

MUIUndoc

Alessandro Zummo

COLLABORATORS

	<i>TITLE :</i> MUIUndoc		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Alessandro Zummo	April 12, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	MUIUndoc	1
1.1	The Hacker's Guide to MUI	1
1.2	The Hacker's Guide to MUI - Index	1
1.3	The Hacker's Guide to MUI - Intro	3
1.4	The Hacker's Guide to MUI - Introduction	3
1.5	The Hacker's Guide to MUI - Installation	4
1.6	The Hacker's Guide to MUI - Crawling.mcc	4
1.7	The Hacker's Guide to MUI - Crawling.mcc - Description	4
1.8	The Hacker's Guide to MUI - Crawling.mcc - Warning	5
1.9	The Hacker's Guide to MUI - Crawling.mcc - A question...	5
1.10	The Hacker's Guide to MUI - Crawling.mcc - Example	5
1.11	The Hacker's Guide to MUI - Suggestions	6
1.12	The Hacker's Guide to MUI - Suggestions - MUIM_List_CreateImage	6
1.13	The Hacker's Guide to MUI - Suggestions - MUI_AddClipping()	7
1.14	The Hacker's Guide to MUI - Suggestions - MUI_Begin/EndRefresh	8
1.15	The Hacker's Guide to MUI - Suggestions - MUIA_Gauge_Divide	9
1.16	The Hacker's Guide to MUI - Suggestions - Stefan's anger	9
1.17	The Hacker's Guide to MUI - Suggestions - Stack	11
1.18	The Hacker's Guide to MUI - Undocumented items	11
1.19	The Hacker's Guide to MUI - Undocumented items - MUIA_CustomBackfill	12
1.20	The Hacker's Guide to MUI - Undocumented items - MUIM_Window_Setup/Cleanup	12
1.21	The Hacker's Guide to MUI - Undocumented items - MUIA_Group_Forward	13
1.22	The Hacker's Guide to MUI - Undocumented items - MADF_DRAWALL	13
1.23	The Hacker's Guide to MUI - Undocumented items - MUIA_Window_DisableKeys	14
1.24	The Hacker's Guide to MUI - Undocumented items - MUIM_Window_Snapshot	15
1.25	The Hacker's Guide to MUI - Undocumented items - MUIM_Window_ActionIconify	15
1.26	The Hacker's Guide to MUI - Undocumented items - MUIA_Text_HiCharIdx	16
1.27	The Hacker's Guide to MUI - Undocumented items - MUIM_GoActive/GoInactive	16
1.28	The Hacker's Guide to MUI - Undocumented items - MUIA_Prop_DoSmooth	17
1.29	The Hacker's Guide to MUI - Undocumented items - MUIM_Create/DeleteShortHelp	17

1.30	The Hacker's Guide to MUI - Others	18
1.31	The Hacker's Guide to MUI - Others - Drag & Drop	19
1.32	The Hacker's Guide to MUI - Others - Drag & Drop - Gilles says ALL!	20
1.33	The Hacker's Guide to MUI - Others - Optimized versions?	21
1.34	The Hacker's Guide to MUI - Others - Rastport troubles	22
1.35	The Hacker's Guide to MUI - Others - OM_ADD/REMMEMBER	22
1.36	The Hacker's Guide to MUI - Others - Clipping	23
1.37	The Hacker's Guide to MUI - Others - AboutMUI	23
1.38	The Hacker's Guide to MUI - Others - MUIC_ImageDisplay & MUIC_Popimage	23
1.39	The Hacker's Guide to MUI - Others - MUIC_Framedisplay && MUIC_Popframe	25
1.40	The Hacker's Guide to MUI - Others - mad_Flags & mri_Flags	26
1.41	The Hacker's Guide to MUI - Others - _parent(obj)	26
1.42	The Hacker's Guide to MUI - Internals	27
1.43	The Hacker's Guide to MUI - Internals - MUIA_Background	27
1.44	The Hacker's Guide to MUI - Internals - AppMessage	27
1.45	The Hacker's Guide to MUI - Internals - MUI_DisposeObject()	28
1.46	The Hacker's Guide to MUI - Examples	28
1.47	The Hacker's Guide to MUI - Examples - Datatypes	28

Chapter 1

MUIUndoc

1.1 The Hacker's Guide to MUI

Welcome to...

The Hacker's Guide to MUI :-)

Release 1.4

Alessandro Zummo, azummo@ita.flashnet.it

Converted to AmigaGuide by Stephan Schreiber, septh@isicom.fr

Current MUI version: 19

Hope next versions will be more documented

Get to the

[index](#)

, or read MUIUndoc.h.

1.2 The Hacker's Guide to MUI - Index

Index - The Hacker's Guide to MUI 1.4

New or changed: 3.11

0 The Hacker's Guide to MUI

0.1 Introduction

0.2 Installation

1 What's Crawling.mcc all about?

1.1 Description

-
- 1.2 But the author says...
 - 1.3 Another question
 - 1.4 Little example
 - 2 Suggestions by Stefan
 - 2.1 About MUIM_List_CreateImage
 - 2.2 MUI_AddClipping()
 - 2.3 MUI_Begin/End_Refresh
 - 2.4 MUIA_Gauge_Divide
 - 2.5 What about _screen(_app(obj)) ?
 - 2.6 Stack
 - 3 Undocumented methods, tags and goodies
 - 3.1 MUIA_CustomBackfill
 - 3.2 MUIM_Window_Setup/Cleanup
 - 3.3 MUIA_Group_Forward
 - 3.4 MADF_DRAWALL
 - 3.5 MUIA_Window_DisableKeys
 - 3.6 MUIM_Window_Snapshot
 - 3.7 MUIM_Window_ActionIconify
 - 3.8 MUIA_Text_HiCharIdx
 - 3.9 MUIM_GoActive/Inactive
 - 3.10 MUIA_Prop_DoSmooth
 - 3.11 MUIM_Create/DeleteShortHelp
 - 4 Some other fine things
 - 4.1 Customizing Drag & Drop operations
 - 4.1.1 All Good Things
 - 4.2 680x0 optimized versions
 - 4.3 RastPort troubles
 - 4.4 OM_ADD/REMEMBER
 - 4.5 Clipping
-

- 4.6 AboutMUI calling techniques
- 4.7 MUIC_Imagedisplay & MUIC_Popimage
- 4.8 MUIC_Framedisplay & MUIC_Popframe
- 4.9 mad_Flags & mri_Flags
- 4.10 _parent(obj)
- 5 MUI internals
 - 5.1 MUIA_Background
 - 5.2 AppMessage
 - 5.3 MUI_DisposeObject

6 Examples

6.1 Datatypes

That's all folks! I hope this doc will get smaller. No, i'm not ←
crazy,

the dimension of this doc is directly proportional to the amount
of undocumented features of MUI. :-)

1.3 The Hacker's Guide to MUI - Intro

0 The Hacker's Guide to MUI

0.1 Introduction

0.2 Installation

1.4 The Hacker's Guide to MUI - Introduction

0.1 Introduction

This doc tries to explain you all of the "undocumented" features of MUI. Most of the texts are directly snapped from the MUI Mailing List, so the messages i reported here are copyright by the respective authors.

Please, don't ask Stefan about unofficial fetures listed here, or he may kill you :-) MUI is complex enough...so i've marked official undocumented features with [SAFE] in the chapter four.

This doc is unofficial, read at your own risk!

Copyright (c) 1997 by Alessandro Zummo.
Feedbacks and/or flames to azummo@ita.flashnet.it

MUI is Copyright (c) 1997 by Stefan Stuntz

Thanks to: Jerome Fleury, Gilles Masson, Maik Schreiber
and all the others

1.5 The Hacker's Guide to MUI - Installation

0.2 Installation

1. Copy muiundoc.h in include:mui/
2. Add the line `#include "mui/muiundoc.h"` in your libraries/mui.h
3. Read muiundoc.h to discover some options

PLEASE, follow these rules and do not change the include's name.
It's the only way to allow easy source exchanging.

1.6 The Hacker's Guide to MUI - Crawling.mcc

1 What's Crawling.mcc all about?

1.1 Description

1.2 But the author says...

1.3 Another question

1.4 Little example

1.7 The Hacker's Guide to MUI - Crawling.mcc - Description

1.1 Description

Crawling.mcc is a standard undocumented MUI mcc which simply create a timed-scrolling virtgroup.

Here is an example:

```
#define MUIC_Crawling "Crawling.mcc"  
#define CrawlingObject MUI_NewObject(MUIC_Crawling
```

To use Crawling.mcc, simply create it as any other VirtGroup object:

```

Child, CrawlingObject,
  TextFrame,
  MUIA_Background, MUII_TextBack,
  MUIA_FixHeightTxt, "\n",
  Child, TextObject,
    MUIA_Text_Contents, "\033cHello guys...\nThis is some text.\n"
    "I hope you like it.\nBye!\n\n",
  End,
End,

```

1.8 The Hacker's Guide to MUI - Crawling.mcc - Warning

1.2 But the author says...

It's not public and subject to change. Be warned!

Bye, Klaus

1.9 The Hacker's Guide to MUI - Crawling.mcc - A question...

1.3 Oh..here's another question

> Fine, but how do I make the overlap smoother?

Copy the text from top to bottom.

Bye, Klaus

1.10 The Hacker's Guide to MUI - Crawling.mcc - Example

1.4 Little example

[....]

```

Child, CrawlingObject,
  TextFrame,
  MUIA_Background, MUII_TextBack,
  MUIA_FixHeightTxt, "\n",

  Child, VGroup,

    Child, TextObject,
      MUIA_Text_Contents, __DCOPYRIGHT
      MUIA_Text_PreParse, MUIX_C,
    End,

    [... more Textobjects here ...]

  Child, TextObject,
    MUIA_Text_Contents, __DCOPYRIGHT,

```

```

        MUIA_Text_PreParse, MUIX_C,
    End,
    End,
End,

[....]

```

The TextContents of the last TextObject (the repeated one, in this case __DCOPYRIGHT) have to have exactly the number of lines your Crawling object can display.

1.11 The Hacker's Guide to MUI - Suggestions

2 Some suggestions by Stefan

2.1 About MUIM_List_CreateImage

2.2 MUI_AddClipping()

2.3 MUI_Begin/End_Refresh

2.4 MUIA_Gauge_Divide

2.5 What about _screen(_app(obj)) ?

2.6 Stack

1.12 The Hacker's Guide to MUI - Suggestions - MUIM_List_CreateImage

2.1 About MUIM_List_CreateImage

> So is it legal to keep images in the instance data *after* the cleanup
> Method, and reopen the window with images in it ?

Absolutely *not*.

> In order to put images into a list, you have to create a bitmap
> object, and turn that into a "black box". Now, the problem is that I
> want "a few" images (there could be a couple of hundred images (some
> people like me like to pile up objects -:), but usually there will be
> a lot less --- maybe I should just skip the images if there are more
> than 52) in a list, each of which is taken from a larger bitmap (which
> is already remapped, and is a friend bitmap of the screen).

>

> So, it seems that I have to do this for every image:

> - create a small bitmap to hold the image
> - copy the image from the large bitmap to the small bitmap
> - create a bitmap object around it
> - create the blackbox (MUIM_List_CreateImage) object
> - put that into the list

>
> Now, this seems like a lot of overhead, considering that I only want a
> couple of 16*16 images (ready to blast into the window) in a list
>
> So, the question is: can I dispose the small bitmap and/or the bitmap
> object after creating the blackbox object? Which is almost equivalent
> to asking: does the blackbox object contain the image?

You must not dispose the bitmap object after `MUIM_List_CreateImage`.
`MUIM_List_CreateImage` does not copy anything, it just sets up the bitmap
object for display. Basically, `MUIM_List_CreateImage` does the same
"setup work" on an object that it otherwise receives when being attached
to a window.

> Why don't we get a hook to render images?

Why would you want a hook? Hooks suck! :-) Too bad I didn't notice that
earlier, otherwise MUI wouldn't have any strange callback hooks but
instead methods for overloading in subclasses all over the place.

Anyway, the solution for your problem is probably easy. Hrm... MUI
should be able to accept any subclass of area class as input for
`MUIM_List_CreateImage`, not just bitmap classes. I know, the
documentation speaks of `Bitmap` classes only, but I just checked the
sources and see no real reason why it shouldn't work with other things
too.

So, you could create a subclass of area class that has full access to
your big, already remapped picture. Make it return reasonable (small,
fixed) min/max/default sizes in `MUIM_AskMinMax` and make it draw a small
part of the big `Bitmap` in `MUIM_Draw`. MUI won't do any sophisticated
layout stuff on `\330[]` objects, it will simply render them in their
default width & height.

That should work out well, and you only need one single
`MUIM_List_CreateImage` on this "fake" object and not a few thousands for
every line of your listview.

The only problem now is to tell your object what part of the bitmap to
draw. To solve that, invent a little attribute which identifies this
part and set it on each invocation of your display hook.

Sure, this sounds quite hacky. A cleaner solution would of course be a
listclass that has real objects as children and not just pointers. But
then, this is more like a virtual group than a list and is unfortunately
slower.

> Also, just for completeness: why doesn't the text engine support
> `INVERSVID`? Is there a good reason for that, or just the usual "99% of
> all applications don't need it"? -:)

The latter. Doesnt it look extremely ugly?

1.13 The Hacker's Guide to MUI - Suggestions - `MUI_AddClipping()`

2.2 MUI_AddClipping()

Allan Odgaard wrote in article <33424095.7134@DIKU.DK>:

```
> Is it possible to MUI_AddClipping() to a cloned rastport, so that I do
> not need to remove it again?
```

No. MUI clipping calls should only be used on `_rp(obj)`.

--

Greetings, Stefan

1.14 The Hacker's Guide to MUI - Suggestions - MUI_Begin/EndRefresh

2.3 MUI_Begin/End_Refresh

Daniel wrote in article <9704011053.AA11676@sparcserver.lrz-muenchen.de>:

```
> As I haven't found info about these two methods in the autodocs,
> could someone explain me what are these methods for.
>
> Are they parallel to those of intuition?
> What are their effect?
```

They are used for restoring window contents after custom scrolling operations that might have screwed up some damage regions. You probably will never need them unless you write your own list classes.

--

Greetings, Stefan

```
#define LayerCovered(l) ((!(l)->ClipRect) || (l)->ClipRect->bounds!=(l)->bounds)
#define LayerDamaged(l) ((l)->DamageList && (l)->DamageList->RegionRectangle)
#define NeedZeroScrollRaster(l) (LayerCovered(l) || LayerDamaged(l))
```

// V39 only apps should probably use ScrollWindowRaster() instead.

```
static VOID MyScrollRaster(struct IClass *cl, Object *obj, LONG dy, LONG left, LONG ←
    top, LONG width, LONG height)
{ struct RastPort *rp = _rp(obj);
  struct Window *w = _window(obj);

  if      (dy>0 && dy<height) ClipBlit(rp, left, top+dy, rp, left, top    , width, ←
    height-dy, 0xc0);
  else if (dy<0 && -dy<height) ClipBlit(rp, left, top    , rp, left, top-dy, width, ←
    height+dy, 0xc0);

  if (dy && (w->Flags & WFLG_SIMPLE_REFRESH))
  {
    struct Layer *l = w->WLayer;

    if (NeedZeroScrollRaster(l))
    {
```

```

    UBYTE oldmask = rp->Mask;
    SetWrMsk(rp,0);
    ScrollRaster(rp,0,dy,left,top,left+width-1,top+height-1);
    SetWrMsk(rp,oldmask);
}

if (MUI_BeginRefresh(muiRenderInfo(obj),0))
{
    /* find and redraw root object of window */

    Object *o = obj;

    while (_parent(o))
        o = _parent(o);

    MUI_Redraw(o,MADF_DRAWALL);

    MUI_EndRefresh(muiRenderInfo(obj),0);
}
}
}

```

1.15 The Hacker's Guide to MUI - Suggestions - MUIA_Gauge_Divide

2.4 MUIA_Gauge_Divide

Marcin Orlowski wrote in article <yam6984.1299.136418104@195.116.59.5>:

```

> Gauge.doc says:
>
>     NAME
>     MUIA_Gauge_Divide -- (V4 ) [ISG], BOOL
>
>     FUNCTION
>     If this attribute is != 0, every value set with
> [...]
>
> If so, it can't be BOOL.

```

This is a doc bug. The correct type of this attribute is ULONG.

--

Greetings, Stefan

1.16 The Hacker's Guide to MUI - Suggestions - Stefan's anger

2.5 What about _screen(_app(obj)) ?

Dr. Karl Bellve wrote

```

> This is my point. I think the _screen(obj) is NULL if the object has not
> been drawn yet. Therefore, I am using _screen(_app(obj)) instead, which
> works if the window of the object has not been drawn yet. This is

```

> important for mapping datatypes to the screen but don't have to be drawn
> at this time but later.

RUBBISH! NONSENSE! CRAP! EVERY WORD IN HERE IS TOTALLY WRONG! :-))

A little more detailed:

> I think the `_screen(obj)` is NULL if the object has not been drawn yet.

WRONG!

> Therefor, I am using `_screen(_app(obj))` instead,

ACCESS TO RANDOM MEMORY!

> which works if the window of the object has not been drawn yet.

WRONG!

> This is important for mapping datatypes to the screen but don't
> have to be drawn at this time but later.

NO WAY!

Shit... nobody believes me. It doesnt matter what you "think" in this case. Important is what the MUI docs say and what I write! :)

Well, lets try again from scratch...

mui.h:

```
#define _app(obj)    [...] /* valid between MUIM_Setup/Cleanup */
#define _screen(obj) [...] /* valid between MUIM_Setup/Cleanup */
```

So... it is FUCKING ILLEGAL to use either `_app(obj)` or `_screen(obj)` before your object received a `MUIM_Setup` and after your object received a `MUIM_Cleanup`. It doesnt make any sense to use one instead of the other just because you might "think" one is valid and one is not.

This was the first point that makes your statements totally absurd. The second point follows now:

- (a) `_app(obj)`, when used correctly between setup and cleanup, returns a pointer to an object of Application class.
- (b) `_screen(obj)` is defined as `muiRenderInfo(obj)->mri_Screen`.
- (c) `muiRenderInfo(obj)` is a macro that accesses fields of the instance data of Area class.
- (d) Application class is NO FUCKING CHILD of Area class. Only objects that are to be displayed are children of Area class, like Groups, Gadgets, Text, etc. Only these objects have a `muiRenderInfo(obj)`.

(a) + (b) + (c) + (d):

`_screen(_app(obj))` is a COMPLETELY FUCKING NONSENSE which returns anything but a valid screen pointer.

So... finally... please stop confusing other programmers with these tails! MUI is complex enough, there's no need to cause more trouble with wrong advices!

Remapping of a picture has to be done in the Setup method of the custom class that displays this picture. If you dont want to do this, you loose.

--
Greetings, Stefan

1.17 The Hacker's Guide to MUI - Suggestions - Stack

2.6 Stack

Hi!

If I see one more MUI program out there with less than 8k stack size, I promise to kick the programmer's ass very hard as soon as I get the chance! :-)

Is it really so difficult to put a LONG __stack = 8192 in your source?

Is it really too much work to edit your project icons?

Does it really cause too much trouble to insert NP_StackSize, 8192 into your CreateNewProc() calls?

I don't know how often you wanna hear it... well... here we go again:

```
*****
THE DEFAULT STACK SIZE OF 4K IS NOT ENOUGH FOR ANY MUI PROGRAM
AND WILL CAUSE MAJOR TROUBLE AND FANCY CRASHES SOONER OR LATER
*****
```

--
Greetings, Stefan

1.18 The Hacker's Guide to MUI - Undocumented items

3 Undocumented methods & tags

3.1 MUIA_CustomBackfill

3.2 MUIM_Window_Setup/Cleanup

3.3 MUIA_Group_Forward

3.4 MADF_DRAWALL

- 3.5 MUIA_Window_DisableKeys
- 3.6 MUIM_Window_Snapshot
- 3.7 MUIM_Window_ActionIconify
- 3.8 MUIA_Text_HiCharIdx
- 3.9 MUIM_GoActive/Inactive
- 3.10 MUIA_Prop_DoSmooth
- 3.11 MUIM_Create/DeleteShortHelp

1.19 The Hacker's Guide to MUI - Undocumented items - MUIA_CustomBackfill

3.1 I was asking about and undocumented tag: :-)

```
> >MUIA_CustomBackfill
>
> What's this?
```

Nothing to worry about :)

--

Greetings, Stefan

1.20 The Hacker's Guide to MUI - Undocumented items - MUIM_Window_Setup/Cleanup

3.2~MUIM_Window_Setup/Cleanup [SAFE]

```
> >Why don't windows subclasses receive MUIM_Setup/MUIM_Cleanup on
> >opening/closing ?
>
> >I had to notify MUIA_Window_Open to react on this.
>
> Setup/Cleanup are methods of Area class. Window is not a
> subclass of area class. What you do is subclass window
> class and put your code to setup/cleanup in the Set
> method where the attribute for MUIA_Window_Open is set. Do
> your setup when TRUE, and your cleanup when FALSE.
```

Although this might work, OM_SET is usually not a good place to do init/cleanup stuff for windows. Better override the yet private but soon public methods

```
#define MUIM_Window_Cleanup 0x8042ab26 /* private */ /* V18 */
#define MUIM_Window_Setup   0x8042c34c /* private */ /* V18 */

struct MUIP_Window_Cleanup { ULONG MethodID; }; /* private */
struct MUIP_Window_Setup   { ULONG MethodID; }; /* private */
```


MUIM_Window_Setup is called right before MUIM_Setup of the root object. MUIM_Window_Cleanup is called right after MUIM_Cleanup of the root object.

Usage of those methods is similar to MUIM_Setup and MUIM_Cleanup, ie you should do something like this:

```
MUIM_Window_Setup:
{
    if (!DoSuperMethodA(cl,obj,msg))
        return(FALSE);

    if (setup_my_stuff())
        return(TRUE);

    CoerceMethod(cl->cl_Super,obj,MUIM_Window_Cleanup);
    return(FALSE);
}
```

Note that the window is still closed when MUIM_Window_Setup is called. If you return FALSE, it will refuse to open. The parent screen, however, is already available by getting MUIA_Window_Screen. This allows reading some datatypes or allocating some colors or whatever else.

Anyway, MUIM_Window_Setup is not useful very often. In general, it's much better and cleaner if you encapsulate things in your custom subclasses of area class.

--
Greetings, Stefan

1.21 The Hacker's Guide to MUI - Undocumented items - MUIA_Group_Forward

3.3 MUIA_Group_Forward [SAFE]

Steve Koren wrote in article <yam7040.2013.142054680@mail.frii.com>:

> Stefan, if you're listening, it'd be really nice to have a way to turn
> this feature off for group class.

```
#define MUIA_Group_Forward 0x80421422 /* V11 .s. BOOL */ /* private */
```

When this special attribute is found FALSE in the tag list of an OM_SET passed to group class, it won't forward the whole OM_SET to any of its children. It will, however, pass the OM_SET to area & notify class.

This attribute will go public.

1.22 The Hacker's Guide to MUI - Undocumented items - MADF_DRAWALL

3.4 MADF_DRAWALL

Gilles MASSON wrote in article <199704090914.LAA02400@ogpsrv.unice.fr>:

```
> > while (_parent(o))
> >   o = _parent(o);
>   ^^^^^^^
>   What is this ???
```

Thats the same as getting MUJA_Parent, only that I can do it faster than you! :-) Basically, this loop finds the root object of the window starting with the current object.

```
> > MUI_Redraw(o,MADF_DRAWALL);
>   ^^^^^^^^^^^^^^^
>   And what is this ???
```

Uhm. Replace width MADF_DRAWOBJECT for now.

```
> Actually i just do a MUI_Redraw(obj,MADF_DRAWOBJECT) and it seems to
> work. What is the difference between MADF_DRAWALL and MADF_DRAWOBJECT ?
> Is it necessary to refresh the whole window when the scroll where within
> the obj bounds ?
```

Yes. There can be more damage to your window by other layer operations and you would miss to refresh these regions if you wouldnt redraw the whole thing. MUI will take care of scanning damage regions and only forward draw methods to objects that actually were damaged.

```
> Is there a way in the Draw method to know that it's
> a refresh asked by MUI ? because i make many things in it and for a
> refresh it would be better for my class to skip some parts....
```

You can simply set/delete a flags in your instance data before/after calling MUI_Redraw().

--

Greetings, Stefan

1.23 The Hacker's Guide to MUI - Undocumented items - MUIA_Window_DisableKeys

3.5 MUIA_Window_DisableKeys [SAFE]

Allan Odgaard wrote in article <333F3658.5715@DIKU.DK>:

```
> Hi, how is it possible, within a custom gadget, to react on the tab key,
> if the key is configured go to next gadget???
```

You must disable MUIs TAB handling for this window in the setup method of your custom class:

```
#define MUIA_Window_DisableKeys 0x80424c36 /* V15 isg ULONG */ /* private */

set(_win(obj),MUIA_Window_DisableKeys,
```

```
(1<<MUIKEY_GADGET_NEXT) | (1<<MUIKEY_GADGET_PREV));
```

>

> Will this definitely go public or keep private ?

Well, at least it will remain there so you can use it. But please, use it sparingly! It's not a good idea to disable MUI's keyboard handling.

--

Greetings, Stefan

1.24 The Hacker's Guide to MUI - Undocumented items - MUIM_Window_Snapshot

3.6 MUIM_Window_Snapshot [SAFE]

Thomas Igracki wrote in article <23D25760H0006D7FAH@igracki.jana.berlinet.de>:

> Hi!

>

> Is there a method/function I can call when THE PROGRAM wants to
> snapshot the windows position/size?

> Like it does the menu in the top border, but I want to do it from my
> application, CxBar, which optionally can switch off the window
> borders, and with that, the snapshot menuitem isn't there;-(

```
#define MUIM_Window_Snapshot 0x8042945e /* V11 */ /* private */  
struct MUIM_Window_Snapshot { ULONG MethodID; LONG flags; };
```

flags==0 => unsnapshot position

flags==1 => snapshot position

This method will be public in MUI V18.

BTW... just in case you forgot. You can unsnapshot a window by doubleclicking on the snapshot gadget.

--

Greetings, Stefan

1.25 The Hacker's Guide to MUI - Undocumented items - MUIM_Window_ActionIconify

3.7 MUIM_Window_ActionIconify [SAFE]

Christian Stieber wrote

> > That's an interersing topic. How do others handle multiple projects
> > with MUI?

>

> One app object. SInce MUI doesn't let me iconify specific windows,
> it just isn't implemented.

Write a subclass of window class, override the method

```
#define MUIM_Window_ActionIconify 0x80422cc0 /* private */ /* V18 */
and do whatever you wish. Make sure to give feedback!

If you dont override it, it does
set(_app(obj),MUIA_Application_Iconified,TRUE);

--
Greetings, Stefan
```

1.26 The Hacker's Guide to MUI - Undocumented items - MUIA_Text_HiCharIdx

3.8 MUIA_Text_HiCharIdx [SAFE]

```
SG> However if I want to create the label object by hand with
SG> the TextObject class I cannot use
SG>
SG> obj = TextObject,
SG>     MUIA_Text_Contents, "F_oo",
SG>     ...
SG>     End;
```

Just use this private tag:

```
#define MUIA_Text_HiCharIdx 0x804214f5
```

and do something like that:

```
obj=TextObject,
    MUIA_Text_HiCharIdx, '_',
    MUIA_Text_Contents, "F_oo",
    ...,
    End;
```

This tag will simply underscore the character in the string that follows the one gived as his argument.

However, i'm not sure of this. Just try!

1.27 The Hacker's Guide to MUI - Undocumented items - MUIM_GoActive/GoInactive

3.9 MUIM_GoActive/GoInactive [SAFE]

Considering the "active gadget" problem, I'm working on a solution. MUI basically does have methods like MUIM_GoActive and MUIM_GoInactive which indicate whether an object gets activated/deactivated. I'm not absolutey confident with their behaviour yet, so they're still private and subject to change. If you want to experiment with that, here we go:

```
#define MUIM_GoActive    0x8042491a
#define MUIM_GoInactive 0x80422c0c
```

Note that you may not send these methods yourself, they are sent to you by MUI when your object becomes the active one. Also note that these methods are still subject to change, so please don't release code based on this yet without contacting me first!

BTW, if you pass these methods to your superclass, MUI will draw the active object frame. If you dont, nothing will be drawn and you're responsible for displaying this state yourself, e.g. with a cursor. Remember not to draw anything outside of MUIM_Draw, use MUI_Redraw() instead!

--

Greetings, Stefan

some notes:

Seems you don't get these events when selected by the tab-key; you have to physically set yourself as MUIA_Window_ActiveObj, e.g. when you get a mouse-click within your bounds, then you get an MUIM_GoActive immediately and subsequently a MUIM_GoInactive when you get deselected... (although sometimes you don't get this event when activating a StringObject with a click).

There's also something very weird; although I'm using an MUI_EventHandlerNode, I still get MUIM_HandleInput method calls occasionally, which I seem to need to handle in order to be sure of picking up all MUIKEY events...

Still... Stefan never said it worked... !

Ellis.

1.28 The Hacker's Guide to MUI - Undocumented items - MUIA_Prop_DoSmooth

3.10 MUIA_Prop_DoSmooth [SAFE]

This is not an option of the list, the prop object of the scrollbar does it automatically. This implies however that the linkage between your scrollbar and listobject is based on pixels, not lines.

If this is the case, simply set the

```
#define MUIA_Prop_DoSmooth 0x804236ce
```

to TRUE to enable smoothing.

1.29 The Hacker's Guide to MUI - Undocumented items - MUIM_Create/DeleteShortHelp

```
** UNOFFICIAL, INVENTED BY ME (BUT TESTED :- ) **
```

MUIM_CreateShortHelp

```
DoMethod(obj,MUIM_CreateShortHelp, LONG mx, LONG my);
```

When MUI is about to show a new short help bubble, it does not simply use the `MUIA_ShortHelp` field of area class. Instead, it sends a `MUIM_CreateShortHelp` to the object in question and uses the return value as the string to be displayed.

When `MUIM_CreateShortHelp` reaches area class, it just return the contents of `MUIA_ShortHelp` so you neednt care about this method if you only have static, non-changing short help bubbles.

However, if your short help bubble depend on some internal states of your objects or on the mouse position within your objects, you have to have a subclass which overrides `MUIM_CreateShortHelp`, creates an appropriate string and returns it.

INPUTS

`mx` - current x position of mouse
`my` - current y position of mouse

Since MUI does (unfortunately) not use relative coordinates at all, these two aren't relative either.

RESULT

You must return a pointer to a string or `NULL` if you failed to create one.

NOTES

Even when overriding `MUIM_CreateShortHelp`, you *must* set `MUIA_ShortHelp` of your object to something different from `NULL`. MUI will find out that your object actually has a short help by directly checking the contents of `MUIA_ShortHelp` in the instance data of area class due to speed reasons.

MUIM_DeleteShortHelp

```
DoMethod(obj,MUIM_DeleteShortHelp, STRPTR help);
```

You will receive this method after a the deletion of a short help bubble. You can then perform some memory freeing operations.

INPUTS

`help` - pointer to the string which was displayed in the bubble

1.30 The Hacker's Guide to MUI - Others

- 4 Some other fine things
 - 4.1 Customizing Drag & Drop operations
 - 4.1.1 All Good Things
 - 4.2 680x0 optimized versions
 - 4.3 RastPort troubles
 - 4.4 OM_ADD/REMEMBER
 - 4.5 Clipping
 - 4.6 AboutMUI calling techniques
 - 4.7 MUIC_Imagedisplay & MUIC_Popimage
 - 4.8 MUIC_Framedisplay & MUIC_Popframe
 - 4.9 mad_Flags & mri_Flags
 - 4.10 _parent(obj)

1.31 The Hacker's Guide to MUI - Others - Drag & Drop

4.1 Customizing Drag & Drop operations [SAFE]

Hi!

Here's the experimental API on customizing drag & drop operations, which means starting d&d under program control and creating custom drag images. Feel free to test it but beware... things might change. Make sure to report your experiences to this list!

To initiate D&D on an object, send it the method

```
#define MUIM_DoDrag 0x804216bb /* private */ /* V18 */
struct MUIP_DoDrag { ULONG MethodID; LONG touchx; LONG touchy; ULONG flags; }; ←
/* private */
touchx/y are the "handle" position relative to the objects left/top.
flags has to be 0.
```

Whenever d&d is initiated (either like above or under user control), MUI will send the object some methods to create the dragging image. Override them to customize the image.

```
#define MUIM_CreateDragImage 0x8042eb6f /* private */ /* V18 */
struct MUIP_CreateDragImage { ULONG MethodID; LONG touchx; LONG touchy; ULONG ←
  flags; }; /* private */
#define MUIM_DeleteDragImage 0x80423037 /* private */ /* V18 */
struct MUIP_DeleteDragImage { ULONG MethodID; struct MUI_DragImage *di; }; /* ←
  private */
```

MUIM_CreateImage has to return an allocated struct MUI_DragImage:

```
struct MUI_DragImage
{
    struct BitMap *bm;
    WORD width; /* exact width and height of bitmap */
    WORD height;
    WORD touchx; /* position of pointer click relative to bitmap */
    WORD touchy;
    ULONG flags; /* must be set to 0 */
};
```

MUIM_DeleteDragImage has to free the structure and bitmap allocated by MUIM_CreateDragImage. If you do not override these methods, area class will create a drag image that contains a copy of the complete object.

Never send Create/DeleteDragImage methods yourself!

--
Greetings, Stefan

1.32 The Hacker's Guide to MUI - Others - Drag & Drop - Gilles says ALL!

4.1.1 All Good Things [SAFE]

Thanks to Gilless Masson for the text below:

well... go on :)

MUIM_DragQuery, you should know what it does :)
MUIM_DragDrop too.

MUIM_DragBegin is called after a MUIV_DragQuery_Accept reply of DragQuery. If you call the supermethod, the dropmark will be drawn around the full object, else it will not be drawn.

MUIM_DragReport is called after MUIM_DragBegin, and for each mouse moves. What can be done is :

- redraw all or part of the object, but only when msg->update is TRUE.
- return one of the special values :
 - o MUIV_DragReport_Abort :
 - stop to use this object as droppable one.
 - o MUIV_DragReport_Continue :
 - continue default mui stuff (stay drop object if in object bounds)
 - o MUIV_DragReport_Lock :
 - keep this object as current droppable one, even if mouse is not on it !
 - o MUIV_DragReport_Refresh :
 - need to redraw something in the object. MUIM_DragReport method

will be re-called by MUI with `msg->update` set to TRUE.

Using `MUIV_DragReport_Refresh` to ask for a redraw is needed because any redraw will be clipped by the dragged layer if it is visible. When you return `MUIV_DragReport_Refresh`, MUI will close the dragged layer to permit a correct redraw, will call `MUIV_DragReport_Refresh` with `msg->update` set to TRUE to tell you that you can make a `MUI_Redraw(obj,MADF_DRAWUPxxxx)`, and will re-open the dragged layer after.

`MUIM_DragFinish` is called after `MUIM_DragReport` have returned `MUIV_DragReport_Abort` (or `MUIV_DragReport_Continue` and mouse is not on the object). You can call the redraw method in it, and return 0, or call the supermethod which will erase the standard dropmark (which were drawn by `MUIM_DragBegin` supermethod).

That's all folks ;)

Gilles Masson

1.33 The Hacker's Guide to MUI - Others - Optimized versions?

4.2 About 680x0 optimized versions of MUI

Hello MUI users!

I have recently been getting a lot of inquiries about processor optimized versions of the MUI libraries. Sorry, I wasn't able to reply to each of them individually. This article is meant to be a kind of a "public answer" on this topic.

There are currently *no* optimized versions of MUI for 020, 030, 040 or 060 processors. These versions are planned for a later release as a special bonus for registered users. Please do not ask me about specific release dates yet, I really don't know when this will happen. Anyway, I'll do my best to let it happen quickly.

So again, processor optimized versions are not yet done but will be available some day. However, they'll only work for registered users. The public demo will always use plain 68000 code.

I apologize for including the string "generic 68000 version" in the About window of MUI 3.8. This mistake was probably the cause for the current confusion about the availability of other releases. Please be patient.

Let me answer another big part of the questions in my email folder as well. Due to popular demand, you can now use the secure First Virtual internet payment system to register MUI online. Please check our web site at <http://www.sasg.com/> for details.

--

```
> Does the same apply to Application class with Window objects?
>
> I hope that you (Stefan) will answer both questions with yes, because that
> simplifies a lot concerning dynamic adding of children to objects.
```

This is completely wrong and can cause major trouble! Never ever do this!

```
--
Greetings, Stefan
```

1.36 The Hacker's Guide to MUI - Others - Clipping

4.5 Clipping

Christian Stieber wrote

```
> clip the cells at the edges. Normal area class objects are still
> difficult to clip (lots of "assume MUI does it this way" stuff around
> the InstallClipRegion() (or whatever the function is called -:())
> (just a hint, in case Stefan reads this...).
```

Normal area class objects are very easy to clip :) as long as you use the muimaster.library calls MUI_InstallClipping() and MUI_RemoveClipping(). Never ever use any other clip region calls besides those provided by MUI!

```
--
Greetings, Stefan
```

1.37 The Hacker's Guide to MUI - Others - AboutMUI

4.6 AboutMUI calling techniques

Without notify (ie return ids):

```
case MENU_ABOUTMUI:
    DoMethod(ApplicationObject, MUIM_Application_AboutMUI, MainWindow);
```

With notify:

```
DoMethod(MENU_ABOUTMUI, MUIM_Notify, MUIA_Menuitem_Trigger, MUIV_EveryTime,
    ApplicationObject, 2, MUIM_Application_AboutMUI, MainWindow);
```

1.38 The Hacker's Guide to MUI - Others - MUIC_ImageDisplay & MUIC_Popimage

4.7 MUIC_Imagedisplay & MUIC_Popimage

MUIA_Imagedisplay_Spec is the same for MUIC_Popimage object

as MUIA_Pendisplay_Spec is for MUIC_Poppen object.

MUIA_Imageadjust_Type permit to choose what kind of images will be selectable in the popup window, giving it MUIV_Imageadjust_xxx special values.

struct MUI_ImageSpec is the same as struct MUI_PenSpec but for image/popimage objects.

an example:

```
Child, data->mcp_PenList = PoppenObject,
  MUIA_CycleChain, 1,
  MUIA_Window_Title, "List Pen",
  MUIA_Draggable, TRUE,
  MUIA_ShortHelp, "Adjust Color of List Pen.",
End,
```

```
Child, data->mcp_BG_List = MUI_NewObject(MUIC_Popimage,
  MUIA_CycleChain, 1,
  MUIA_Imageadjust_Type, MUIV_Imageadjust_Type_Background,
  MUIA_Window_Title, "List Background",
  MUIA_Draggable, TRUE,
  MUIA_ShortHelp, "Adjust List Background.",
End,
```

In MUIM_Settingsgroup_ConfigToGadgets :

```
{
  LONG ptrd;
  if (ptrd = DoMethod(msg->configdata, MUIM_Dataspace_Find, MUICFG_Pen_List))
    set(data->mcp_PenList, MUIA_Pendisplay_Spec, ptrd);
  else
    set(data->mcp_PenList, MUIA_Pendisplay_Spec, DEFAULT_PEN_LIST);
}

{
  LONG ptrd;
  if (ptrd = DoMethod(msg->configdata, MUIM_Dataspace_Find, MUICFG_BG_List))
    set(data->mcp_BG_List, MUIA_Imagedisplay_Spec, ptrd);
  else
    set(data->mcp_BG_List, MUIA_Imagedisplay_Spec, DEFAULT_BG_LIST);
}
```

In MUIM_Settingsgroup_GadgetsToConfig :

```
{
  LONG ptrd;
  get(data->mcp_PenList, MUIA_Pendisplay_Spec, &ptrd);
  if (ptrd)
    DoMethod(msg->configdata, MUIM_Dataspace_Add, ptrd, sizeof(struct MUI_PenSpec ←
      ), MUICFG_Pen_List);
}
{
  LONG ptrd;
```

```

get(data->mcp_BG_List, MUIA_Imagedisplay_Spec, &ptrd);
if (ptrd)
    DoMethod(msg->configdata, MUIM_Dataspace_Add, ptrd, sizeof(struct ←
        MUI_ImageSpec), MUICFG_BG_List);
}

```

After, a successfull

```

DoMethod(obj, MUIM_GetConfigItem, MUICFG_Pen_List, &ptrd)
will give a (struct MUI_PenSpec *) ptrd usable in MUI_ObtainPen()
for example.

```

And a successfull

```

DoMethod(obj, MUIM_GetConfigItem, MUICFG_BG_List, &ptrd)
will give a (struct MUI_ImageSpec *) ptrd usable as MUIA_Background
or MUIA_Image_Spec value.

```

1.39 The Hacker's Guide to MUI - Others - MUIC_Framedisplay && MUIC_Popframe

4.8 MUIC_Framedisplay & MUIC_Popframe

See

chapter 4.7
, they would probably work similar.

If someone has a complete example or a better explanation for both 4.7 & 4.8, simply email me!

These classes are really needed for a configurable application and they should be documented in the official MUI documentation.

MUIA_Framedisplay_Spec is defined as 0x... ,
but i don't really know where to use the FrameSpec in an Area class.

For better understanding look there:

```

*****
** Class Tree
*****
**
** rootclass                (BOOPSI's base class)
** +--Notify                (implements notification mechanism)
** ! +--Area                (base class for all GUI elements)
** !   +--Framedisplay      (displays frame specification)
** !   ! \--Popframe        (popup button to adjust a frame spec)
** !   +--Imagedisplay      (displays image specification)
** !   ! \--Popimage        (popup button to adjust an image spec)
** !   +--Pendisplay        (displays a pen specification)
** !   ! \--Poppen          (popup button to adjust a pen spec)
** !   +--Group              (groups other GUI elements)
** !       +--Register      (handles page groups with titles)
** !       ! \--Penadjust    (group to adjust a pen)

```

```

** !      +--Frameadjust      (group to adjust a frame)
** !      +--Imageadjust      (group to adjust an image)

```

1.40 The Hacker's Guide to MUI - Others - mad_Flags & mri_Flags

4.9 mad_Flags & mri_Flags

```

<This section is completely unofficial, results are not guaranteed>
<The define names are invented by me. Use at your own risk      >

```

Unofficial flags: (valid only while MUIM_Draw)

```

#define MADF_OBJECTVISIBLE      (1<<14) // The object is visible

#define MUIMRI_INVIRTUALGROUP   (1<<29) // The object is inside a virtual group
#define MUIMRI_ISVIRTUALGROUP   (1<<30) // The object is a virtual group

```

A note for Stefan: Don't hate me... i really like knowledge :-)

1.41 The Hacker's Guide to MUI - Others - _parent(obj)

4.10 _parent(obj)

If you want to use `_parent(obj)` you need to change the structure `MUI_NotifyData`. His `priv2` field is equal to `xget(obj,MUIA_Parent)`.

<muiundoc.h>

```

struct MUI_NotifyData
{
    struct MUI_GlobalInfo *mnd_GlobalInfo;
    ULONG                mnd_UserData;
    ULONG                mnd_ObjectID;
    ULONG priv1;
    Object               *mnd_ParentObject; // The name may not be the real one
    ULONG priv3;
    ULONG priv4;
};

#define _parent(obj)      (muiNotifyData(obj)->mnd_ParentObject) /* valid between ↔
    MUIM_Setup/Cleanup */

```

Please use `xget(obj,MUIA_Parent)` instead.

1.42 The Hacker's Guide to MUI - Internals

5 MUI internals

5.1 MUIA_Background

5.2 AppMessage

5.3 MUI_DisposeObject

1.43 The Hacker's Guide to MUI - Internals - MUIA_Background

5.1 MUIA_Background

Steve Koren wrote in article <yam6986.2967.143671672@mail.frii.com>:

> I have a MUIA_Background problem I don't quite understand.

MUI tries to do some tricky optimizations on backgrounds. Eg, when a window is first opened, it's filled completely with the window background. The following gadget rendering will skip drawing the background again if it's equal to the windows background.

>From your explanations, it sounds like MUI messes something up here. But

I currently dont have any clue why and how and where.

--

Greetings, Stefan

1.44 The Hacker's Guide to MUI - Internals - AppMessage

5.2 AppMessage

Trond Werner Hansen wrote in article

<Pine.SGI.3.95.970208155124.13090B-100000@storm.stud.ntnu.no>:

> What kind of "checks" does MUI do on a received AppMessage?
 > I can't get MUI apps to "understand" appmessages sent by
 > my Workbench replacement. The AppMessages are????@lied by ???Â but
 > nothing happens. (ie. no entries are added to, say, the AppWindow demo in
 > the MUI-archive)

```
#define IsAppMessage(msg) (((struct AppMessage *)msg)->am_Type== ←
    AMTYPE_APPWINDOW && ((struct AppMessage *)msg)->am_ID==APPMESSAGE_ID)
```

```
if (IsAppMessage(msg))
{
    if (((struct AppMessage *)msg)->am_UserData)
    {
```

```

        set((APTR)((struct AppMessage *)img) -> am_UserData, MUIA_AppMessage, img);
    }

    ReplyMsg((struct Message *)img);
}

```

1.45 The Hacker's Guide to MUI - Internals - MUI_DisposeObject()

5.3 MUI_DisposeObject()

For external classes, MUI stores the library base pointer in the `h_Data` field of the class structure. Thus, MUI closes libraries on `MUI_DisposeObject` only if `(cl->cl_ID && cl->cl_Dispatcher.h_Data)`:

```

VOID __asm __saves MUI_DisposeObject(_a0 APTR obj)
{
    if (obj)
    {
        struct IClass *cl = OCLASS(obj);

        DisposeObject(obj);

        if (cl->cl_ID && cl->cl_Dispatcher.h_Data)
            CloseLibrary((struct Library *)cl->cl_Dispatcher.h_Data);
    }
}

```

1.46 The Hacker's Guide to MUI - Examples

6 Examples

6.1 Datatypes

1.47 The Hacker's Guide to MUI - Examples - Datatypes

6.1 Datatypes

```

#include "symbols.h"

#include "muimaster_protos.h"
#include "muimaster_pragmas.h"

#include <datatypes/pictureclass.h>
#include <clib/datatypes_protos.h>
#include <pragmas/datatypes_pragmas.h>

/*
** Make sure to *always* use V43 datatype tags, even if you dont

```



```

** have a gfxboard yourself and still use older datatypes.
** If you don't, your program will annoy many gfxboard-users!
** Check aminet for developer material!
*/

#include <datatypes/pictureclassex.h>

#define CLASS      MUIC_Dtpic
#define SUPERCLASS MUIC_Area

struct Data
{
    struct Library *dtbase;
#define DataTypesBase (data->dtbase)
    char *name;
    APTR dto;
    struct BitMapHeader *bmhd;
    struct BitMap *bitmap;
};

#include "copyright.h"
/* $setver$ */
static const char UserLibID[] = "$VER: \"CLASS\" 19.10 (05.02.97)\" FULLVERS;
#define VERSION 19
#define REVISION 10
#include "classheader.c"

/* ----- */

ULONG mNew(struct IClass *cl, Object *obj, Msg msg)
{
    struct Data *data;

    if (!(obj=(Object *)DoSuperMethodA(cl,obj,msg)))
        return(0);

    data = INST_DATA(cl,obj);
    ClearInstanceData(data);

    data->name = (char *)GetTagData(MUIA_Dtpic_Name,NULL,inittags(msg));

    /* tell MUI not to care about filling our background during MUIM_Draw */
    set(obj,MUIA_FillArea,FALSE);

    return((ULONG)obj);
}

/*
** free the datatypes object.
** called from cleanup method or when setup failed somehow
*/

static VOID freedto(struct Data *data)

```

```

{
    data->bitmap = NULL;
    data->bmhd = NULL;

    if (data->dto)
    {
        DisposeDTObject (data->dto);
        data->dto = NULL;
    }

    if (data->dtbase)
    {
        CloseLibrary (data->dtbase);
        data->dtbase = NULL;
    }
}

ULONG mSetup(struct IClass *cl, Object *obj, Msg msg)
{
    struct Data *data = INST_DATA(cl, obj);

    if (!DoSuperMethodA (cl, obj, msg))
        return (FALSE);

    if (data->name)
    {
        if (data->dtbase = OpenLibrary ("datatypes.library", 39))
        {
            /* tell DOS not to bother us with requesters */
            struct Process *myproc = (struct Process *)FindTask (NULL);
            APTR oldwindowptr = myproc->pr_WindowPtr;
            myproc->pr_WindowPtr = (APTR)-1;

            /* create the datatypes object */
            data->dto = NewDTObject (data->name,
                DTA_GroupID          , GID_PICTURE,
                OBP_Precision        , PRECISION_EXACT,
                PDTA_Screen          , _screen(obj),
                PDTA_FreeSourceBitMap, TRUE,
                PDTA_DestMode        , MODE_V43,
                PDTA_UseFriendBitMap , TRUE,
                TAG_DONE);

            myproc->pr_WindowPtr = oldwindowptr;

            /* do all the setup/layout stuff that's necessary to get a bitmap from the ↔
             * dto */
            /* note that when using V43 datatypes, this might not be a real "struct ↔
             * Bitmap *" */

            if (data->dto)
            {
                struct FrameInfo fri = {NULL};

                DoMethod (data->dto, DTM_FRAMEBOX, NULL, &fri, &fri, sizeof (struct FrameInfo), 0) ↔
                ;
            }
        }
    }
}

```

```

    if (fri.fri_Dimensions.Depth>0)
    {
        if (DoMethod(data->dto,DTM_PROCLAYOUT,NULL,1))
        {
            get(data->dto,PDTA_BitMapHeader,&data->bmhd);

            if (data->bmhd)
            {
                GetDTAttrs(data->dto,PDTA_DestBitMap,&data->bitmap,TAG_DONE);

                if (!data->bitmap)
                    GetDTAttrs(data->dto,PDTA_BitMap,&data->bitmap,TAG_DONE);

                if (data->bitmap)
                {
                    return(TRUE);
                }
            }
        }
    }
}
freedto(data);
return(TRUE);
}

```

```

ULONG mCleanup(struct IClass *cl,Object *obj,Msg msg)
{
    struct Data *data = INST_DATA(cl,obj);
    freedto(data);
    return(DoSuperMethodA(cl,obj,msg));
}

```

```

ULONG mAskMinMax(struct IClass *cl,Object *obj,struct MUIP_AskMinMax *msg)
{
    struct Data *data = INST_DATA(cl,obj);
    struct MUI_MinMax *mi;

    DoSuperMethodA(cl,obj,msg);

    mi = msg->MinMaxInfo;

    if (data->bitmap)
    {
        mi->MinWidth += data->bmhd->bmh_Width ;
        mi->MinHeight += data->bmhd->bmh_Height;
        mi->DefWidth += data->bmhd->bmh_Width ;
        mi->DefHeight += data->bmhd->bmh_Height;
        mi->MaxWidth += data->bmhd->bmh_Width ;
        mi->MaxHeight += data->bmhd->bmh_Height;
    }

    /* if we have no bitmap, our object's size will be 0 */
}

```

```
    return(0);
}
```

```
ULONG mDraw(struct IClass *cl, Object *obj, struct MUIP_Draw *msg)
{
    struct Data *data = INST_DATA(cl, obj);

    DoSuperMethodA(cl, obj, msg);

    if (msg->flags & MADF_DRAWOBJECT)
    {
        if (data->bitmap)
        {
            BltBitMapRastPort(data->bitmap, 0s/p_rp(obj), _mleft(obj), _mtop(obj), _mwidth( ←
                obj), _mheight(obj), 0xc0);
        }
    }

    return(0);
}
```

```
ULONG __saveds __asm MUI_Dispatcher(register __a0 struct IClass *cl, register __a2 ←
    Object *obj, register __a1 Msg msg)
{
    switch (msg->MethodID)
    {
        case OM_NEW          : return(mNew          (cl, obj, (APTR)msg));
        case MUIM_Setup      : return(mSetup        (cl, obj, (APTR)msg));
        case MUIM_Cleanup    : return(mCleanup      (cl, obj, (APTR)msg));
        case MUIM_AskMinMax : return(mAskMinMax    (cl, obj, (APTR)msg));
        case MUIM_Draw       : return(mDraw         (cl, obj, (APTR)msg));
    }

    return(DoSuperMethodA(cl, obj, msg));
}
```
