

message

COLLABORATORS

	TITLE : message		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY		November 28, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	message	1
1.1	GUIEnvironment/Message Guide	1
1.2	GUIEnvironment Message Handling	1
1.3	Resizing GUI message	2
1.4	Refreshing GUI message	2
1.5	Other message classes	3
1.6	rcs	3

Chapter 1

message

1.1 GUIEnvironment/Message Guide

GUIEnvironment

Message Handling

```
=====
© 1994   Carsten Ziegeler
         Augustin-Wibbelt-Str.7
         D-33106 Paderborn
         Germany
=====
```

General message handling

Gadget messages

Menu message

Resizing message

Refreshing message

Other message classes

1.2 GUIEnvironment Message Handling

Because GUIEnvironment is based on intuition and gadtools it is necessary to use own message handling and parsing !

So GUIEnvironment offers some own message handling functions which have to be called instead of the exec or gadtools message handling functions !

GUIEnvironment uses the window's user port for all message handling as

default message port. This can be changed using the `GUI_MsgPort` tag.

You can use `WaitGUIMsg` to wait for a message concerning the GUI or if you have to wait for different signals, use the `GetGUIMsg` function.

If one of these functions is called and a message arrives, the message entry fields of the `GUIInfo` structure are filled in with the appropriate values.

If you have specified a hook function, now this hook is called. You can examine this message and handle it.

If you don't want `GUIEnv` to handle this message any more, return `TRUE`.

If you return `FALSE`, `GUIEnvironment` now will also handle this message.

When the message was handled by your hook function, the application will never hear of this message ! This is also the case if the message could be handled internal by `GUIEnvironment`.

For more information about the message handling see the chapters about the different message classes.

The `intuiMsg` entry of the `GUIInfo` structure points to a copy of the message. This could be a "real" `IntuiMessage` or if running under OS3.0+ a extended `IntuiMessage` !

The `msgClass` and the `msgCode` fields contain the belonging message information. It might be that they differ from the entries in the `intuiMsg` copy ! This happens every time `GUIEnvironment` converts message, for example for the key equivalents !

SEE ALSO

The `GUIInfo` structure

The message hook

1.3 Resizing GUI message

Every time a `IDCMP_NEWSIZE` message arrives, `GUIEnvironment` tries to resize the GUI. This is only possible if you have used the gadget description flags !

If the GUI can't be resized, the application will get this message, otherwise it won't hear of any resizing message !

Actually, if the application really should get this message, it has to free the GUI first and then to recreate it or to exit ! Because the first time an error occurs concerning the GUI, `GUIEnvironment` refuses to handle this GUI !

1.4 Refreshing GUI message

`GUIEnvironment` does all the refreshing for the application. The refresh must be done by `GUIEnvironment`, so your hook function should NOT handle this message class.

You can specify a hook function which is called with every refreshing message. It is called after `GUIEnvironment` has done it's refreshing and between the calls to gadtools `GTBeginRefresh` and `GTEndRefresh` !

SEE ALSO

The refreshing hook

1.5 Other message classes

All other message classes must be handled by the application. If a `IDCMP_VANILLAKEY` message arrives you can use the `msgCharCode` entry of the `GUIInfo` structure to ask for the message character code ! But use this entry ONLY WITH `IDCMP_VANILLAKEY` messages !

1.6 rcs

`$RCSfile: Message.guide $`

`$Revision: 1.5 $`

`$Date: 1994/11/03 15:52:27 $`

GUIEnvironment Message Handling Guide

Copyright © 1994, Carsten Ziegeler

Augustin-Wibbelt-Str.7, 33106 Paderborn, Germany