Post a message to the widget and the logger

Synopsis
post()

Description
This is an event. When sent to this widget the value of this event (a string) will be displayed in the widget and sent to the default logger via note.

---

messageline.postnoforward - Function

Package	**display**
Module	**widgets**
Tool	**messageline**

Post a message to the widget

Synopsis
postnoforward()

Description
This is an event. When sent to this widget the value of this event (a string) will be displayed in the widget without forwarding the message on to the logger.

1.4.24 optionmenu - Tool

Package	**display**
Module	**widgets**

Tk widget where the menu label changes with selection
include "optionmenu.g"

Constructors

optionmenu	*Construct an optionmenu*

Functions

disabled	*Disable or enable the optionmenu widget*
done	*Destroy ourselves*
extend	*Extend the menu with more items*
findlabel	*Find the index of the item with the given label*
findname	*Find the index of the item with the given name*
findvalue	*Find the index of the item with the given value*
geteventmask	*Get the event mask*
getindex	*Get the index of the current selection*
getlabel	*Get the label of the current selection*
getlabels	*Gets all of the labels associated with the menu*
getname	*Get the name of the current selection*
getnames	*Gets all of the names associated with the menu*
getpreviousindex	*Get the index previous to the current selection*
getvalue	*Get the value of the current selection*
getvalues	*Gets all of the values associated with the menu*
isenabled	*Is the optionmenu widget enabled ?*
replace	*Replace the menu with the provided items*
selectindex	*Select an item in the menu by index*
selectlabel	*Select an item in the menu by label*
selectname	*Select an item in the menu by name*
selectvalue	*Select an item in the menu by value*
setbackground	*Set the color of the background of the optionmenu button*
seteventmask	*Set the event mask*
setforeground	*Set the color of the foreground of the optionmenu button*
setlabel	*Set the label of the optionmenu*
setwidth	*Set the width of the optionmenu button*

Description

This Tk widget enables you to create a flat (i.e. no submenus) menu in a specified frame. You can associate a label (that which appears on the top menu button), a name (that which appears in the menu list) and a value (whatever you like, it may be different from the label) for each menu item. When you select one of the items in the menu, the label of the menu button changes to reflect the label of the selected item.
If your list of items is very long, a menu may be too long to sensibly manage. Therefore, you can optionally choose (see constructor) to use a popup scrolling listbox rather than a menu (it's really implemented with a selectabelist widget). This widget can be treated like any of the built in Tk widgets, and has almost all of the same construction arguments as the button Tk widget.
Note that the `width` *argument can be used in three ways. If* `width=-1` *(the default), the menu button will be wide enough (and of fixed width) to accomodate all of the item labels in the menu. This means that the GUI won't jitter when you write different length labels into it. If* `width=0`*, then the menu button will change*

its width to accomodate whatever label is written into it. If `width > 0`*, then the button is again of fixed width, with the width that you specify.*
Whenever a selection is made, this widget emits an event called `select`*. In the $value of this event, you can find a record with fields* `label`, `name`, `value`*, and* `index` *(index of item in menu) containing that information for the selected item. The event generation can be disabled if you desire with the seteventmask tool function.*
Whenever the menu is replaced with the replace tool function, this widget emits an event called `replaced` *(it has no value). Note that the tool function extend invokes the* `replace` *function, so a* `replaced` *event is generated then as well.*
The widget accepts an event called `setwidth`*. It is the same as calling the setwidth tool function.*

optionmenu - Constructor

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Construct an optionmenu

Synopsis

optionmenu(parent, labels, names, values, hlp, hlp2, nbreak, padx, pady, width, updatelabel, height, justify, font, relief, borderwidth, foreground, background, disabled, anchor, fill, widgetset)

Arguments

parent	*Parent of the menu* *Allowed: Widget*
labels	*The menu item labels* *Allowed: Vector of strings* *Default: Empty menu*
names	*The menu item names* *Allowed: Vector of strings* *Default: Empty menu*
values	*The menu item values* *Allowed: Vector of anything* *Default: Empty menu*
hlp	*Short popuphelp text* *Allowed: String* *Default: No help*
hlp2	*Long popuphelp text* *Allowed: String* *Default: No help*
nbreak	*The break point for menu or listbox usage* *Allowed: Integer* *Default: 20*
padx	*Horizontal padding around text* *Allowed: Integer* *Default: 7*
pady	*Vertical padding around text* *Allowed: Integer* *Default: 3*
width	*Width in character units* *Allowed: Integer* *Default: Auto-scale*
updatelabel	*Update menu label with selected item* *Allowed: T or F* *Default: T*
height	*Height in character units* *Allowed: Integer* *Default: 1*
justify	*Justification of text* *Allowed: String from 'center', 'left', 'right'* *Default: 'center'*
font	*Font of text* *Allowed: X font*
relief	*Edge relief* *Allowed: String from 'flat', 'ridge', 'raised', 'sunken', 'groove'* *Default: 'groove'*
borderwidth	*Border width* *Allowed: Integer* *Default: 2*
foreground	*Color of text* *Allowed: X color* *Default: 'black'*
background	*Background color* *Allowed: X color* *Default: 'lightgrey'*
disabled	*Is inactivated ?* *Allowed: Boolean* *Default: F*
anchor	*Location of text* *Allowed: String from 'c', 'n', 's', 'e', 'w', 'ne', 'nw', 'se', 'sw'* *Default: 'c'*
fill	*How to expand when resized* *Allowed: String from 'x', 'y', 'both', 'none'* *Default: 'none'*
widgetset	*The widget server to use* *Allowed: widgetserver tool* *Default: dws*

Returns
Agent

Description
This is the constructor. Note that if you give just the `labels` *argument, then* `names` *and* `values` *are set equal to* `labels`. *If you give either of* `names` *or* `values`, *you must also give* `labels` *(it is the primary).*

Example

```
- include 'optionmenu.g'
- labels := "Absolute RelRef RelCen";
- names := ['Absolute', 'Relative to ref. pixel', 'Relative to center'];
- values := [1,2,3];
- f := frame();
- hlp := 'Select type of coordinate';
- m := optionmenu(f, labels, names, values, hlp);
```

Creates an optionmenu with a popup menu.

Example

```
- include 'optionmenu.g'
- labels := "Absolute RelRef RelCen";
- names := ['Absolute', 'Relative to ref. pixel', 'Relative to center'];
- values := [1,2,3];
- f := frame();
- hlp := 'Select type of coordinate';
- m := optionmenu(f, labels, names, values, hlp, nbreak=1)
```

Creates an optionmenu with a popup listbox.

---

optionmenu.extend - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Extend the menu with more items

Synopsis
extend(labels, names, values, width)

Arguments

labels	*The menu item labels* *Allowed: Vector of strings* *Default: Empty menu*
names	*The menu item names* *Allowed: Vector of strings* *Default: Empty menu*
values	*The menu item values* *Allowed: Vector of anything* *Default: Empty menu*
width	*Width in character units* *Allowed: Integer* *Default: Auto-scale*

Returns
Bool

Description

This tool function extends the menu with the supplied items.

Example

```
- labels := "Absolute RelRef RelCen";
- f := frame();
- m := optionmenu(f, labels);
- m.extend(labels="RelDir");
- m.getlabels();
Absolute RelRef RelCen RelDir
-
```

---

optionmenu.replace - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Replace the menu with the provided items

Synopsis
replace(labels, names, values, width)

Arguments

labels	*The menu item labels* *Allowed: Vector of strings* *Default: Empty menu*
names	*The menu item names* *Allowed: Vector of strings* *Default: Empty menu*
values	*The menu item values* *Allowed: Vector of anything* *Default: Empty menu*
width	*Width in character units* *Allowed: Integer* *Default: Auto-scale*

Returns
Bool

Description

This tool function replaces the menu with the supplied items.

Example

```
- labels := "Absolute RelRef RelCen";
- f := frame();
- m := optionmenu(f, labels);
m.getlabels();
Absolute RelRef RelCen
- m.replace("a b c", "A B C", [10,20,40]);
- m.getlabels();
a b c
- m.getnames();
A B C
- m.getvalues();
[10 20 40]
-
```

optionmenu.selectindex - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Select an item in the menu by index

Synopsis
selectindex(idx)

Arguments

idx	*The menu item index* *Allowed: Integer*

Returns
Bool

Description

This tool function selects a menu item by its index. It is equivalent to selecting the item with the GUI interface. If the index is invalid, it returns `F`*.*

Example

```
- labels := "Absolute RelRef RelCen";
- f := frame();
- m := optionmenu(f, labels);
- m.selectindex(3);
- m.getname();
RelCen
-
```

optionmenu.selectlabel - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Select an item in the menu by label

Synopsis
selectlabel(label)

Arguments

label	*The menu item label* *Allowed: Same type as the array "labels" used in construction*

Returns
Bool

Description

This tool function selects a menu item by its label. It is equivalent to selecting the item with the GUI interface. If the label is invalid, it returns `F`.

Example

```
- labels := "Absolute RelRef RelCen";
- values := [10, 20, 30];
- f := frame();
- m := optionmenu(f, labels, values=values);
- m.selectlabel('RelRef');
- m.getlabel();
RelRef
-
```

---

optionmenu.selectname - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Select an item in the menu by name

Synopsis
selectname(name)

Arguments

name	*The menu item name*	
	Allowed:	*Same type as the array "names" used in construction*

Returns
Bool

Description

This tool function selects a menu item by its name. It is equivalent to selecting the item with the GUI interface. If the name is invalid, it returns F*.*

Example

```
- labels := "Absolute RelRef RelCen";
- names := "A B C";
- f := frame();
- m := optionmenu(f, labels, names);
- m.selectname("C");
- m.getindex();
3
-
```

---

optionmenu.selectvalue - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Select an item in the menu by value

Synopsis
selectvalue(value)

Arguments

value	*The menu item value* *Allowed: Same type as the array "values" used in construction*

Returns
Bool

Description

This tool function selects a menu item by its value. It is equivalent to selecting the item with the GUI interface. If the value is invalid, it returns `F`*.*

Example

```
- labels := "Absolute RelRef RelCen";
- values := [10, 20, 30];
- f := frame();
- m := optionmenu(f, labels, values=values);
- m.selectvalue(20);
- m.getname();
RelRef
-
```

---

optionmenu.getindex - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Get the index of the current selection

Synopsis
getindex()

Returns
Integer or F

Description

This tool function recovers the index of the currently selected item in the menu. If the menu is empty, it returns `F`*. If no menu item has been selected by the user, it returns unity.*

Example

```
- labels := "Absolute RelRef RelCen";
- f := frame();
- m := optionmenu(f, labels=labels);
- m.selectindex(2);
- m.getindex();
2
```

---

optionmenu.getpreviousindex - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Get the index previous to the current selection

Synopsis

getpreviousindex()

Returns

Integer or F

Description

This tool function recovers the index that you selected previous to the current selection. If the menu is empty, it returns `F`*. If no menu item has been selected by the user, it returns unity.*

Example

```
- labels := "Absolute RelRef RelCen";
- f := frame();
- m := optionmenu(f, labels=labels);
- m.selectindex(2);
- m.selectindex(3);
- m.getpreviousindex();
2
```

---

optionmenu.getlabel - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Get the label of the current selection

Synopsis
getlabel()

Returns
String or F

Description

This tool function recovers the label of the currently selected item in the menu. If the menu is empty, it returns `F`*. If no menu item has been selected by the user, it returns the first item.*

Example

```
- labels := "Absolute RelRef RelCen";
- f := frame();
- m := optionmenu(f, labels=labels);
- m.selectindex(2);
- m.getlabel();
RelRef
```

---

optionmenu.getlabels - Function

Package display
Module widgets
Tool optionmenu

Gets all of the labels associated with the menu

Synopsis
getlabels()

Returns
Vector of String

Description

This tool function recovers the labels associated with each item in the menu. If the menu is empty, it returns an empty array.

Example

```
- labels := "Absolute RelRef RelCen";
- f := frame();
- m := optionmenu(f, labels);
- m.getlabels();
Absolute RelRef RelCen
```

---

optionmenu.getname - Function

Package display
Module widgets
Tool optionmenu

Get the name of the current selection

Synopsis
getname()

Returns
String or F

Description

This tool function recovers the name of the currently selected item in the menu. If the menu is empty, it returns `F`*. If no menu item has been selected by the user, it returns the first item.*

Example

```
- labels := "Absolute RelRef RelCen";
- names := ['Absolute', 'Relative to ref. pix', 'Relative to center'];
- f := frame();
- m := optionmenu(f, labels=labels, names=names);
- m.selectindex(2);
- m.getname();
Relative to ref. pix
```

---

optionmenu.getnames - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Gets all of the names associated with the menu

Synopsis
getnames()

Returns
Vector of String

Description

This tool function recovers the names associated with each item in the menu. If the menu is empty, it returns an empty array.

Example

```
- labels := "Absolute RelRef RelCen";
- names := ['Absolute', 'Relative to ref. pix', 'Relative to center'];
- f := frame();
- m := optionmenu(f, labels=labels, names=names);
- m.selectindex(2);
- m.getnames();
Absolute Relative to ref. pix Relative to center'
```

---

optionmenu.getvalue - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Get the value of the current selection

Synopsis
getvalue()

Returns
Same type as "values" array in constructor or Fail

Description

This tool function recovers the value of the currently selected item in the menu. If the menu is empty, it returns a `Fail`. *If no menu item has been selected by the user, it returns the first item.*

Example

```
- labels := "Absolute RelRef RelCen";
- values := [1,2,3];
- f := frame();
- m := optionmenu(f, labels=labels, values=values);
- m.selectindex(2);
- m.getvalue();
2
```

---

optionmenu.getvalues - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Gets all of the values associated with the menu

Synopsis
getvalues()

Returns
The "values" array given (explicitly or implicitly) in the constructor

Description

This tool function recovers the values associated with each item in the menu. If the menu is empty, it returns an empty array.

Example

```
- labels := "Absolute RelRef RelCen";
- values := [1,2,3]
- f := frame();
- m := optionmenu(f, labels, values=values);
- m.getvalues();
[1 2 3]
```

---

optionmenu.findlabel - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Find the index of the item with the given label

Synopsis
findlabel(label)

Arguments

label	*The menu item label*	
	Allowed:	*Same type as the array "labels" used in construction*

Returns

Integer or F

Description

This tool function find the index in the menu of the item specified by its label. If you give an invalid label, F *is returned.*

Example

```
- labels := "Absolute RelRef RelCen";
- values := [10, 20, 30];
- f := frame();
- m := optionmenu(f, labels, values=values);
- m.findlabel('Absolute');
1
```

---

optionmenu.findname - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Find the index of the item with the given name

Synopsis

findname(name)

Arguments

name	*The menu item name*	
	Allowed:	*Same type as the array "names" used in construction*

Returns
Integer or F

Description

This tool function find the index in the menu of the item specified by its name. If you give an invalid name, `F` *is returned.*

Example

```
- labels := "Abs RelRef RelCen";
- names := ['Absolute', 'Relative to ref. pixel', 'Relative to center'];
- values := [10, 20, 30];
- f := frame();
- m := optionmenu(f, labels, names, values);
- m.findname('Relative to ref. pixel');
2
```

---

optionmenu.findvalue - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Find the index of the item with the given value

Synopsis
findvalue(value)

Arguments

value	*The menu item value* *Allowed: Same type as the array "value" used in construction*

Returns
Integer or F

Description

This tool function find the index in the menu of the item specified by its value. If you give an invalid value, `F` *is returned.*

Example

```
- labels := "Absolute RelRef RelCen";
- values := [10, 20, 30];
- f := frame();
- m := optionmenu(f, labels, values=values);
- m.findvalue(20);
2
-
```

---

optionmenu.geteventmask - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Get the event mask

Synopsis
geteventmask()

Returns
Vector of Bool

Description

This tool function recovers the event mask. This mask indicates whether the `select` *event will be emitted or not when you select a menu item. By default the mask is all T. You can modify it with the seteventmask tool function.*

Example

```
- labels := "Absolute RelRef RelCen";
- values := [1,2,3];
- f := frame();
- m := optionmenu(f, labels=labels, values=values);
- m.geteventmask();
[T T T]
- m.seteventmask([T,F,T]);
- m.geteventmask();
[T F T]
```

---

optionmenu.seteventmask - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Set the event mask

Synopsis

seteventmask(mask)

Arguments

mask	*The event mask* *Allowed: Vector of Bool*

Returns

Bool

Description

This tool function sets the event mask. This mask indicates whether the `select` *event will be emitted or not when you select a menu item. By default the mask is all T. You can get the current mask with the geteventmask tool function. If the mask is invalid, it returns* `F`*.*

Example

```
- labels := "Absolute RelRef RelCen";
- values := [1,2,3];
- f := frame();
- m := optionmenu(f, labels=labels, values=values);
- m.seteventmask([T,F,T]);
- m.geteventmask();
[T F T]
- whenever m->select do {print $value;};
```

In this example, when you selected the second item, no event would be emitted.

---

optionmenu.setlabel - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Set the label of the optionmenu

Synopsis
setlabel(label)

Arguments

label	*The label* *Allowed: String*

Returns
Bool

Description

This tool function sets the label of the optionmenu. This label can be anything, it does not have to be in the optionmenu list of labels.

Example

```
- labels :=
- values := [1,2,3];
```

```
- f := frame();
- m := optionmenu(f, labels=labels, values=values);
- m.setlabel('fish');
- m.getlabel()
fish
- m.getlabels()
Absolute RelRef RelCen
```

Although the label of the menu is now "fish", the labels of the items in the menu are unchanged.

---

optionmenu.setforeground - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Set the color of the foreground of the optionmenu button

Synopsis
setforeground(color)

Arguments

color	*The color* *Allowed: Any X color encoded as a String*

Returns
Bool

Description

This tool function sets the color of the foreground of the optionmenu button.

---

optionmenu.setbackground - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Set the color of the background of the optionmenu button

Synopsis
setbackground(color)

Arguments

color	*The color* *Allowed: Any X color encoded as a String*

Returns
Bool

Description

This tool function sets the color of the background of the optionmenu button.

---

optionmenu.setwidth - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Set the width of the optionmenu button

Synopsis
setwidth(width)

Arguments

width	*The width in character units* *Allowed: Integer*

Returns
Bool

Description

This tool function sets the width of the optionmenu button in character units.

---

optionmenu.disabled - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Disable or enable the optionmenu widget

Synopsis
disabled(disable)

Arguments

disable	*disable the widget* *Allowed: T or F* *Default: T*

Returns
Bool

Description

This tool function disables or enables the optionmenu.

---

optionmenu.isenabled - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Is the optionmenu widget enabled ?

Synopsis
isenabled()

Returns

Bool

Description

This tool function indicated whether the optionmenu has been disabled or not.

Example

```
- f := frame();
- m := optionmenu(f, "dog cat");
- m.disable();
- m.isenabled()
F
-
```

---

optionmenu.done - Function

Package	**display**
Module	**widgets**
Tool	**optionmenu**

Destroy ourselves

Synopsis
done()

Returns
Bool

Description

This tool function destroys the widget.

Example

```
- f := frame();
```