Package 'gridExtra'

May 8, 2026

License GPL (>= 2)

Title Miscellaneous Functions for ``Grid'' Graphics

Type Package

Description Provides a number of user-level functions to work with ``grid'' graphics, notably to arrange multiple grid-based plots on a page, and draw tables.

Version 2.3

VignetteBuilder knitr

Imports gtable, grid, grDevices, graphics, utils

Suggests ggplot2, egg, lattice, knitr, testthat

RoxygenNote 6.0.1

NeedsCompilation no

Author Baptiste Auguie [aut, cre],
Anton Antonov [ctb]

Maintainer Baptiste Auguie <baptiste.auguie@gmail.com>

Repository CRAN

Date/Publication 2017-09-09 14:12:08 UTC

Contents

gridExtra-package *Miscellaneous Functions for "Grid" Graphics*

Description

Provides a number of user-level functions to work with "grid" graphics, notably to arrange multiple grid-based plots on a page, and draw tables.

Author(s)

baptiste Auguie <baptiste.auguie@gmail.com>

References

R Graphics by Paul Murrell (Chapman & Hall/CRC, August 2005)

See Also

Grid

arrangeGrob *Arrange multiple grobs on a page*

Description

Set up a gtable layout to place multiple grobs on a page.

Usage

```
arrangeGrob(..., grobs = list(...), layout_matrix, vp = NULL,
  name = "arrange", as.table = TRUE, respect = FALSE, clip = "off",
  nrow = NULL, ncol = NULL, widths = NULL, heights = NULL, top = NULL,
  bottom = NULL, left = NULL, right = NULL, padding = unit(0.5, "line"))

grid.arrange(..., newpage = TRUE)

marrangeGrob(grobs, ..., ncol, nrow, layout_matrix = matrix(seq_len(nrow *
  ncol), nrow = nrow, ncol = ncol), top = quote(paste("page", g, "of",
  npages)))
```

Arguments

...	grobs, gtables, ggplot or trellis objects
grobs	list of grobs
layout_matrix	optional layout
vp	viewport
name	argument of gtable
as.table	logical: bottom-left to top-right (TRUE) or top-left to bottom-right (FALSE)
respect	argument of gtable
clip	argument of gtable
nrow	argument of gtable
ncol	argument of gtable
widths	argument of gtable
heights	argument of gtable
top	optional string, or grob
bottom	optional string, or grob
left	optional string, or grob
right	optional string, or grob
padding	unit of length one, margin around annotations
newpage	open a new page

Details

Using marrangeGrob, if the layout specifies both nrow and ncol, the list of grobs can be split into multiple pages. On interactive devices print opens new windows, whilst non-interactive devices such as pdf call grid.newpage() between the drawings.

Value

arrangeGrob returns a gtable.

marrangeGrob returns a list of class arrangelist

Functions

- `arrangeGrob`: return a grob without drawing
- `grid.arrange`: draw on the current device
- `marrangeGrob`: interface to arrangeGrob that can dispatch on multiple pages

Examples

```
library(grid)
grid.arrange(rectGrob(), rectGrob())
## Not run:
library(ggplot2)
pl <- lapply(1:11, function(.x) qplot(1:10, rnorm(10), main=paste("plot", .x)))
ml <- marrangeGrob(pl, nrow=2, ncol=2)
## non-interactive use, multipage pdf
ggsave("multipage.pdf", ml)
## interactive use; open new devices
ml

## End(Not run)
```

---

gtable_combine *Combine gtables based on row/column names.*

Description

Combine gtables based on row/column names.

Usage

```
gtable_combine(..., along = 1L, join = "outer")

combine(..., along = 1L, join = "outer")
```

Arguments

...	gtables
along	dimension to align along, 1 = rows, 2 = cols.
join	when x and y have different names, how should the difference be resolved? `inner` keep names that appear in both, `outer` keep names that appear in either, `left` keep names from `x`, and `right` keep names from `y`.

---

gtable_rbind *rbind gtables*

Description

rbind gtables

cbind gtables

Usage

```
gtable_rbind(..., size = "max", z = NULL)

gtable_cbind(..., size = "max", z = NULL)
```

Arguments

...	gtables
size	how should the widths be calculated? max maximum of all widths min minimum of all widths first widths/heights of first gtable last widths/heights of last gtable
z	optional z level

---

ngonGrob *Regular polygon grob*

Description

Regular polygons with optional rotation, stretching, and aesthetic attributes.

Usage

```
ngonGrob(x, y, n = 5, size = 5, phase = pi/2, angle = 0, ar = 1,
  gp = gpar(colour = "black", fill = NA, linejoin = "mitre"), ...,
  position.units = "npc", size.units = "mm")

grid.ngon(...)

ellipseGrob(x, y, size = 5, angle = pi/4, ar = 1, n = 50,
  gp = gpar(colour = "black", fill = NA, linejoin = "mitre"), ...,
  position.units = "npc", size.units = "mm")

grid.ellipse(...)

polygon_regular(n = 5, phase = 0)
```

Arguments

x	x unit
y	y unit
n	number of vertices
size	radius of circumscribing circle
phase	angle in radians of first point relative to x axis
angle	angle of polygon in radians

ar	aspect ratio
gp	gpar
...	further parameters passed to polygonGrob
position.units	default units for the positions
size.units	grid units for the sizes

Value

A grob.

Functions

- ngonGrob: return a polygon grob
- grid.ngon: draw a polygon grob on the current device
- ellipseGrob: return an ellipse grob
- grid.ellipse: draw an ellipse grob
- polygon_regular: return the x,y coordinates of a regular polygon inscribed in the unit circle

Examples

```
library(grid)
N <- 5
xy <- polygon_regular(N)*2

# draw multiple polygons
g <- ngonGrob(unit(xy[,1],"cm") + unit(0.5,"npc"),
              unit(xy[,2],"cm") + unit(0.5,"npc"),
              n = seq_len(N) + 2, gp = gpar(fill=1:N))

grid.newpage()
grid.draw(g)

# rotated and stretched
g2 <- ngonGrob(unit(xy[,1],"cm") + unit(0.5,"npc"),
              unit(xy[,2],"cm") + unit(0.5,"npc"),
              n = seq_len(N) + 2, ar = seq_len(N),
              phase = 0, angle = pi/(seq_len(N) + 2),
              size = 1:N + 5)

grid.newpage()
grid.draw(g2)

# ellipse
g3 <- ellipseGrob(unit(xy[,1],"cm") + unit(0.5,"npc"),
                  unit(xy[,2],"cm") + unit(0.5,"npc"),
                  angle = -2*seq(0,N-1)*pi/5 + pi/2,
                  size = 5, ar = 1/3)

grid.newpage()
grid.draw(g3)
```

tableGrob *Graphical display of a textual table*

Description

Create a gtable containing text grobs representing a character matrix.

Usage

```
tableGrob(d, rows = rownames(d), cols = colnames(d),
  theme = ttheme_default(), vp = NULL, ...)

grid.table(...)

ttheme_default(base_size = 12, base_colour = "black", base_family = "",
  parse = FALSE, padding = unit(c(4, 4), "mm"), ...)

ttheme_minimal(base_size = 12, base_colour = "black", base_family = "",
  parse = FALSE, padding = unit(c(4, 4), "mm"), ...)
```

Arguments

d	data.frame or matrix
rows	optional vector to specify row names
cols	optional vector to specify column names
theme	list of theme parameters
vp	optional viewport
...	further arguments to control the gtable
base_size	default font size
base_colour	default font colour
base_family	default font family
parse	logical, default behaviour for parsing text as plotmath
padding	length-2 unit vector specifying the horizontal and vertical padding of text within each cell

Value

A gtable.

Functions

- tableGrob: return a grob
- grid.table: draw a text table
- ttheme_default: default theme for text tables
- ttheme_minimal: minimalist theme for text tables

Examples

```
library(grid)
d <- head(iris, 3)
g <- tableGrob(d)
grid.newpage()
grid.draw(g)
```

Index