

toolbar

COLLABORATORS

	<i>TITLE :</i> toolbar		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		August 22, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	toolbar	1
1.1	toolbar_plugin: Introduction	1
1.2	toolbar_plugin: Constructors / Destructor	1
1.3	toolbar_plugin: New Methods	2
1.4	toolbar_plugin: Tags	2
1.5	toolbar_plugin: Exceptions	4
1.6	toolbar_plugin: History	4

Chapter 1

toolbar

1.1 toolbar_plugin: Introduction

toolbar_plugin

by Ali Graham <agraham@hal9000.net.au>

toolbar_plugin is a PLUGIN that places several gadgets in a horizontal or vertical row, and calls a specified function when one of them is pressed.

- Constructor
- Methods
- Tags
- Exceptions
- History

1.2 toolbar_plugin: Constructors / Destructor

Constructor

```
toolbar(tags=NIL;PTR TO tagitem)
```

For creating a new plugin object use for example:

```
DEF toolbar:PTR TO toolbar_plugin  
NEW toolbar.toolbar([..., TAG_DONE])
```

1.3 toolbar_plugin: New Methods

```
set(tag, value)
```

By calling this method it's possible to change attributes at runtime. You can use all tags with the S flag set. This method can also be used before the GUI is created and when the window is closed. Value is a LONG and contains the argument for the used tag.

Example:

```
DEF toolbar:PTR TO toolbar_plugin
...
NEW toolbar.toolbar([..., TAG_DONE])
...
toolbar.set(PLA_ToolBar_Disabled, TRUE)
...
```

```
value,check:=get(tag)
```

This method is the counterpart to set. All tags with G flag can be used. Argument is the tag you want to get. Return values are the requested value and as second a boolean value. So if check is FALSE the used tag can't be get.

```
...
value,check:=toolbar.get(PLA_ToolBar_Disabled)
...
```

After this:

```
value=TRUE
check=TRUE
```

But if you try:

```
...
value,check:=toolbar.get(PLA_ToolBar_DisplayAll)
...
```

Then you get this:

```
value:=-1
check:=FALSE
```

1.4 toolbar_plugin: Tags

The letters [ISG] show you when the tags can be used.

I = Initialisation

S = Set Method

G = Get Method

```
PLA_ToolBar_Contents      [I..]
```

The text contents of the gadgets in the toolbar; this should be passed in as a list of strings, e.g.

```
[PLA_ToolBar_Contents,  ['First', 'Second', 'Third'],  
TAG_DONE]
```

PLA_ToolBar_Disabled [ISG]

Disable or enable the toolbar. Setting this tag causes the gadgets to become disabled; they are all ghosted.

PLA_ToolBar_DisplayAll [I..]

Setting this tag forces the PLUGIN to display all of the gadgets. The default for this tag is FALSE; this means that the PLUGIN will show a minimum of one gadget, and more depending on how much space it is allocated.

PLA_ToolBar_Font [I.G]

The font that the gadgets should all use. This is a pointer to a textattr structure which represents an available font. Default is NIL; this means that the PLUGIN will use the window's font.

PLA_ToolBar_Function [IS.]

The address of a function which will be called when a button on the toolbar is pressed, passed in like this:

```
PLA_ToolBar_Function,  {toolbar_pressed},
```

The function should be of the form

```
PROC toolbar_pressed(toolbar:PTR TO toolbar_plugin, gad_num)
```

toolbar will be a PTR to the PLUGIN that was pressed, and gad_num will be the number of the gadget that was pressed, with the first gadget being 1 (not 0).

PLA_ToolBar_Vertical [I..]

Boolean; whether or not the toolbar will be placed vertically. Defaults to FALSE.

1.5 toolbar_plugin: Exceptions

Constructor

"util" will be raised if the utility.library has not been opened.

1.6 toolbar_plugin: History

v1.0 (28.11.97)

- o Initial release.

v1.1 (25.3.98)

- o Now areas outside the gadgets in the toolbar's space are ghosted if toolbar layout has finished in that direction.
 - o PLA_ToolBar_DisplayAll toolbars are now resizable as well.
-