Package ‘customLayout’
January 17, 2020

**Type**    Package

**Title**   Arrange Elements on the R's Drawing Area or Inside the PowerPoint's Slide

**Version** 0.3.1

**Maintainer** Zygmunt Zawadzki <zygmunt@zstat.pl>

**Description** Create complicated drawing areas for multiple elements by combining much simpler layouts. It is an extended version of layout function from the 'graphics' package, but it also works with 'grid' graphics. It also supports arranging elements inside 'PowerPoint' slides created using the 'officer' package.

**License** GPL-3

**RoxygenNote** 7.0.2

**Imports** gridExtra, utils, graphics, RColorBrewer, officer (>= 0.3.6), flexible (>= 0.5.6), assertthat, rvg (>= 0.2.2)

**Suggests** covr, testthat, ggplot2, knitr, rmarkdown, vdiffr, gdtools, magrittr, dplyr, FSelectorRcpp, klaR, stringr, cowplot, png

**URL** https://www.customlayout.zstat.pl/,
        https://github.com/zzawadz/customLayout

**BugReports** https://github.com/zzawadz/customLayout/issues

**VignetteBuilder** knitr

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Zygmunt Zawadzki [aut, cre]

**Repository** CRAN

**Date/Publication** 2020-01-17 12:20:02 UTC

R topics documented:

lay_bind_col ................................................................. 2
lay_bind_row ................................................................. 3
lay_bind_col

Take two Layout objects and combine by rows.

Description

Take two Layout objects and combine by rows.

Usage

```r
lay_bind_col(x, y, widths = c(1, 1), addmax = TRUE)
```

Arguments

- `x`: object of class Layout
- `y`: object of class Layout
- `widths`: a vector with relative widths used in combining the x and y layouts.
- `addmax`: if true (default) the ids of the plots in the second layout will be shifted by the number of plots in the first layout.

Examples

```r
l1 = lay_new(matrix(c(1:2), ncol = 2), widths=c(4,1))
l2 = lay_new(matrix(c(1:4), ncol = 2), widths=c(1,1))
lb = lay_bind_col(l1, l2)
lay_show(lb)
```
**lay_bind_row**

*Take two Layout objects and combine by rows.*

**Description**

Take two Layout objects and combine by rows.

**Usage**

```r
lay_bind_row(x, y, heights = c(1, 1), addmax = TRUE)
layRowBind(x, y, heights = c(1, 1), addmax = TRUE)
```

**Arguments**

- `x`: object of class `Layout`
- `y`: object of class `Layout`
- `heights`: a vector with relative heights used in combining the x and y layouts.
- `addmax`: if true (default) the ids of the plots in the second layout will be shifted by the number of plots in the first layout.

**Examples**

```r
l1 = lay_new(matrix(c(1:2),ncol = 2),widths=c(4,1))
l2 = lay_new(matrix(c(1:4),ncol = 2),widths=c(1,1))
lb = lay_bind_row(l1,l2)
lay_show(lb)
```

---

**lay_grid**

*Use Layout object with grid graphics.*

**Description**

Use Layout object with grid graphics.

**Usage**

```r
lay_grid(grobs, lay, ...)
layGrid(grobs, lay, ...)
```

---
Arguments

- grobs: list of grobs.
- lay: a Layout object.
- ... other parameters passed to `grid.arrange`.

Examples

```r
library(ggplot2)

l1 <- lay_new(matrix(1:2, ncol = 1), heights = c(2, 3))
l2 <- lay_new(matrix(1:2, ncol = 1), heights = c(1, 3))
l3 <- lay_bind_col(l1, l2)

pl1 <- qplot(mpg, wt, data = mtcars)
pl2 <- qplot(mpg, gear, data = mtcars)
pl3 <- qplot(cyl, gear, data = mtcars)
pl4 <- qplot(qsec, am, data = mtcars)

lay_grid(list(pl1, pl2, pl3, pl4), l3)
```

lay_new

Create custom layout.

Description

Create custom layout.

Usage

```r
lay_new(mat, widths = NULL, heights = NULL)
```

Arguments

- mat: a matrix specifying the location of the figures. See `layout` for more information.
- widths: a vector of values for the relative heights of rows in mat.
- heights: a vector of values for the relative heights of rows in mat.
Examples

library(customLayout)
set.seed(123)
par(mar = c(3, 2, 2, 1))

# Prepare layout
lay <- lay_new(matrix(1:4, nc = 2),
               widths = c(3, 2),
               heights = c(2, 1))
lay2 <- lay_new(matrix(1:3))
cl <- lay_bind_col(lay, lay2, widths = c(3, 1))
lay_set(cl) # initialize drawing area

# add plots
plot(1:100 + rnorm(100))
plot(rnorm(100), type = "l")
hist(rnorm(500))
acf(rnorm(100))
pie(c(3, 4, 6), col = 2:4)
pie(c(3, 2, 7), col = 2:4 + 3)
pie(c(5, 4, 2), col = 2:4 + 6)

lay_set

Set custom layout.

Description

Set custom layout.

Usage

lay_set(layout)

laySet(layout)

Arguments

layout object of class Layout.

Examples

lplots = lay_new(matrix(1:2))
lpie = lay_new(1)
lay = lay_bind_col(lplots,lpie)
lay_set(lay)
plot(1:10)
plot(1:10)
lay_show

Print the layout structure to the graphical device.

Description
Print the layout structure to the graphical device.

Usage
lay_show(layout)

Arguments
layout an object of class Layout.

Examples
l1 <- lay_new(matrix(c(1:2), ncol = 2), widths = c(4, 1))
l2 <- lay_new(matrix(c(1:3), ncol = 3), widths = c(2, 1, 3))
l3 <- lay_bind_row(l1, l2, heights = c(2, 1))
lay_show(l3)

l4 <- lay_new(matrix(c(1:2), ncol = 2), widths = c(4, 1))
l5 <- lay_new(matrix(c(1:3), ncol = 1), heights = c(2, 1, 1))
l6 <- lay_bind_col(l4, l5, widths = c(1, 1))
lay_show(l6)

lay_split_field

Split a selected field from layout using a schema from another layout.

Description
Split a selected field from layout using a schema from another layout.

Usage
lay_split_field(lay, newlay, field)
laySplitField(lay, newlay, field)
Arguments

- lay: a Layout object.
- newlay: a Layout object used to split a field from lay.
- field: id of a field from lay.

Examples

```r
l1 <- lay_new(matrix(c(1:4), ncol = 2), widths = c(4, 1))
l2 <- lay_new(matrix(c(1:4), ncol = 2), widths = c(1, 1))
l3 <- lay_split_field(l1, l2, 2)
lay_show(l3)
```

---

### phl_adjust_table

Create flextable for layout’s placeholder.

**Description**

Create flextable from data.frame and try to fit the result into layout’s placeholder.

**Usage**

```r
phl_adjust_table(x, olay, id, method = c("all", "height"))
```

**Arguments**

- x: data.frame.
- olay: officer layout created using `phl_layout`.
- id: of placeholder in olay.
- method: if 'all' (default) fits both the width and height. If 'height' fits only height.

**Value**

A flextable object, which should fit into the layout’s placeholder.

The result should be ready to pass it into `phl_with_flextable`.

**Examples**

```r
lay <- lay_new(matrix(1:4,nc=2),widths=c(3,2),heights=c(2,1))
lay2 <- lay_new(matrix(1:3))
lay3 <- lay_bind_col(lay,lay2, widths=c(3,1))
offLayout <- phl_layout(lay3)
x <- tail(iris, 10)[,c(1,5)]
```
**Description**

Calculate optimal fontsize and height of the cell for given height for flextable.

**Usage**

```r
phl_calc_fontsize(data, height)
```

**Arguments**

- `data` data.frame.
- `height` single numeric value with desired height.

**Value**

A named numeric vector containing two elements:

- `fs` font size
- `height` of the single cell.

**Examples**

```r
x <- tail(iris, 10)[,c(1,5)]
phl_calc_fontsize(x, 5)
```

---

**Description**

Create layout for the officer PowerPoint slide.

Create layout for the officer PowerPoint slide.
Usage

```r
phl_layout(
  cl,
  slideWidth = 10,
  slideHeight = 7.5,
  margins = c(bottom = 0.25, left = 0.25, top = 0.25, right = 0.25),
  innerMargins = c(bottom = 0.025, left = 0.025, top = 0.025, right = 0.025)
)
```

Arguments

- `cl` layout object
- `slideWidth` width of the slide in inches (default 10)
- `slideHeight` height of the slide in inches (default 7.5)
- `margins` A numerical vector of the form c(bottom, left, top, right) which gives the size of margins on the four sides of the layout. The default is c(0.25, 0.25, 0.25, 0.25).
- `innerMargins` A numerical vector of the form c(bottom, left, top, right) which gives the size of margins on the four sides of the each placeholder in the layout. The default is c(0.025, 0.025, 0.025, 0.025).

Value

A list containing the coordinates of the slide segments created from layout scheme.

Examples

```r
library(officer)
library(customLayout)
library(magrittr)
library(ggplot2)

lay = lay_new(matrix(1:4,nc = 2),widths=c(3, 2),heights=c(2, 1))
lay2 = lay_new(matrix(1:3))
cl = lay_bind_col(lay,lay2, widths = c(3,1))
allPositions <- phl_layout(cl, innerMargins = rep(0.1,4))

my_pres <- read_pptx() %>%
  add_slide(master = "Office Theme", layout = "Two Content")

p <- qplot(mpg, wt, data = mtcars)

for(pos in allPositions) {
  my_pres <- my_pres %>% officer::ph_with(
    p, location = ph_location(
      width = pos["width"],
      height = pos["height"],
      left = pos["left"],
```
phl_with_flextable

add flextable into layout placeholder

Description

add flextable into layout placeholder

Usage

phl_with_flextable(x, olay, id, value)

Arguments

x  
rptx object

olay  
an OfficerLayout object created using phl_layout.

id  
an single integer with an id of the placeholder from olay object.

value  
a flextable object. Possibly the result of the phl_adjust_table

Examples

library(officer)
lay <- lay_new(matrix(1:4,nc=2),widths=c(3,2),heights=c(2,1))
lay2 <- lay_new(matrix(1:3))
lay3 <- lay_bind_col(lay,lay2, widths=c(3,1))
offLayout <- phl_layout(lay3)

pptx <- read_pptx()
pptx <- add_slide(
    pptx,
    master = "Office Theme",
    layout = "Title and Content"
)

# add table to pptx file
x <- tail(iris, 10)[,c(1,5)]
xf <- phl_adjust_table(x, offLayout, 1)
pptx <- phl_with_flextable(pptx, offLayout, 1, xf)
```r
x2 <- tail(iris, 10)[,c(1,5)]
xf2 <- phl_adjust_table(x, offLayout, 2)
pptx <- phl_with_flextable(pptx, offLayout, 2, xf2)

## Not run:
file <- tempfile(fileext = ".pptx")
print(pptx, target = file)

## End(Not run)
```

---

**phl_with_gg**  
*add ggplot into layout placeholder*

**Description**

Add ggplot into layout placeholder.

**Usage**

```r
phl_with_gg(x, olay, id, value, ...)
```

**Arguments**

- `x`: `rptx` object
- `olay`: an `OfficerLayout` object created using `phl_layout`
- `id`: an integer with an id of the placeholder from `olay` object.
- `value`: a `ggplot` object
- `...`: other arguments passed to `phl_with`

---

**phl_with_plot**  
*add plot into layout placeholder*

**Description**

Add plot into layout placeholder.

**Usage**

```r
phl_with_plot(x, olay, id, plotFnc, ...)
```
**Arguments**

- **x**: rpptx object
- **olay**: an OfficerLayout object created using `phl_layout`
- **id**: an single integer with an id of the placeholder from `olay` object.
- **plotFnc**: a function which creates a plot when called.
- **...**: other arguments passed to `png` function.

---

**Description**

*add table into layout placeholder*

**Usage**

```r
phl_with_table(x, olay, id, value, ...)
```

**Arguments**

- **x**: rpptx object
- **olay**: an OfficerLayout object created using `phl_layout`
- **id**: an single integer with an id of the placeholder from `olay` object.
- **value**: a data.frame
- **...**: other arguments passed to `ph_with`

---

**Description**

*add text into layout placeholder*

**Usage**

```r
phl_with_text(x, olay, id, str, type = "title", ...)
```

**Arguments**

- **x**: rpptx object
- **olay**: an OfficerLayout object created using `phl_layout`
- **id**: an single integer with an id of the placeholder from `olay` object.
- **str**: text to add.
- **type**: type of the text placeholder. See `ph_add_text` for more details.
- **...**: other arguments passed to `ph_add_text`. 
**phl_with_vg**

add a plot as vector graphics into layout placeholder

### Description

add a plot as vector graphics into layout placeholder

### Usage

```r
phl_with_vg(x, olay, id, code, ggobj = NULL, ...)
```

### Arguments

- **x**: rpptx object
- **olay**: an OfficerLayout object created using `phl_layout`
- **id**: an single integer with an id of the placeholder from olay object.
- **code**: plot instructions.
- **ggobj**: ggplot objet to print. Argument code will be ignored if this argument is supplied.
- **...**: other arguments passed to `dml_pptx`

---

**print.CustomLayout**

Print a CustomLayout object.

### Description

Print a CustomLayout object.

### Usage

```r
## S3 method for class 'CustomLayout'
print(x, ...)  
```

### Arguments

- **x**: object of class CustomLayout.
- **...**: optional arguments to print or plot methods. Not used here.

### See Also

`lay_new` `lay_show`
Examples

```r
lay <- lay_new(matrix(1:4,nc=2),widths=c(3,2),heights=c(2,1))
lay2 <- lay_new(matrix(1:3))
cl <- lay_bind_col(lay,lay2, widths=c(3,1))
print(cl)

c12 <- lay_bind_col(cl,cl, c(2,1))
print(c12)

c13 <- lay_bind_row(cl,cl, c(20,1))
print(c13)
```

---

**print.OfficerCustomLayout**

*Print a OfficerCustomLayout object.*

Description

Print a OfficerCustomLayout object.

Usage

```r
## S3 method for class 'OfficerCustomLayout'
print(x, ...)
```

Arguments

- `x` object of class OfficerCustomLayout
- `...` optional arguments to print or plot methods. Not used here.

See Also

- `lay_new`
- `lay_show`
- `phl_layout`

Examples

```r
lay <- lay_new(matrix(1:4,nc = 2),widths = c(3,2),heights = c(2,1))
lay2 <- lay_new(matrix(1:3))
c1 <- lay_bind_col(lay,lay2, widths=c(3,1))
ofl <- phl_layout(c1, innerMargins = rep(0.1,4))
print(ofl)
```
Index

dml_pptx, 13

ggrid.arrange, 4
lay_bind_col, 2
lay_bind_row, 3
lay_grid, 3
lay_new, 4
lay_set, 5
lay_show, 6
lay_split_field, 6
layColBind (lay_bind_col), 2
layCreate (lay_new), 4
layGrid (lay_grid), 3
layout, 4
layRowBind (lay_bind_row), 3
laySet (lay_set), 5
layShow (lay_show), 6
laySplitField (lay_split_field), 6

ph_add_text, 12
ph_with, 11, 12
phl_adjust_table, 7, 10
phl_calc_fontsize, 8
phl_layout, 7, 8, 10–13
phl_with_flextable, 7, 10
phl_with_gg, 11
phl_with_plot, 11
phl_with_table, 12
phl_with_text, 12
phl_with_vg, 13
png, 12
print.CustomLayout, 13
print.OfficerCustomLayout, 14