The tcltk2 Package

February 10, 2006

Title  SciViews GUI API - More tcltk

Version  0.9-5

Date  2006-02-10

Depends  R (>= 2.1.0), tcltk

Author  Philippe Grosjean <phgrosjean@sciviews.org>

Description  A series of widgets (themed controls, tktabe, combobox, multi-column list, etc...) and various functions (under Windows: DDE exchange, access to the registry and icon manipulation) to supplement tcltk.

Maintainer  Philippe Grosjean <phgrosjean@sciviews.org>

License  GPL 2 or above, and respective individual licenses for the Tk widgets

URL  http://www.sciviews.org/SciViews-R

R topics documented:

  tclVarFun  ................................................................. 1
  tk2commands ............................................................ 3
  tk2theme ................................................................. 4
  tk2tip ................................................................. 6
  tk2widgets ............................................................. 7
  tk2dde ................................................................. 10
  tk2ico ................................................................. 10
  tk2reg ................................................................. 11

Index  12
tclVarFun

Manipulate R variables and functions from tcl and back

Description

These functions are intended to provide a better "duality" between the name of variables in both R and tcl, including for function calls. It is possible to define a variable with the same name in R and tcl (the content is identical, but copied and coerced in the two respective environments). It is also possible to get the value of a tcl variable from R, and to call a R function from within tcl. These functionalities are provided in the tcltk package, but Tcl variable usually have different internal names as R equivalents.

Usage

makeTclNames(names, unique = FALSE)
tclFun(f, name = deparse(substitute(f)))
tclGetValue(name)
tclSetValue(name, value)
tclVarExists(name)
tclVarFind(pattern)
tclVarname(name, init = "", keep.existing = TRUE)

Arguments

names Transform names so that they are valid for variables in tcl
unique Should these names be unique in the vector?
f An R function. currently, do no support functions with arguments.
name The name of a variable
value The value to place in a variable
pattern A pattern to search for
init Initial value to use when creating the variable
keep.existing If the tcl variable already exist, should we keep its content?

Details

These functions are similar to tclVar() from package tcltk, except for the following change: here, it is possible to propose a name for the created tcl variable, or to set or retrieve the content of a tcl variable that is not mirrored in R.

Value

Most of these functions return a tclVar object.

Author(s)

Philippe Grosjean
tk2commands

See Also

  tk2theme

Examples

### Not run:
### These cannot be run by examples() but should be OK when pasted
### into an interactive R session with the tcltk package loaded

### Tcl functions and variables manipulation

tclVarExists("tcl_version")
tclVarExists("probably_non_existant")
tclVarFind("tcl_*")

# Using tclVarname() and tclGetValue()... intended for better match between R and Tcl

Test <- tclVarname("Test", "this is a test!")
# Now 'Test' exist both in R and in Tcl... In R, you need to use tclvalue(Test) to get
# If a variable already exists in Tcl, its content is preserved using keep.existing =
.Tcl("set A_Variable (just to test...)") # Create a variable in Tcl and assign "just
A_Variable <- tclVarname("A_Variable", "something else?") # Create the dual variable
tclvalue(A_Variable) # Content of the variable is not changed!

# If you want to retrieve the content of a Tcl variable, but do not want to create a
.Tcl("set Another_Variable (something in it)") # Create a Tcl variable, not visible
tclGetValue("Another_Variable") # Get its content in R

### End(Not run)

tk2commands

Tk commands associated with the tk2XXX widgets

Description

These commands supplement those available in the tcltk package to ease manipulation of tk2XXX
widgets.

Usage

  tk2column(widget, action = c("add", "configure", "delete", "names", "cget", "n")
  tk2insert.multi(widget, where = "end", items)
  tk2listinsert(widget, ...)
  tknotetraverse(nb)

Arguments

  widget The widget to which these actions apply
  action Which kind of action?
  where Where are these item added in the list (by default, at the end)
  items The items to add (either a vector for a single line, or a matrix for more items)
  nb A tk2notebook widget
  ... Further arguments to the action
Details

tk2column manipulate columns of a tk2mclistbox widget, tk2insert.multi is used to insert multiple field entries in a tk2mclistbox widget, tk2listinsert eases the insertion of items in a tk2combobox widget, tk2notetraverse helps manage the tabs in a tk2notebook.

Author(s)

Philippe Grosjean

See Also

tk2widgets, tk2theme, tk2tip

---

**tk2theme**

*Set or get the theme in use for themed widgets (tile)*

Description

In tcltk2, we propose a replacement for standard tcltk widgets. These new widgets can be "themed", that is, adapted to take various visual appearances. It can get the visual appearance of various Unix/Linux systems, of Windows (including XP) and of Aqua under MacOS. Custom themes can also be defined, or imported from GTK.

Usage

```r
  tk2theme(theme = NULL)
  tk2theme.list()
  tk2theme.elements()
```

Arguments

- **theme**
  The new theme to use

Value

A list of available themes for `tk2theme.list()` or of theme elements for `tk2theme.elements()`

Note

Themes are provided by the tk package Tile (currently in version 0.6.2). It is automatically installed under Windows, but for other platforms, you must download and compile Tile yourself (http://tktable.sourceforge.net/tile/)

Author(s)

Philippe Grosjean

See Also

tk2widgets, tk2tip
Examples

## Not run:
## These cannot be run by examples() but should be OK when pasted
## into an interactive R session with the tcltk package loaded

## Test all themes one after the other on the demo window
tk2demo <- function() {
  tt <- tktoplevel()
  tktitle(tt) <- "tk2 demo"

  ## Change icon (under Windows)
  Info <- tk2ico.load(res = "information")
  tk2ico.set(tt, Info)
  tk2ico.destroy(Info)

  ## Add a menu with quit and all themes
  topMenu <- tk2menu(tt)
  tkconfigure(tt, menu = topMenu)
  fileMenu <- tk2menu(topMenu, tearoff = FALSE)
  tkadd(fileMenu, "command", label = "Quit",
        command = function() tkdestroy(tt))
  tkadd(topMenu, "cascade", label = "File", menu = fileMenu)
  Themes <- sort(tk2theme.list())
  themeMenu <- tk2menu(topMenu, tearoff = FALSE)
  for (i in 1:length(Themes)) {
    tkadd(themeMenu, "command", label = Themes[i],
          command = eval(parse(text = paste("function() tk2theme(""", Themes[i], ", menu = themeMenu)

  ## TO DO: Add a toolbar

  ## Add a notebook with three tabs
  nb <- tk2notebook(tt)
  tcli1 <- tk2frame(nb)
  tkadd(nb, tcli1, text = "Tab 1")
  tcli2 <- tk2label(nb, text = "Nothing here.")
  tkadd(nb, tcli2, text = "Tab 2")
  tcli3 <- tk2label(nb, text = "Nothing here either.")
  tkadd(nb, tcli3, text = "Tab 3")
  tkselect(nb, tcli1)
  tkgrid(nb)

  ## Add a labelframe with new tile widgets
  tfrml <- tk2labelframe(tcli1, text = "tile widgets")
  tkgrid(tfrml)

# Add a label
  tlab <- tk2label(tfrml, text = "This is a label... hover me to see a tooltip")
  tkgrid(tlab)
# Define a tooltip for the label
  tk2tip(tlab, "A tooltip for the label \ndisplayed on two lines")

# Add a button
  but <- tk2button(tfrml, text = "Click me", width = 12,
command = function() tkmessageBox(title = "tk2 demo",
message = "Hello, world!", icon = "info", type = "ok")
tkgrid(but)

# Add a checkbutton
tcb <- tk2checkbutton(tfrm1, text = "A check button")
tkgrid(tcb)

# Add radio buttons
rbValue <- tclVar("oranges")
rb1 <- tk2radiobutton(tfrm1, text = "Apples", variable = rbValue,
value = "apples")
rb2 <- tk2radiobutton(tfrm1, text = "Oranges", variable = rbValue,
value = "oranges")
tkgrid(rb1, sticky = "w")
tkgrid(rb2, sticky = "w")

# Add a progressbar
tprog <- tk2progress(tfrm1, orient = "horizontal", from = 0, to = 100)
tkgrid(tprog)
tkset(tprog, 0.5)

# Add a slider #### TO DO: fix command!
tscal <- tk2scale(tfrm1, orient = "horizontal", from = 0, to = 100,
command = tkset(tprog, 0.2))
tkgrid(tscal)

## Add a labelframe with additional widgets
tfrm2 <- tk2labelframe(tcli1, text = "additional widgets", width = 50)
tkgrid(tfrm2)

# Add a combobox
cb <- tk2combobox(tfrm2, value = "Banana")
tkgrid(cb)

# Fill the combobox list
fruits <- c("Apple", "Orange", "Banana", "Pear")
# TO DO: change this!!!
#for (i in 1:4)
# tk2listinsert(cb, "end", fruits[i])
tkinsert(cb, "end", "pear")
tkfocus(tt)
return(TRUE)
}
tk2demo()
themes <- tk2theme.list()
for (i in 1:length(themes)) {
tk2theme(themes[i])
Sys.sleep(2)
}
## End(Not run)
(tk2widgets

Description

`tk2tip` provides a simple mechanism to display tooltips on tk widgets when the mouse hovers on top of them.

Usage

```tcl
    tk2tip(widget, message)
    tk2killtip()
```

Arguments

- `widget`: The widget to which a tooltip is attached
- `message`: The message of the tooltip ("" to remove the tooltip)

Note

This implementation is done in pure tcl code

Author(s)

Philippe Grosjean

See Also

`tk2widgets`, `tk2theme`

Examples

```tcl
## Not run:
## These cannot be run by examples() but should be OK when pasted
## into an interactive R session with the tcltk package loaded

    tt2 <- tktoplevel()
    lb <- tk2label(tt2, text = "Move mouse over me, or over the button to see tooltip")
    tkgrid(lb)
    tk2tip(lb, "A tooltip for the label \ndisplayed on two lines")
    but <- tk2button(tt2, text = "OK", width = 10,
                     command = function() tkdestroy(tt2))
    tkgrid(but)
    tk2tip(but, "Another tooltip for the button")
    #### TO DO: show the use of tk2killtip()
    ## End(Not run)
```

---

(tk2widgets

Description

A themed series of tk widget

A series of widgets you can use in your tk windows/dialog boxes.

---

(tk2widgets

Description

A themed series of tk widget

A series of widgets you can use in your tk windows/dialog boxes.
Usage

tk2button(parent, ...)
tk2canvas(parent, ...)
tk2checkbutton(parent, ...)
tk2comboxbox(parent, ...)
tk2entry(parent, ...)
tk2frame(parent, ...)
tk2label(parent, ...)
tk2labelframe(parent, ...)
tk2listbox(parent, ...)
tk2mclistbox(parent, ...)
tk2menu(parent, ...)
tk2menubutton(parent, ...)
tk2message(parent, ...)
tk2notebook(parent, ...)
tk2panedwindow(parent, ...)
tk2progress(parent, ...)
tk2radiobutton(parent, ...)
tk2scale(parent, ...)
tk2scrollbar(parent, ...)
tk2separator(parent, ...)
tk2spinbox(parent, ...)
tk2table(parent, ...)
tk2text(parent, ...)
tk2tree(parent, ...)
tk2treeview(parent, ...)

Arguments

parent The parent window

... Further arguments passed to the widget

Value

The reference to the created widget.

Note

Most of these widgets comes from the tile (currently 0.4) library

Author(s)

Philippe Grosjean

See Also

tk2theme, tk2tip, tk2commands

Examples

## Not run:
## These cannot be run by examples() but should be OK when pasted
## into an interactive R session with the tcltk package loaded
### A tk2notebook example

```r
tt2 <- tk2plevel()
nb <- tk2notebook(tt2)
tcll1 <- tk2label(nb, text = "Nothing here.")
tkadd(nb, tcll1, text = "Test")
tcll2 <- tk2frame(nb)
but <- tk2button(tcll2, text = "Click me", command = function() tkdestroy(tt2))
tkgrid(but)
tkadd(nb, tcll2, text = "Button")
tkselect(nb, tcll2)
tkgrid(nb)
```

### A simple tk2panedwindow example

```r
tt2 <- tk2plevel()
pw <- tk2panedwindow(tt2, orient = "vertical")
lpw.1 <- tk2label(pw, text = "Panel 1")
lpw.2 <- tk2label(pw, text = "Panel 2")
tkadd(pw, lpw.1, minsize = 100)
tkadd(pw, lpw.2, minsize = 70)
but <- tk2button(lpw.2, text = "OK", width = 10,
    command = function() tkdestroy(tt2))
tkgrid(but)
tkpack(pw, fill = "both", expand = "yes")
```

# Resize the window and move the panel separator with the mouse

### A tk2combobox example

```r
tt2 <- tk2plevel()
cb <- tk2combobox(tt2)
tkgrid(cb)
```

# Fill the combobox list

```r
fruits <- c("Apple", "Orange", "Banana", "Pear")
# TO DO: change this!!!
# for (i in 1:4)
#    tk2listinsert(cb, "end", fruits[i])
    tkinsert(cb, "end", "pear")
# Set the current value for the combobox
# or better: link it to a variable
Fruit <- tclVar("Pear")
tkconfigure(cb, textvariable = Fruit)
```

# Create a button to get the content of the combobox

```r
but <- tk2button(tt2, text = "OK", width = 10,
    command = function() {tkdestroy(tt2); cat(tclvalue(Fruit), "\n")})
tkgrid(but)
```

### An example of a tk2spinbox widget

```r
tt2 <- tk2plevel()
tspin <- tk2spinbox(tt2, relief = "groove", from = 2, to = 20, increment = 2)
tkgrid(tspin)
```

#tdial <- tk2dial(tt2, from = 0, to = 20, resolution = 0.5, width = 70,
#    tickinterval = 2)

```r
but <- tk2button(tt2, text = "OK", width = 10,
    command = function() tkdestroy(tt2))
tkgrid(but)
```

### A tk2mclistbox example

```r
```
tk2ico

tt2 <- tktoplevel()
mlb <- tk2mclistbox(tt2, width = 55, resizablecolumns = TRUE)
# Define the columns
tk2column(mlb, "add", "name", label = "First name", width = 20)
tk2column(mlb, "add", "lastname", label = "Last name", width = 20)
tk2column(mlb, "add", "org", label = "Organisation", width = 15)
tkgrid(mlb)
# Fill the multicolumn list (we can use a vector, or a matrix of character strings)
item1 <- c("Bryan", "Oackley", "ChannelPoint")
items <- matrix(c("John", "Ousterhout", "Scriptics",
"Steve", "Miller", "TclTk inc."), ncol = 3, byrow = TRUE)
tk2insert.multi(mlb, "end", item1)
tk2insert.multi(mlb, "end", items)
### TO DO: bind events
# Ex: .listbox label bind date <ButtonPress-1> "sortByDate"
# See the example.tcl in \libs\mclistbox1.02 for a more complex example
# Create a button to close the dialog box
but <- tk2button(tt2, text = "OK", width = 10,
                command = function() tkdestroy(tt2))
tkgrid(but)

### A simple tk2table example
myRarray <- c("Animal", "\"sphinx moth\"", "oyster",
               "Type", "insect", "mollusk")
dim(myRarray) <- c(3, 2)
for (i in (0:2))
  for (j in (0:1))
    .Tcl(paste("set tclarray(", i, ",", j, ") ", myRarray[i+1, j+1], sep = "\"")

tt2 <- tktoplevel()
table1 <- tk2table(tt2, variable = "tclarray", rows = "3",
                   cols = "2", titlerows = "1", selectmode = "extended",
                   colwidth = "25", background = "white")
tkpack(table1)
## End(Not run)

tk2dde

Use DDE (Dynamic Data Exchange) under Windows

Description

These functions are Windows-specific. They issue an error under a different platform.

tk2ico

Manipulate icons under Windows

Description

These function are only useful for Windows, but they can be used without error on other platform
for making platform-independent code that has an additional behaviour under Windows. On the
other platforms, these function just return NULL.
Usage

```r
tk2ico.create(icofile)
tk2ico.destroy(icon)
tk2ico.hicon(icon)
tk2ico.info(icon, convert = TRUE)
tk2ico.load(file = "shell32.dll", res = "application")
tk2ico.pos(icon, pos)
tk2ico.set(win, icon, pos = NULL, type = c("all", "small", "big"))
tk2ico.text(icon, text = NULL)
tk2ico.taskbar.add(icon, pos = 0, text = tk2ico.text(icon), callback = NULL)
tk2ico.taskbar.delete(icon)
tk2ico.taskbar.modify(icon, pos = NULL, text = NULL)
```

Arguments

- **icofile**: A file with a `.ico` extension, containing one or more Windows icons
- **icon**: An icon object
- **convert**: Do we convert the result into a data.frame?
- **file**: A file having icon resources (.exe, or .dll)
- **res**: The name of the resource from where the icon should be extracted
- **pos**: A position (starting from 0) pointing to an icon in a multi-icon object
- **win**: A tk window whose icon is changed
- **type**: Do we change only the small, the large, or both icons?
- **text**: Change a text for an icon
- **callback**: Give a callback function called when the user clicks in the taskbar icon

Value

These function do nothing and return `NULL` on other platforms than Windows.

Author(s)

Philippe Grosjean

---

### Description

These functions are Windows-specific. They issue an error on other platforms.
Index

*Topic IO
  tk2dde, 10

*Topic utilities
  tclVarFun, 1
  tk2commands, 3
  tk2dde, 10
  tk2ico, 10
  tk2reg, 11
  tk2theme, 3
  tk2tip, 6
  tk2widgets, 7
  .tk2dde.require (tk2dde), 10

makeTclNames (tclVarFun), 1

tclFun (tclVarFun), 1
  tclGetValuet (tclVarFun), 1
  tclSetValuet (tclVarFun), 1
  tclVarExists (tclVarFun), 1
  tclVarFind (tclVarFun), 1
  tclVarFun, 1
  tclVarname (tclVarFun), 1
  tk2button (tk2widgets), 7
  tk2canvas (tk2widgets), 7
  tk2checkbutton (tk2widgets), 7
  tk2column (tk2commands), 3
  tk2combox (tk2widgets), 7
  tk2commands, 3, 8
  tk2dde, 10
  tk2entry (tk2widgets), 7
  tk2frame (tk2widgets), 7
  tk2ico, 10
  tk2ico.taskbar.add (tk2ico), 10
  tk2ico.taskbar.delete (tk2ico), 10
  tk2ico.taskbar.modify (tk2ico), 10
  tk2ico.text (tk2ico), 10
  tk2insert.multi (tk2commands), 3
  tk2killtip (tk2tip), 6
  tk2label (tk2widgets), 7
  tk2labelframe (tk2widgets), 7
  tk2listbox (tk2widgets), 7
  tk2listinsert (tk2commands), 3
  tk2mclistbox (tk2widgets), 7
  tk2menu (tk2widgets), 7
  tk2menubutton (tk2widgets), 7
  tk2message (tk2widgets), 7
  tk2notebook (tk2widgets), 7
  tk2notetraverse (tk2commands), 3
  tk2panedwindow (tk2widgets), 7
  tk2progress (tk2widgets), 7
  tk2radiobutton (tk2widgets), 7
  tk2reg, 11
  tk2scale (tk2widgets), 7
  tk2scrollbar (tk2widgets), 7
  tk2separator (tk2widgets), 7
  tk2spinbox (tk2widgets), 7
  tk2table (tk2widgets), 7
  tk2text (tk2widgets), 7
  tk2theme, 2, 3, 3, 6, 8
  tk2tip, 3, 4, 6, 8
  tk2tree (tk2widgets), 7
  tk2treeview (tk2widgets), 7
  tk2widgets, 3, 4, 6, 7