Package 'tclish'

July 6, 2017

Version 1.0.2

Date 2015-07-27

Title 'Tcl/Tk' Utilities Package

Author P. Roebuck

Maintainer P. Roebuck <proebuck@mdanderson.org>

Description Variety of 'Tcl/Tk' methods used for GUI development.

Depends R (>= 2.15)

Imports tcltk

SystemRequirements Tcl/Tk (>= 8.5)

URL http://supercurve.r-forge.r-project.org/#tclish

License Artistic-2.0

Copyright file COPYRIGHTS

LazyLoad yes

Collate tclutil.R tkutil.R tkoptiondb.R tkmessagebox.R tknotebook.R tktabnotebook.R tkprogressbar.R tkappdefaults.R zzz.R

Repository R-Forge

Repository/R-Forge/Project supercurve

NeedsCompilation no

R topics documented:

tclish-package *Tcl/Tk Additions*

Description

A package containing a variety of Tcl/Tk methods used for GUI development.

Details

Package:	tclish
Type:	Package
Version:	1.0.2
Date:	2015-07-27
License:	Artistic-2.0

For a complete list of functions, use `library(help="tclish")`.
For a high-level summary of the changes for each revision, use `file.show(system.file("NEWS", package="tclish")`

Author(s)

P. Roebuck <proebuck@mdanderson.org>

tkappdefaults *Tk options database routines*

Description

This `tkloadappdefaults` method provides a high-level method that allows an application to load the Tk options resource database from various application-default files.

Usage

```
tkloadappdefaults(classNames,
                  priority="startupFile")
```

Arguments

`classNames`	character vector specifying class names, though typically just the name of the application
`priority`	character string (or integer) specifying priority level for this option. Default is `"startupFile"`

Details

The Tk options database is loaded from user-specific defaults files, such as '.Xdefaults', and resource databases loaded into the X server. The package will attempt to load its resources upon startup, following the X11R5 method of merging app-default files from multiple sources. The standard X11 paths are searched first, followed by paths specified by the environment variables XFILESEARCHPATH, XAPPLRESDIR, XUSERFILESEARCHPATH, respectively. Unlike X11, **ALL** matching files will be loaded, not just the first.

The value of each of the environment variables XFILESEARCHPATH and XUSERFILESEARCHPATH is a colon-separated list of pathnames.
The pathnames may contain replacement characters as follows:

%N	value of the application's class name
%T	value of the file's type, the literal string "'app-defaults'"
%C	customization resource
%L	language, locale, and codeset (e.g., "'ja_JP.EUC'")
%l	language part of '%L' (e.g., "'ja'")

If getOption("verbose") is TRUE, a traceback of all pathnames considered is displayed, regardless of existence.

Value

Though invoked for its side-effect, returns (invisibly) a list of all pathnames from which resource loading was attempted.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

optiondb_readfile

Examples

```
  ## Not run:
## Set directory paths for app-defaults processing
homedir <- path.expand("~")
xuserfilesearchpath <- paste(file.path(homedir, "%L", "%T", "%N%C"),
                             file.path(homedir, "%l", "%T", "%N%C"),
                             file.path(homedir, "%T", "%N%C"),
                             file.path(homedir, "%N.ad"),
                             sep=.Platform$path.sep)
Sys.setenv(XAPPLRESDIR=system.file())
Sys.setenv(XUSERFILESEARCHPATH=xuserfilesearchpath)

## Create user 'app-defaults' directory, if necessary
appdefaultsdir <- file.path(homedir, 'app-defaults')
if (!file.exists(appdefaultsdir)) {
    mkdir(appdefaultsdir)
}

## Create some fake resource files
```

```
classname <- "Foo"    ## For an R Tcl/Tk GUI named "foo"
customization <- "-color"
appdefaultsfiles <- c(file.path(homedir,
                                sprintf("%s.ad", classname)),
                      file.path(appdefaultsdir, classname),
                      file.path(appdefaultsdir,
                                sprintf("%s%s", classname, customization)))
file.create(appdefaultsfiles)

## Load application defaults
message('*** loading application defaults')
options(verbose=TRUE)
tkloadappdefaults(classname)
cat('\n')

## Add customization aspect
optiondb_add("*customization", customization)

## Load application defaults
message('*** loading application defaults (customized)')
tkloadappdefaults(classname)

## Cleanup
on.exit(file.remove(appdefaultsfiles))
on.exit(options(verbose=FALSE), add=TRUE)
on.exit(optiondb_add("*customization", ""), add=TRUE)

## End(Not run)
```

`tkmessagebox` *Tk alert dialog routines*

Description

These functions are wrappers around the tkmessageBox() method, in the style of TkInter.

Usage

```
showinfo(title, message, ...)
showwarning(title, message, ...)
showerror(title, message, ...)
askquestion(title, message, ...)
askokcancel(title, message, ...)
askyesno(title, message, ...)
askretrycancel(title, message, ...)
```

Arguments

`title`	character string specifying title for dialog window
`message`	character string specifying message displayed inside the alert
`...`	extra arguments for `tkmessageBox` routine

Details

All methods above display a modal message dialog to allow the user to provide input. The application will wait until the user responds.

A list of other arguments is shown here:

default character string specifying the default button of the dialog

detail character string specifying a secondary message, usually displayed in a smaller font under the main message

parent object of the class `tkwin` representing the window of the application for which this dialog is being posted.

type character string specifying predefined set of buttons to be displayed (`askquestion` only). Possible values are:

abortretryignore displays three buttons whose symbolic names are '`abort`', '`retry`' and '`ignore`'

ok displays one button whose symbolic name is '`ok`'

okcancel displays two buttons whose symbolic names are '`ok`' and '`cancel`'

retrycancel displays two buttons whose symbolic names are '`retry`' and '`cancel`'

yesno displays two buttons whose symbolic names are '`yes`' and '`no`'

yesnocancel displays three buttons whose symbolic names are '`yes`', '`no`' and '`cancel`'

Value

The `askquestion` method returns character string of the button's symbolic name corresponding to the user response.

The `askokcancel` method returns `TRUE` if user responds '`OK`'.

The `askyesno` method returns `TRUE` if user responds '`YES`'.

The `askretrycancel` method returns `TRUE` if user responds '`RETRY`'.

Other methods are invoked for their side-effect.

Note

The `parent` argument controls display location. If specified, the dialog will be centered over the parent widget; otherwise, center screen.

Author(s)

P. Roebuck <`proebuck@mdanderson.org`>

See Also

`tkmessageBox`

Examples

```
  ## Not run:
## Display informational dialog
showinfo('Gumptions', 'Stupid is as stupid does.')

## Ask a question (string response)
button <- askquestion("Proceed?",
```

```
                      "Destroy Western Civilization?",
                      detail="It's going down the tubes anyway...",
                      type="yesnocancel",
                      default="cancel")

## Ask a question (logical response)
toplevel <- tktoplevel()
if (askyesno('Install',
             'Are you sure you want to install SuperVirus?',
             default="no",
             parent=toplevel)) {
    cat("yes", "\n")
} else {
    cat("no", "\n")
}

## End(Not run)
```

---

tknotebook *Tk notebook routines*

Description

These functions provide/control a simulated notebook widget.

Usage

```
notebook_create(parent)
notebook_display(notebook, pagename)
notebook_page(notebook, pagename)
```

Arguments

parent	object of the class tkwin representing the parent widget
notebook	object of the class tkwin representing the notebook widget
pagename	character string specifying name of notebook page

Details

Notebook widgets are created with notebook_create.

The function notebook_page creates a notebook page, a container in which to embed other widgets to display.

The function notebook_display displays the requested page.

Value

The notebook_create method returns a tkwin object representing the notebook.

The notebook_page method returns a tkwin object representing a page in the notebook.

Other methods are invoked for their side-effect.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

Examples

```
  ## Not run:
toplevel <- tktoplevel()
## Create notebook
notebook <- notebook_create(toplevel)
tkpack(notebook, side="bottom", expand=TRUE, fill="both", padx=4, pady=4)
npages <- 0
## Create notebook page and contents
pg1 <- notebook_page(notebook, "Page 1")
label <- tklabel(pg1, bitmap="info")
tkpack(label, side="left", padx=8, pady=8)
label <- tklabel(pg1, text="Something\non\nPage 1")
tkpack(label, side="left", expand=TRUE, pady=8)
npages <- npages + 1
## Create another notebook page and contents
pg2 <- notebook_page(notebook, "Page 2")
label <- tklabel(pg2, text="Something else on Page 2")
tkpack(label, side="left", expand=TRUE, padx=8, pady=8)
npages <- npages + 1
## Create page controls
choice.var <- tclVar("1")
controls <- tkframe(toplevel, class="Radiobox")
border <- tkframe(controls, borderwidth=2, relief="groove")
tkpack(border, expand=TRUE, fill="both")
tkpack(controls, side="top", fill="x", padx=4, pady=4)
for (i in seq_len(npages)) {
    pg <- sprintf("Page
    rb <- tkradiobutton(controls,
                        text=paste("Show", pg),
                        variable=choice.var,
                        value=as.integer(i))
    cmd <- substitute(function() notebook_display(notebook, pgname),
                      list(pgname=pg))
    tkconfigure(rb, command=eval(cmd))
    tkpack(rb, side="top", anchor="w")
}

## End(Not run)
```

---

tkoptiondb *Tk options database routines*

Description

These functions allow interaction with the Tk options resource database.

Usage

```
optiondb_add(pattern,
             value,
```

```
            priority="interactive",
            verbose=FALSE)
optiondb_get(widget=".",
            rsrcName,
            rsrcClass,
            verbose=FALSE)
optiondb_readfile(filename,
                priority="userDefault",
                verbose=FALSE)
```

Arguments

pattern	character string containing the option being specified, and consists of names and/or classes separated by asterisks or dots, in the usual X format
widget	object of the class tkwin or character string specifying Tk widget from which to retrieve resource
filename	character string specifying pathname to resource file
value	character string specifying text to associate with pattern
rsrcName	character string specifying resource name
rsrcClass	character string specifying resource class name
priority	character string (or integer) specifying priority level for this option
verbose	logical scalar. If TRUE, displays the Tcl command

Details

optiondb_add adds an entry into Tcl options database with given priority; optiondb_readfile adds a file's contents. To clear an entry, invoke optiondb_add with an empty string for value.

optiondb_get fetches a value from Tcl options database.

There is a fundamental difference between the way Tcl handles options and the way Xt handles options; Tcl has separate priorities and within a level prefers the latest matching option, Xt prefers options with "more exact" specifiers and has no level semantics.

A resource class name begins with a capital letter; the resource name begins with a lowercase letter.

There are four primary priority levels. From lowest to highest, they are:

widgetDefault Level 20. Used for default values hard-coded into widgets.

startupFile Level 40. Used for options specified in application-specific startup files.

userDefault Level 60. Used for options specified in user-specific defaults files, such as .Xdefaults, resource databases loaded into the X server, or user-specific startup files.

interactive Level 80. Used for options specified interactively after the application starts running. (default)

In addition, priorities may be specified numerically using integers between 0 and 100, inclusive. However, the numeric form is discouraged.

Value

The optiondb_get method returns a tclObj object representing the Tcl variable. Use the tclvalue method to extract its character string value.

The other methods are invoked for their side-effect.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

See Also

tclvalue

Examples

```
  ## Not run:
## Set a couple button resource values
optiondb_add("*Button.background", "red", "startupFile")
optiondb_add("*Button.width", 100, "startupFile")

## Even make up "fake" application-specific resources
optiondb_add("*foo", "bar", "userDefault")

## Prove it works
toplevel <- tktoplevel()
tkpack(button <- tkbutton(toplevel)) # large red button appears

## Fetch "fake" application-specific value
foo.obj <- optiondb_get(rsrcName="foo", rsrcClass="Foo")
foo <- as.character(tclvalue(foo.obj))  # bar

## Read an application-defaults file containing resources
optiondb_readfile("/path/to/app-defaults/myapp.ad")

## End(Not run)
```

---

tkprogressbar *Tk progressbar routines*

Description

These functions provide/control a simulated progressbar widget.

Usage

```
progressbar_create(parent, col)
progressbar_value(progressbar, value)
progressbar_updatebarcolor(progressbar, col)
```

Arguments

parent	object of the class tkwin representing the parent widget
progressbar	object of the class tkwin representing the progressbar widget
col	character string specifying progress color
value	numeric scalar specifying progress with value in range [0..100]

Details

Progressbar widgets are created with `progressbar_create`.

The function `progressbar_value` sets the percent completion value that allows the widget to reflect the amount of progress that has been made, filling from left to right; that value is displayed as well.

The function `progressbar_updatebarcolor` changes the color used by the widget to display progress.

Value

The `progressbar_create` method returns a `tkwin` object representing the progressbar.

Other methods are invoked for their side-effect.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

Examples

```
  ## Not run:
toplevel <- tktoplevel()
progbar <- progressbar_create(toplevel, "blue")
tkpack(progbar)
for (complete in seq_len(100)) {
    progressbar_value(progbar, complete)
    Sys.sleep(0.1)
}
progressbar_updatebarcolor(progbar, "green")

## End(Not run)
```

---

tktabnotebook *Tk tabnotebook routines*

Description

These functions provide/control a simulated tabbed notebook widget.

Usage

```
tabnotebook_create(parent)
tabnotebook_display(tabnotebook, tabname)
tabnotebook_page(tabnotebook, tabname)
```

Arguments

`parent`	object of the class `tkwin` representing the parent widget
`tabnotebook`	object of the class `tkwin` representing the tabnotebook widget
`tabname`	character string specifying name of tabnotebook page

Details

Tabnotebook widgets are created with `tabnotebook_create`.

The function `tabnotebook_page` creates a tabbed notebook page, a container in which to embed other widgets to display.

The function `tabnotebook_display` displays the requested page.

Value

The `tabnotebook_create` method returns a `tkwin` object representing the tabnotebook.

The `tabnotebook_page` method returns a `tkwin` object representing a page in the tabbed notebook.

Other methods are invoked for their side-effect.

Author(s)

P. Roebuck <proebuck@mdanderson.org>

Examples

```
  ## Not run:
toplevel <- tktoplevel()
## Create tabnotebook
tabnotebook <- tabnotebook_create(toplevel)
tkpack(tabnotebook, expand=TRUE, fill="both")
## Create tabnotebook page and contents
pg1 <- tabnotebook_page(tabnotebook, "Page 1")
label <- tklabel(pg1, bitmap="info")
tkpack(label, side="left", padx=8, pady=8)
label <- tklabel(pg1, text="Something\non\nPage 1")
tkpack(label, side="left", expand=TRUE, pady=8)
## Create another tabnotebook page and contents
pg2 <- tabnotebook_page(tabnotebook, "Page 2")
label <- tklabel(pg2, text="Something else on Page 2")
tkpack(label, side="left", expand=TRUE, padx=8, pady=8)
## Controls are inherent; click on tabs to change pages
  ## :BUG: Example doesn't work, but it should. Why?

## End(Not run)
```

---

undocumented *Wish I Had Written The Documentation...*

Description

Placeholder until I can document all these routines.

Index