Package: tkImgR (via r-universe)

September 12, 2024

Type Package

Title Simple Image Viewer for R Using the 'tcltk' Package

Version 0.0.5

Description A 'Tcl/Tk' Graphical User Interface (GUI) to display images than can be zoomed and panned using the mouse and keyboard shortcuts. 'tkImgR' read and write different image formats (PPM/PGM, PNG and GIF) using the standard 'Tcl/Tk' distribution (>=8.6), but other formats (JPEG, TIFF, CR2) can be handled using the 'tkImg' package for 'Tcl/Tk'.

Depends R (>= 3.5), tcltk

Imports tkRplotR

Suggests testthat

License GPL (>= 2)

SystemRequirements Tcl/Tk (>= 8.6). To read and write other formats than PPM/PGM, PNG and GIF it is required the 'tkImg' for 'Tcl/Tk' (https://sourceforge.net/projects/tkimg/), or the debian 'libtk-img' package (Ubuntu) or the RPM 'tkimg' package (Fedora or openSUSE).

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation no

Author Filipe Campelo [aut, cre] (<https://orcid.org/0000-0001-6022-9948>)

Maintainer Filipe Campelo <fcampelo@ci.uc.pt>

Date/Publication 2022-05-13 08:10:02 UTC

Repository https://filipecampelo.r-universe.dev

RemoteUrl https://github.com/cran/tkImgR

RemoteRef HEAD

RemoteSha 57d6d68fbba922154a2b17269d8202bb3220758a

Contents

`canvasAddBinds` *Commands to zoom and pan the image using the mouse or the keyboard (or by evoking directly the function)*

Description

Functions to zoom and pan the canvas, and add the bind to the canvas.

Usage

```
canvasAddBinds(W)

canvasControlButton4(W)

canvasControlDown(W)

canvasControlUp(W)

canvasSpace(W, ...)

canvasSpaceRelease(W)

canvasMotion(W, ...)

canvasLeft(W)

canvasRight(W)

canvasUp(W)

canvasDown(W)

canvasControlRight(W)

canvasMouseWheel(W, ...)

canvasControlLeft(W)

canvasControlMouseWheel(W, ...)
```

```
canvasPlus(W, ...)

canvasMinus(W, ...)

canvasShiftButton4(W, ...)

canvasShiftButton5(W, ...)

canvasShiftMouseWheel(W, ...)
```

Arguments

W	tktoplevel object with the canvas displaying the image
...	further arguments.

Details

These functions define the keyboard and mouse controls for the toplevel window.

Value

No return value, called for side effects

Examples

```
## Not run:
file_path <- system.file("img", "example.png", package = "tkImgR")
tt <- tkImShow(file_path)
Sys.sleep(0.25)
canvasLeft(tt)
Sys.sleep(0.25)
canvasControlLeft(tt)
Sys.sleep(0.25)
canvasRight(tt)
Sys.sleep(0.25)
tcltk::tkdestroy(tt)

## End(Not run)
```

---

tkimageRead *Tk commands to deal with images*

Description

These commands create, read, copy, write, and delete images using the 'tcltk' package.

Usage

```
tkimageRead(imageName = NULL, fileName, ...)

tkimageCreate(imageName = NULL, ...)

tkimageCopy(imageName, sourceImage, ...)

tkimageWrite(imageName, fileName, ...)

tkimageDelete(imageName)
```

Arguments

imageName	Specifies the name for the image; if is NULL then Tk picks a name of the form image#, where # is an integer.
fileName	The path for the image file.
...	Further arguments.
sourceImage	The name (or the tcl object) of the image to be copied.

Value

tclObj with the image if the function is tkimageCreate, tkimageRead, and tkimageCopy or no value for tkimageWrite or tkimageDelete

Examples

```
#tkimageRead
file_path <- system.file("img", "example.png", package = "tkImgR")
im01 <- tkimageRead("tkImage01", file_path)
"tkImage01" %in% as.character(tcltk::.Tcl("image names"))
tkimageDelete(im01)


#tkimageCreate
file_path <- system.file("img", "example.png", package = "tkImgR")
im1 <- tkimageCreate("tkImage01")
tkimage.height(im1) #0
im1 <- tkimageCreate("tkImage01", file_path)
tkimage.height(im1) #2824
"tkImage01" %in% as.character(tcltk::.Tcl("image names"))
tkimageDelete(im1)


#tkimageCopy
file_path <- system.file("img", "example.png", package = "tkImgR")
im1 <- tkimageCreate("tkImage01", file_path)
im3 <- tkimageCreate("tkImage03")
tkimageCopy(im3, "tkImage01")
c("tkImage01","tkImage03") %in% as.character(tcltk::.Tcl("image names"))
tkimageDelete(im1)
```

```
tkimageDelete(im3)


#tkimageWrite
file_path <- system.file("img", "example.png", package = "tkImgR")
im1 <- tkimageCreate("tkImage01", file_path)
file_path_crop_image <- file.path(tempdir(check = TRUE), "crop.png")
#if is possible to write the file
if (file.access(file_path_crop_image)==0){
 tkimageWrite(im1, file_path_crop_image, from=c(0,1500))
 im1_crop <- tkimageRead("tkImage01_crop", file_path_crop_image)
 print(tkimage.height(im1)) #2824
 print(tkimage.height(im1_crop)) #1324 = 2824 - 1500
 tkimageDelete(im1_crop)
}


#tkimageDelete
file_path <- system.file("img", "example.png", package = "tkImgR")
im1 <- tkimageCreate("tkImage01", file_path)
 "tkImage01" %in% as.character(tcltk::.Tcl("image names"))
 tkimageDelete(im1)
  "tkImage01" %in% as.character(tcltk::.Tcl("image names"))
```

---

tkImShow *Open and Display Image in a Tk Canvas*

Description

Open and display an image in a canvas that can be zoomed and panned using the mouse and keyboard shortcuts

Usage

```
tkImShow(file, zoom = NULL, title = NULL)
```

Arguments

file	path to image file
zoom	the zoom factor (ratio), for zoom = 1 the image is shown with no zoom (original size), when zoom is < (>) than 1 the image is zoomed out (in). The default value of zoom is NULL.
title	the window title

Value

The tkwin object returned by tkImShow is a toplevel window with a canvas that contains several variables (canvasAllowZoom, canvasScrollWidth) and tkwin objects (canvas, canvasScrollHorizontal, canvasScrollVertical) placed in the env, which could be used to implement further methods.

Examples

```
file_path <- system.file("img", "example.png", package = "tkImgR")
tt <- tkImShow(file_path)

if (!identical(tcltk::tclRequire("Img", warn = FALSE),FALSE)){
file_path1 <- system.file("img", "example.jpg", package = "tkImgR")
tt <- tkImShow(file_path1)
}
```

Index