted`' or '`Always`'. '`Shifted`' enables the usual multi select mechanism, i.e. you have to hold down the `Shift` key while you are selecting the entries. If you select '`Always`', you don't have to hold down the `Shift` key.

'`Refresh`' determines the kind of the list-view refreshing. '`Linear`' refreshes the lines as usual from top to bottom, '`Intermixed`' refreshes the lines intermixed, what should result in a "nicer" appearance. Especially on slower machines!

The position of the arrow gadgets at the scroll-bars are controlled with the '`Arrows`' cycle gadget, and three different options are available.

The cycle gadgets of MUI supply a popup menu for easier usage, which is opened as soon as you hit the gadget. It allows an easy and quick selection of the desired entry.

'`Level`' determines the minimal number of entries that are needed to supply a popup menu. If you don't like popup menus, all you have to do is select a high value and you'll never see them.

Usually popup menus appear directly under the gadget. For fast usage and less mouse movement you can configure the popup menus in a way that let the active entry appear just below the mouse pointer. For this reason there are the two options for the '`Position`' gadget.

3.5 Frames Page

Frames are important elements to create a straightforward graphical user interface and to separate single groups from each other. However, frames are a matter of taste, therefore you can change their appearance in several ways.

In usual (non MUI) Amiga programs, all frames have vertical lines with double the thickness of horizontal lines. This outfit was introduced lots of years ago when 640x256 with a pixel aspect ratio of nearly 1:2 was the standard resolution. Now days, hi-res graphic adapters and flicker-fixers often allow an 1:1 aspect ratio, eliminating the use of double width frames. MUI allows you to change the thickness of the frames using the '`Thickness`' cycle gadget.

Framed groups can have a title, and the color of this frame title can be set via the '`Title Color`' gadget. Currently three settings are available: Black, White and 3D.

The title of the group is centered horizontally at the upper part of the frame. Selecting '`Title Pos`' changes the vertical position of this title text. '`Centered`' centers the title text vertically, '`above`' sets the base line of the text to the position of the frame.

When a programmer creates a MUI application, he doesn't set the outfit, but only the type for the frames. For example, a button gadgets gets a button frame and a string gadgets gets a string frame. The appearance of the frames is determined by the user. Therefore all possible frame types are collected in a list:

- Button frame

  for usual button gadgets, as for example for the '`Edit`' button below this list.
- Image frame

  for small buttons that contain nothing but an image, e.g. the arrow gadgets in a scroll-bar.
- Text frame

  for text gadgets on which can neither be clicked upon nor can be edited and are used only to display information, e.g. status lines.
- String frame

  for string gadgets.
- Read list

  for list-views that only display a list and that can't be clicked upon.
- Input-List

  for list-views in which the user can select entries, e.g. all list-views of the MUI preferences program.
- Prop frame

  for all prop gadgets, used for example within scroll-bars and sliders.
- Group frame

  to group objects, for example the buttons for the subwindows of the MUI preferences program are surrounded by a group frame.

– PopUp frame
  frames a cycle gadgets popup menu.
– Virtual frame
  is used in virtual groups.

A double-click on an entry or the '`Edit`' opens a frame edit window to adjust look and spacing of a particular frame. You can open as many frame windows as you want.

3.5.1 Frame Configuration

In the frame window of MUI you can select one out of a list of predefined frames. The different possibilities are displayed in the '`Type`' cycle gadget. There exist two versions of every frame, a '`Raised`' an '`Recessed`' look is possible.

Additionally, for every frame type, the distance between the frame itself and its contents can be changed by setting the appropriate slider gadgets.

3.6 Images Page

Graphical user interfaces often use little images, e.g. arrow buttons or slider knobs. A MUI application usually doesn't define these images itself, it just says it wants an arrow and MUI supplies this arrow. This allows the user to configure, how his arrows actually shall look like.

Here is a list of all available images:

– '`ArrowUp`', '`ArrowDown`', '`ArrowLeft`', '`ArrowRight`'
  Four arrows for the four different directions.
– '`CheckMark`', '`Radio-Button`', '`Cycle`'
  Used as image for the well known user interface elements.
– '`PopUp`', '`PopFile`', '`PopDrawer`'
  Images for popup buttons besides string gadgets. '`PopUp`' is used if neither a file nor a drawer is wanted.
– '`Drawer`', '`HardDisk`', '`Disk`', '`Chip`', '`Volume`', '`Network`', '`Assign`'
  Default images for the entries of a file requester.

– ‘TapePlay’, ‘TapePlayback’, ‘TapePause’, ‘TapeStop’, ‘TapeRecord’

  Used within tapedeck applications.

– ‘Prop-Gadget Knob’

  Image for the knob within the prop gadget

A double click or the ‘Edit’ button below opens the image configuration window for the active entry.

3.7 Background-Select

You can select many different backgrounds for MUI, e.g. one background for windows, another for the button gadgets and another for list-views. If you like colorful user interfaces, you’ll surely be happy with MUI.

– ‘Window’

  The window background is used on every place that doesn’t match another background type, especially where no objects exist.

– ‘Requester’

  Background for MUI requesters, e.g. for the ‘About’ requester of the preferences program.

– ‘Textfield’

  Framed text fields (e.g. status lines) use this background type.

– ‘Button’

  used for buttons containing text and for cycle gadgets.

– ‘Active Gadget’

  a gadget which has been activated by Tab gets this background.

– ‘Selected Gadget’

  a gadget which was selected is marked with this background (besides the inversion of the frame).

– ‘List-View’

  appears behind the lines of a listview.

– ‘List-View Cursor’

  the cursor in a listview.

– ‘List-View Selected’

  selected entries in a listview.

– '`List-View Selected+Cursor`'

  the cursor on a selected entry of a listview.

– '`Prop-Gadget Background`'

  the background in a prop gadget, i.e. the area on which the knob is moved around.

– '`Virtual Groups`'

  background for virtual groups.

A double click or the corresponding '`Edit`' button below the list opens an image configuration window for the active entry. Image configuration is discussed in the next chapter.

3.7.1 Image Configuration

The image configuration window allows configuring a specific outfit for one of MUI's standard images or backgrounds. Several different image types are available:

– built-in Pattern

  Some less complicated pattern are already built into MUI. These patterns are mainly used as backgrounds but might also be useful for some standard images (e.g. for prop gadget knobs).

– built-in Image

  MUI offers a built-in outfit for all standard images, all of them are visible in this list. MUI's built-in images are scalable, they grow with the font size.

– RGB Color Specification (only for Kick 3.0 and above)

  Mainly useful for background settings; you can select a specific RGB color which will be allocated and used for background filling. Since old operating systems don't support pen sharing mechanisms, this feature is only available with Kickstart 3.0 or higher. Of course a multi color workbench (at least 16 colors) is needed.

  Note: Using slightly different grey colors for window, button and listview backgrounds will make your user interfaces really look marvellous.

– External Boopsi Image

  A Boopsi image is some kind of shared system library that contains some commands when drawing is needed. Boopsi images support resizing and are mainly useful for things like prop gadget knobs.

– Special MUI Brush

  MUI Brushs' are traditional ILBM brush files saved with a special color setting that allows MUI to translate these images to different screens with different colors. MUI comes with lots of brushes and you can of course take a paint program and draw your own ones.

Note: These images are not resizeable.

– Alien Datatype Image (only for Kick 3.0 and above)

  Starting with Kickstart 3.0 there are the so called '`datatypes`' available. With these datatypes it is possible to load any picture file, no matter if they are IFF, GIF or other formats. MUI supports that and allows you to use any datatype picture for background or standard images in all applications.

3.8 Font Page

MUI applications may use some user configured fonts:

– '`Normal`'

  This font is used for everything as long as no other font was explicitly specified by the programmer.

– '`List`'

  The default font for list-views.

– '`Tiny`'

  This font is used for small an quite unimportant descriptions. The lettering of the scale object (see '`MUI-Demo`') is using this font for example.

– '`Fixed`'

  If a program needs a fixed width font, it uses this one.

– '`Title`'

  This font is used for the group titles.

If a font field stays empty MUI uses appropriate default fonts, i.e. the system default font for the fixed font and the default font of the screen for all the others.

3.9 Screen Page

MUI applications are able to run on any public screen. The public screen name can be set in the MUI preferences program, and even better, MUI contains kind of a screen manager that allows you to configure any screen with any size and resolution.

The name of A public screen is specified in the string gadget at the bottom of the page. If the selected screen is already available when the application starts up, it will be used. Otherwise MUI scans the list of the built-in screen manager for the desired screen name and opens it if available.

If the wanted public screen either couldn't be found or couldn't be opened (e.g. because of insufficient memory), the application starts on the default public screen (usually this is the Workbench screen).

The biggest part of the window is covered by the screen manager. In a list you'll find the names of all screens already set and two items that correspond to the default public screen and to the Workbench screen. To create a new screen, just hit the '`New`' gadget. A new screen entry is added to the screen list and it's parameters can be configured after a click on the '`Edit`' gadget.

3.9.1 Screen Configuration

This window is the screen manager part of the MUI preferences program. Here you can define all necessary attributes of a new screen, such as title, font, resolution and color palette.

The window is divided into three pages. On the '`Attributes`' page, you will find four string gadgets that allow configuration of a screens public name, his title, the default font and a background picture. The background picture can be any picture file as long as a matching datatype is installed in your system. This feature is only available under Kickstart 3.0 and above.

Besides these essential values, the screen can have the following features:

- '`Auto Scroll`'

  If the screen was set larger than the visible part was defined, it will be scrolled automatically as soon as the mouse touches the screen border.
- '`Draggable`'

  If the screen doesn't have this attribute, it can't be dragged.
- '`Exclusive`'

  The screen cannot share its display with other screens; it will be displayed separately (Kick 3.x only).
- '`Interleaved`'

  This attribute reduces - if set - the flicker, that appears especially during the scrolling of lists on colorful screens (Kick 3.x only).

– ‘Open Behind’

  The screen will be opened behind all other screens.

– ‘System Default’

  The screen is declared to be the system default screen. All windows that are opened on the system's default screen (e.g. shell windows), are automatically routed here.

Size and resolution of the new screen are adjustable on the ‘Display Mode’ page. It offers gadgets similar to the system screen mode preferences program and shouldn't need further explanation.

Finally, the ‘Palette’ page allows configuring a screens color palette. Since MUI programs always use a fixed set of pens or try to allocate RGB colors directly, this is not a palette requester in the traditional way. Instead, you can directly adjust what color should be used for which pens. Therefor, the left side of the display contains a list of possible pens and the right side features some gadgets to allow adjusting any RGB color. With Kick 2.x, there will only be three boring RGB sliders, Kick 3.x users will get a nice color-wheel.

By clicking on a specific pen, you will be able to adjust its color. If you e.g. would like to have a yellow menu, just click on the ‘Menu Background’ pen and configure a yellow color. When the screen is opened, MUI will try to allocate that type of color and use it for the menu backgrounds in that screen.

Usually, you won't want to have different colors for all possible pens. If your screen hasn't enough colors, that's not possible anyway. To make using same colors for different pens easier, the pen list allows connecting several pens to groups. Selecting a pen will also select all other pens in the same group and automatically assign the same color value to all these pens.

You can add a new pen to the current group by clicking on it while holding down the Shift key. If the new pen was in another group before, it's removed and added to the current. If you want to remove a pen from an existing group, just double click on the entry.

Note: Kick 3.x users will have a sample field of the current color beneath the color-wheel. When no color is available on the screen, this field will only show a boring disable pattern and you won't be able to see what you adjust. The palette gadget is absolutely designed for high resolutions with lots of (at least 16) colors which will hopefully become standard quite soon. Users with four color Workbench screens might want to define a colorful public screen for the MUI preferences program.

3.10 Keyboard Page

All keys that are used to control the MUI applications, can be configured in this window. The entries of the list are speaking for themselves, the format is the same as the one used for the input events of the '`commodities.library`'.

Special attention earns the '`Press`' key. MUI needs this key to be able to react on key releases. Therefore this qualifier description has to contain the string '`-upstroke`'.

A new key is only accepted when you acknowledge the string gadget with `Return`. MUI does a syntax check on your inputs and beeps the display when the entered key is invalid.

3.11 System Page

The '`System window`' contains some settings that refer to the cooperation of MUI and the operating system.

Windows can be refreshed either '`smart`' (fast but eating chip memory) or '`simple`' (slower but no need for memory). The '`Window Refresh`' cycle gadget can be used to set one of these refresh types.

When redrawing windows after a resize operation, MUI also offers two possibilities, adjustable with the '`Window Redraw`' gadget. Just choose the one you like better.

The '`Startup`' and '`Shutdown`' string gadgets contain two commands that are automatically executed before and after running an application. It is possible for example to use an external screen manager for opening the desired screen for the application (if the built-in screen manager doesn't fit your needs). Another possibility is to use a simple '`echo`' command to create a log about starting and stopping of applications.

The '`Iconify-Hotkey`' allows you to enter a key combination that iconifys the application (and pops it up again). The format is the same as the one described for the input events of the '`commodities.library`'.

If the '`Iconify-Gadget`' checkmark is set, every window of the application gets an additional gadget in the upper window frame that makes the window iconify as soon as you hit it.

Usually an appicon is created for every iconified application on the Workbench. Double-clicking this icon reactivates the application. If '`Iconify-Icon`' is not set, no AppIcon appears. The application activation can still be done via the iconify hot-key or the '`Commodities Exchange Program`'.

'`Start Iconified`' determines, if the application will be iconified at start-up. This will make sense, for example, if you place some tools into the '`WBStartup`' drawer, to make them available via keystroke.

3.12 General Page

This window is used to set some values used by the MUI preferences program itself, i.e. the preferences of the preferences.

The three string gadgets of this window give the path names which will be used, if the corresponding button in the Section 3.7.1 [PRF˙IMAGES˙EDIT], page 13 is hit.

3.13 CLI Interface

The MUI preferences program supplies a small CLI interface that can be used to give other programs access to MUI's built-in screen manager utility.

The syntax is:

```
NAME,OPEN/S,CLOSE/S
```

NAME: Name of a preconfigured public screen

OPEN: Open screen

CLOSE: Close screen

3.14 ARexx Port

3.15 ARexx Port

The preferences program contains a simple ARexx port, defining the following four commands.

'`SAVE`'
Does the same as the '`Save`'-Button.
'`USE`'
Does the same as the '`Use`'-Button.
'`TEST`'
Does the same as the '`Test`'-Button.
'`CANCEL`'
Does the same as the '`Cancel`'-Button.

Of course the Section 2.5 [USE˙ARexx], page 6 can also be used.

4 Other Topics

4.1 Registration

"MagicUserInterface" is a rather complex product that has always consumed and will continue to consume a large amount of my time. It was a lot of work to finish, but I hope this work will be appreciated and that a lot of MUI based applications with nice and flexible user interfaces will be available soon.

For I cannot afford just working for fun, I decided to release MUI as shareware. The unregistered version is not able to save some of the configuration items of the preferences program. Of course these restrictions won't affect the operation of MUI applications, all important values (window positions, screens, system configs) are usable without registering. Other items will contain reasonable default values. Even with these default values, MUI applications will be more attractive and usable as most other programs.

If you plan to use the full set of MUI's possibilities (different fonts, frames, images, background pattern, ...) with all applications, or if you just feel that MUI is good and should be supported, you should register.

Registered users will be shipped a disk with the newest public release of MUI, along with a personalized, so-called "keyfile" that enables loading and saving of the complete configuration data. This keyfile will work with all future releases of MUI, so you can simply download the latest version from your local bulletin board without having to wait weeks for your update passing through the slow mail channels.

The price for a MUI registration is

```
20.- DM  (D-Mark),
20.- SFr (Schweizer Franken),
90.- FF  (French Francs),
15.- US$ (US Dollar)
```

or an equivalent amount of US$ 20.- (twenty!) in any other currency. "Twenty" because I have to visit my bank and sell your foreign currency which is some kind of expensive here in Germany. If you think your bank does it cheaper, feel free to get some 15.- US$ at home and send them.

The fastest, cheapest and easiest way to register is put the money together with the filled registration form into a letter and send it to

```
Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY
```

Euro cheques (in DM) are also welcome, but please do not send any kind of foreign cheques since the bank charges outrageous fees from both the sender and the receiver. If you really feel you must use a such cheques, please include extra $10.

You can also transfer the money directly to my bank account, but beware: international transfers via banks are very expensive.

```
Stadtsparkasse München, BLZ 701 500 00, Konto 35-169929
```

In any case, be sure to provide me with your name, address, phone number and e-mail address for filing purposes. You can send this information via electronic mail, if you wish. If you don't mind, this data will be stored and processed in electronic form. There is a sample registration form you can fill in the file "OrderForm".

All registrations will be handled as fast as possible and should be finished in about two weeks. Instead of registering MUI directly by the author, you can also use one of the following registration sites. This reduces my mailing costs and makes your registration faster.

– U.S.A. and Canada

```
Robert Blayzor
P.O. Box 807
Johnstown, NY 12095-0807

Phone: (518) 883-5326 (data/fax/bbs)

electronic mail:

InterNet/UUCP: robertb@liquid.albany.ny.us
      FidoNet: 1:267/131.0
     AmigaNet: 40:714/1.0
       C-Link: 911:6150/1.0

Make all payments payable to: Robert Blayzor

Acceptable payment methods: US Money Orders & Bank/Personal Checks
```

```
Personal checks must wait 10-15 days to clear unless certified!
(All payments in US dollars ONLY!)
```

4.2 Updates

Whenever a new release of MUI gets released, I will post some information in the appropriate newsgroups of some electronic networks. The new archive will soon be available on many bulletin boards and on all '`aminet`' FTP servers. Major releases will also come with some PD disks, especially on Fred Fish's collection.

The MUI Support Mailbox '`Amiga Unlimited`' (Node 1: +49-8151-78880 [HST/V.32bis], Node 2: +49-8151-290282 [ISDN X.75/V110], Sysop Andreas Schildbach) always offers the most recent release for Fido file request at node 2:246/46. Use the magic name '`MUI`'. Alternatively you can log in with a user name of '`Download MUI`' and a password of '`MUI`'. In the United States, you can also get MUI from 1:267/131 (V.32bis).

Note: File Requesting '`MUI`' as of V1.4 will send you both the '`muiXXusr.lha`' & the '`muiXXdev.lha`' archives. If you want them seperately, please request either '`MUIDEV`' or '`MUIUSR`' for the latest versions!

As mentioned above, registered users will neither need a new keyfile nor a special personalized program version. They can use all new features immediately.

Of course, every MUI update will be completely compatible to all previously released versions. All applications will continue to run and automatically benefit from possible enhancements in user interface design.

4.3 Support

If you have some questions, comments, suggestions or even flames, please feel free to contact me at one of the following addresses. If you send your letter via e-mail, there's a good chance for getting a quick reply.

```
Snailmail: Stefan Stuntz
~          Eduard-Spranger-Straße 7
~          80935 München
```

~ GERMANY

Phone: +49-89-313-1248

~ e-mail: stuntz@informatik.tu-muenchen.de

4.4 Acknowledgments

The author wishes to thank

– Stefan Becker

... he seemed to have very few time but nevertheless gave some valuable hints. Parts of his '`ToolManager`' source code were a great help during MUI's development.

– Martin Berndt ... solved some tricky problems.

– Robert Blayzor ... reworked the english manual.

– Dirk Federlein

... for his application '`DFView`'. With over 100 kByte of source, locale support and documentation in three different formats I don't dare speaking of an "example-program" any more. Additionally, he translated parts of this user documentation.

– Georg "gucky" Heßmann

... for reporting some bugs and for his demo program '`DVIprint`'.

– Martin Horneffer and Albert Weinert

... for creating the Oberon language interface.

– Martin "XEN" Huttenloher

... has drawn many of the supplied images and also significantly cooperated in other parts of the MUI-Design. Furthermore he contributed the amazing background patterns, which are a small extract of his '`MagicWB`' package. Friends of an impressive and plastic Workbench should definitely take a closer look at his package '`MagicWB`'!

– Oliver "Mr.Coffee" Kilian

... for testing MUI on good old (and slow) 68000.

– Klaus "kmel" Melchior

... for the two sample tools '`WbMan`' and '`MUI-Exchange`' and for endless lists of bug reports. He also painted the demo programs icons and supplied some BOOPSI images.

– Wouter van Oortmerssen

... for the Amige-E interface.

- Armin Sander

  ... for giving me my first ideas about object oriented GUI design. He told me a lot about classes and objects and made me start with MUI.

- Matthias "tron" Scheler und Markus "corwin" Stipp

  ... for writing the first real MUI application, a message editor for the '`Universal Mail System (UMS)`'. Look out for '`IntuiNews`'! Additionally, Matthias wrote the sample program '`Font`'.

- Andreas "goonie" Schildbach

  ... significantly influenced the design and functionality of MUI and is currently working on a MUI application, a phone and answer machine for ISDN. He made me think of some other things during our endless phone calls.

- Wolfgang Schildbach

  ... for his text formatting code.

- Christian Scholz

  ... for the Modula interface.

- Ibrahim "radi" Solmaz

  ... who also prevented me from working with many phone calls but nevertheless was a valuable help sometimes.

- Henri Veistera

  ... for the assembler interface.

The last big thanks is reserved for all registered users of my file requester MFR. Its success made me trying it with shareware again. I'm sorry, that there was no update for MFR for such a long time, but I put all my efforts on this new product. Hope I can release a new, MUI based MFR soon.

4.5 Discussion

- "How do I get a good-looking 3D-Slider with the typical XEN-Look?"

  You'll have to set several options to the following settings:

  - FRAMES: Thickness=thin
  - FRAMES: Edit Prop-Frame: Type=Double, Look=Raised and all spacings set to ZERO!
  - IMAGES: Edit Prop-Gadget Knob: Select BOOPSI-Image '`mui-xenknob`'.
  - IMAGES: Backgrounds: Edit Prop-Gadget Container: Select the pattern '`Fillback`' (4th from right side).

  - Finally you may select the corresponding XEN-Arrows in the IMAGES-Window. These arrows are MUI Brushes. Load them via the function '`Select: MUI Brush`' in the Edit-window. They were especially designed to fit the XEN-Slider.
- "Why don't MUI's string gadgets support the clipboard?"

  There is a utility called '`NewEdit`' that adds clipboard support to all system string gadgets. Of course MUI string gadgets work with this utility too. You can find this thing on aminet or on some PD disks.

4.6 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.

4.7 License

- This license applies to the product called "MagicUserInterface" (short "MUI"), a collection of programs for the Amiga computer, published by Stefan Stuntz under the concepts of shareware, and the accompanying documentation. The terms "Program" and "MUI" below, refer to this product. The licensee is addressed as "you".

- You may copy and distribute verbatim copies of the program's executable code and documentation as you receive it, in any medium, provided that you conspicuously and appropriately publish only the original, unmodified program, with all copyright notices and disclaimers of warranty intact and including all the accompanying documentation, example files and anything else that came with the original.
- Except when otherwise stated in this documentation, you may not copy and/or distribute this program without the accompanying documentation and other additional files that came with the original. You may not copy and/or distribute modified versions of this program.
- You may not copy, modify, sublicense, distribute or transfer the program except as expressly provided under this license. Any attempt otherwise to copy, modify, sublicense, distribute or transfer the program is void, and will automatically terminate your rights to use the program under this license. However, parties who have received copies, or rights to use copies, from you under this license will not have their licenses terminated so long as such parties remain in full compliance.
- By copying, distributing and/or using the program you indicate your acceptance of this license to do so, and all its terms and conditions.
- Each time you redistribute the program, the recipient automatically receives a license from the original licensor to copy, distribute and/or use the program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein.
- You may not disassemble, decompile, re-source or otherwise reverse engineer the program.
- You agree to cease distributing the program and data involved if requested to do so by the author.

4.8 Installer

Along with MUI comes the '`Installer`' from Commodore:

```
Installer and Installer project icon
(c) Copyright 1991-93 Commodore-Amiga, Inc.  All Rights Reserved.
Reproduced and distributed under license from Commodore.

INSTALLER SOFTWARE IS PROVIDED "AS-IS" AND SUBJECT TO CHANGE;
NO WARRANTIES ARE MADE. ALL USE IS AT YOUR OWN RISK. NO LIABILITY
OR RESPONSIBILITY IS ASSUMED.
```