Package ‘clickableImageMap’

May 8, 2026

Version 1.0

Date 2024-05-08

Title Implement 'tableGrob' Object as a Clickable Image Map

Maintainer Barry Zeeberg <barryz2013@gmail.com>

Author Barry Zeeberg [aut, cre]

Depends R (>= 4.2.0)

LazyData true

Imports gridExtra, ggplotify, grid, ggplot2, stats, gtable, grDevices

Description Implement 'tableGrob' object as a clickable image map.
The 'clickableImageMap' package is designed to be more convenient and more configurable than the edit() function.
Limitations that I have encountered with edit() are cannot control
(1) positioning
(2) size
(3) appearance and formatting of fonts
In contrast, when the table is implemented as a 'tableGrob', all of these features are controllable.
In particular, the 'ggplot2' grid system allows exact positioning of the table relative to other graphics etc.

License GPL (>= 2)

Encoding UTF-8

VignetteBuilder knitr

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

RoxygenNote 7.3.1

Config/testthat/edition 3

NeedsCompilation no

Repository CRAN

Date/Publication 2024-05-14 07:43:17 UTC

Contents

`annunciator` *annunciator*

Description

post a message in the annunciator grob of gtab

Usage

```
annunciator(gtab, row, message, name)
```

Arguments

gtab	return value of gtable_replace_grob()
row	integer the row number of the annunciator grob in gtab
message	character string message to be posted
name	character string value of name field in gtab layout matrix

Value

returns the return value of gtable_add_grob()

Examples

```
if(interactive()){
load("data/x_rows.RData")
annunciatorRow<-which(names(x_rows)=="annunciatorRow")
load("data/x_gtab.RData")
annunciator(x_gtab,annunciatorRow,"message","annunciator")
}
```

---

calibrate *calibrate*

Description

use coordinates of upper left and bottom right of matrix to construct mapping between viewport coordinates and matrix cells

Usage

```
calibrate(m, rows, pullDownRow)
```

Arguments

m	matrix
rows	list of row heights in the gtable object
pullDownRow	integer number of the target row in the gtable object

Value

returns a list whose components are matrices representing the upper and lower coordinates of the row and column cells

Examples

```
if(interactive()){
m<-matrix(1:20 * .05,nrow=2,ncol=10)
load("data/x_rows.RData")
pullDownRow<-which(names(x_rows)=="pullDownRow")
load("data/x_m.RData")
cal<-calibrate(x_m,x_rows,pullDownRow)
}
```

`clickableImageMapDemo` *clickableImageMapDemo*

Description

demo to illustrate how to implement calibrate() and grid.locator() for a numerical matrix. This is just a stub to be replaced by the user's actual program.

Usage

```
clickableImageMapDemo(
  n = 3,
  bounds = list(xmin = 0.534, xmax = 0.553, ymin = 0.057, ymax = 0.067),
  sleepTime = 0.5
)
```

Arguments

`n`	integer number of values to be edited in matrix m
`bounds`	list of 4 numerical values xmin, xmax, ymin, ymax
`sleepTime`	numeric number of seconds to sleep to avoid potential race condition

Details

this package emulates edit() but allows full control over formatting and management of the edited matrix. sleepTime parameter can be set to nonzero (suggest trying sleepTime=0.5) in case a complicated
graphic causes a race condition evidenced by incomplete redrawing of the window. Too large a value
might cause a noticeable annoying delay in redrawing the window.

Value

returns the updated numerical matrix

Examples

```
if(interactive()){
m<-clickableImageMapDemo(2,bounds=list(xmin=.534,xmax=.553,ymin=.057,ymax=.067))
}
```

`construct_entire_gtab` *construct_entire_gtab*

Description

construct the main gtable into which grobs will be inserted

Usage

```
construct_entire_gtab(m, rows, message, clickCoord)
```

Arguments

`m`	a matrix
`rows`	numerical vector defining rows for inserting grobs into main gtable
`message`	character string message to display in annunciator grob of gtable
`clickCoord`	numerical matrix of 2 columns, each row contains x and y coords of a mouse click

Value

returns a list whose components are

- m.pullDown component m of return value of pullDown()
- cal.pullDown return value of calibrate()
- cal.m return value of calibrate()
- gtab return value of annunciator()

Examples

```
if(interactive()){
load("data/x_m.RData")
load("data/x_rows.RData")
load("data/x_clickCoord.RData")
gtab<-construct_entire_gtab(x_m,x_rows,"x_message",x_clickCoord)
}
```

decode *decode*

Description

map the screen coordinates to a cell of a matrix

Usage

```
decode(y, cal, rcnames)
```

Arguments

y	parsed return value of grid.locator()
cal	return value of calibrate()
rcnames	Boolean if TRUE matrix has row names and col names

Value

returns an integer vector of the index of a cell in a matrix or returns -1 if rcnames is TRUE and vector y is not within valid range

Examples

```
if(interactive()){
load("data/x_y.RData")
load("data/x_rcnames.RData")
load("data/x_cal2.RData")
decode(x_y,x_cal2,x_rcnames)
}
```

defineBounds *defineBounds*

Description

use mouse clicks to define bounding box

Usage

```
defineBounds()
```

Details

use in conjunction with exitClick()

Value

returns a list of numeric xmin xmax ymin ymax defining screen target for exit

Examples

```
if(interactive()){
defineBounds()
}
```

`doubleClick` *doubleClick*

Description

detect a (left) double click (without moving cursor position)

Usage

```
doubleClick(tol = 0.001)
```

Arguments

`tol`	numeric tolerance for detecting same position

Details

I realized this is not very useful, as processing is stopped until 2 clicks are detected

Value

returns TRUE if a double click was detected

Examples

```
if(interactive()){
doubleClick()
}
```

exitClick *exitClick*

Description

test position of mouse click to see if user wants to exit

Usage

```
exitClick(bounds, y)
```

Arguments

bounds	list of numeric xmin xmax ymin ymax defining screen target for exit
y	numeric vector of x and y cursor position

Details

use in conjunction with defineBounds()

Value

Boolean TRUE if y is within bounds

Examples

```
if(interactive()){
load("data/x_bounds.RData")
load("data/x_y.RData")
exitClick(x_bounds,x_y)
}
```

gtable_replace_grob *gtable_replace_grob*

Description

replace an existing grob (in a row of a gtable) with an updated version

Usage

```
gtable_replace_grob(gtab, row, new_grob, name)
```

Arguments

gtab	a gtable object
row	integer target row number within the gtable
new_grob	update grob to insert into gtable
name	character string entry in the "name" field of gtable$layout

Value

returns the updated gtable object

Examples

```
if(interactive()){
load("data/x_gtab.RData")
load("data/x_tab.RData")
load("data/x_rows.RData")
ptabRow<-which(names(x_rows)=="ptabRow")
gtab<-gtable_replace_grob(x_gtab,ptabRow,x_tab,name="ptab")
}
```

---

highlight *highlight*

Description

invoke highlight() to set highlight font color and size

Usage

```
highlight(gtab, color, fontsize)
```

Arguments

gtab	a gtable object
color	character string representing a color
fontsize	integer font size

Value

returns gtab

Examples

```
if(interactive()){
load("data/x_gtab.RData")
highlight(x_gtab,"red",16)
}
```

highlightOneCell *highlightOneCell*

Description

highlight one cell of grob matrix in gtab

Usage

```
highlightOneCell(gtab, row, col, currentPick)
```

Arguments

gtab	a gtable object
row	integer row number of cell to highlight
col	integer col number of cell to highlight
currentPick	Boolean TRUE if this is the most recently chosen cell and we are to apply special highlighting

Value

returns gtab, a gtable object

Examples

```
if(interactive()){
load("data/x_mtab.RData")
load("data/x_clickCoord.RData")
highlightOneCell(x_mtab,x_clickCoord[1,"x"],x_clickCoord[1,"y"],FALSE)
}
```

pullDown *pullDown*

Description

generate and insert a matrix, acting as a pull down menu, into a gtable object

Usage

```
pullDown(gtab, row, focus)
```

Arguments

gtab	a gtable object
row	integer target row number within the gtable
focus	Boolean if TRUE add emphasis to matrix cell

Value

returns a list whose components are the generated matrix and the gtable object

Examples

```
if(interactive()){
load("data/x_gtab.RData")
load("data/x_rows.RData")
pullDownRow<-which(names(x_rows)=="pullDownRow")
message<-"select a new value from the pull down menu: "
pd<-pullDown(x_gtab,pullDownRow,grepl("pull down",message))
}
```

---

tabify *tabify*

Description

adjust the width and height of a matrix to exactly fill the grob

Usage

```
tabify(m, focus = FALSE, clickCoord = NULL)
```

Arguments

m	a matrix
focus	Boolean if TRUE add emphasis to matrix cell
clickCoord	param for highlightOneCell()

Value

returns the grob containing the matrix

Examples

```
if(interactive()){
load("data/x_m.RData")
t<-tabify(x_m,FALSE,NULL)
}
```

unhighlight *unhighlight*

Description

invoke highlight() to set font color and size to default

Usage

```
unhighlight(gtab)
```

Arguments

gtab	a gtable object

Value

returns the return value of highlight()

Examples

```
if(interactive()){
load("data/x_gtab.RData")
unhighlight(x_gtab)
}
```

x_bounds *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_bounds)
```

x_cal.m *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_cal.m)
```

x_cal.pullDown *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_cal.pullDown)
```

x_cal2 *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_cal2)
```

x_clickCoord *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_clickCoord)
```

x_gtab *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_gtab)
```

x_l *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_l)
```

x_m *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_m)
```

x_mtab *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_mtab)
```

x_mtab2 *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_mtab2)
```

x_rcnames *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_rcnames)
```

x_rows *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_rows)
```

x_tab *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_tab)
```

x_y *clickableImageMap data sets*

Description

clickableImageMap data sets

Usage

```
data(x_y)
```

zlocator *zlocator*

Description

wrapper to perform and decode grid.locator()

Usage

```
zlocator(cal, rcnames, bounds)
```

Arguments

cal	return value of calibrate()
rcnames	parameter passed to decode()
bounds	parameter passed to exitClick()

Details

keeps looping until a valid click is detected

Value

returns the return value of decode()

Examples

```
if(interactive()){
load("data/x_cal.m.RData")
load("data/x_rcnames.RData")
load("data/x_bounds.RData")
zlocator(x_cal.m,x_rcnames,x_bounds)
}
```

Index