Package 'wheatmap'

May 10, 2016

Type Package

Title WHeatmap

Version 0.1.0

Author Wanding Zhou

Maintainer Wanding Zhou <zhouwanding@gmail.com>

Description Plot heatmap in a sequential manner.

License MIT license

LazyData TRUE

RoxygenNote 5.0.1

R topics documented:

`+.WGroup` *merge plotting objects*

Description

merge plotting objects

Usage

```
## S3 method for class 'WGroup'
group + p
```

`AddWGroup` *Add a plotting object to a group*

Description

The object to be added are in the same coordinate system as the group.

Usage

```
AddWGroup(group.obj, new.obj)
```

Arguments

`group.obj`	WGroup object to be added to
`new.obj`	plotting object to be added

Value

a WGroup object where new.obj is added.

`Beneath` *Beneath*

Description

Generate dimension beneath another object

Usage

```
Beneath(x = NULL, height = NULL, pad = 0.01, min.ratio = 0.02,
  h.aln = NULL, v.scale = NULL, v.scale.proportional = FALSE)
```

Arguments

`x`	an object with dimension
`height`	the height of the new object (when NULL set proportional to the data)
`pad`	padding between the target and current
`min.ratio`	minimum ratio of dimensions when auto-scale
`h.aln`	object for horizontal alignment (when NULL, set to x)
`v.scale`	object for vertical scaling (when NULL, set to x)
`v.scale.proportional`	when v.scale is provided, whether to make proportional to data

Value

a dimension generator beneath x

`both.cluster` *row- and column-cluster a matrix*

Description

row- and column-cluster a matrix

Usage

```
both.cluster(mat, hc.method = "ward.D2")
```

Arguments

`hc.method`	method to use in hclust
`at`	input matrix

Value

a list of clustered row, column and matrix

BottomLeftOf *Bottom left of*

Description

Place a new object to the bottom left corner of another.

Usage

```
BottomLeftOf(x = NULL, just = c("bottomright", "topright", "bottomleft",
  "topleft"), v.pad = 0, h.pad = 0)
```

Arguments

x	target object, either a name, a object or NULL which refers to the last plotting object
just	the part from the new object that should be attached to from c(bottomright, topright, bottomleft, topleft)
v.pad	vertical translational padding [0.0]
h.pad	horizontal translational padding [0.0]

Value

a WDimGenerator

BottomRightOf *Bottom right of*

Description

Place a new object to the bottom right corner of another.

Usage

```
BottomRightOf(x = NULL, just = c("bottomleft", "topleft", "bottomright",
  "topright"), v.pad = 0, h.pad = 0)
```

Arguments

x	target object, either a name, a object or NULL which refers to the last plotting object
just	the part from the new object that should be attached to from c(bottomright, topright, bottomleft, topleft)
v.pad	vertical translational padding [0.0]
h.pad	horizontal translational padding [0.0]

Value

a WDimGenerator

CalcTextBounding *Calculate Text Bounding*

Description

Calculate bounding box including texts.

Usage

```
CalcTextBounding(x, ...)
```

Arguments

x	object

Details

W.R.T lower left corner of the view port in the unit of points

CalcTextBounding.WHeatmap *Calculate Texting Bounding for WHeatmap*

Description

Calculate Texting Bounding for WHeatmap

Usage

```
## S3 method for class 'WHeatmap'
CalcTextBounding(hm, group)
```

Arguments

hm	object of class WHeatmap

Value

an object of class WDim in coordinate points

CMPar *Color Map Parameters*

Description

Create color map parameters

Usage

```
CMPar(dmin = NULL, dmax = NULL, brewer.name = NULL, brewer.n = 3,
  colorspace.name = NULL, colorspace.n = 2, cmap = NULL,
  stop.points = NULL, grey.scale = FALSE)
```

Arguments

dmin	minimum for continuous color map
dmax	maximum for continuous color map
brewer.name	palette name for RColorbrewer
brewer.n	number of stop points in RColorbrewer for continuous color map
colorspace.name	colorspace name
colorspace.n	number of stops in colorspace palettes
cmap	customized colormap name
stop.points	custome stop points
grey.scale	whether to use grey scale
cm	existing color maps

Value

an object of class CMPar

ColorMap *Constructor for ColoMap object*

Description

Create color maps

Usage

```
ColorMap(continuous = TRUE, colors = NULL, dmin = NULL, dmax = NULL,
  scaler = NULL, mapper = NULL)
```

Arguments

colors	colors for each data point
dmin	miminum in continuous color map
dmax	maximum in continuous color map
scaler	scaler function from data range to 0-1
mapper	function that maps data to color
discrete	whether colormap is discrete

Value

an object of class ColorMap

---

column.cluster *column cluster a matrix*

Description

column cluster a matrix

Usage

```
column.cluster(mat, hc.method = "ward.D2")
```

Arguments

mat	input matrix
hc.method	method to use in hclust

Value

a list of clustered row, column and matrix

FromAffine *Convert from affine coordinates to absolute coordinates*

Description

Convert from affine coordinates to absolute coordinates

Usage

```
FromAffine(dm.affine, dm.sys)
```

Arguments

dm.affine	dimension on affine coordinates (relative coordinates)
dm.sys	dimension of the affine system

Value

dimension on the same coordinate system

grid.dendrogram *Draw dendrogram under grid system*

Description

The dendrogram can be renderred. A viewport is created which contains the dendrogram.

Usage

```
grid.dendrogram(dend, facing = c("bottom", "top", "left", "right"),
  max_height = NULL, order = c("normal", "reverse"), ...)
```

Arguments

dend	a stats::dendrogram object.
facing	facing of the dendrogram.
max_height	maximum height of the dendrogram.

Details

-order should leaves of dendrogram be put in the normal order (1, ..., n) or reverse order (n, ..., 1)? -... pass to ‘grid::viewport‘ which contains the dendrogram.

This function only plots the dendrogram without adding labels. The leaves of the dendrogram locates at unit(c(0.5, 1.5, ...(n-0.5))/n, "npc").

Source

adapted from the ComplexHeatmap package authored by Zuguang Gu <z.gu@dkfz.de>

GroupCheckNameUnique *Check whether group names are unique*

Description

Check whether group names are unique

Usage

```
GroupCheckNameUnique(group.obj)
```

GroupDeepGet *Get an plotting object from a group's descendants*

Description

Get an plotting object from a group's descendants

Usage

```
GroupDeepGet(x, nm, force.unique = TRUE)
```

Arguments

x	a WGroup object
nm	name
force.unique	assume the name is unique in the descendants and get one object instead of a list

Value

if 'force.unique==FALSE' return a list. Otherwise, one plotting object.

LeftOf *LeftOf*

Description

Generate dimension to the left of another object

Usage

```
LeftOf(x = NULL, width = NULL, pad = 0.01, min.ratio = 0.02,
  v.aln = NULL, h.scale = NULL, h.scale.proportional = FALSE)
```

Arguments

x	an object with dimension
width	the width of the new object (when NULL, set proportional to data)
pad	padding between the target and current
min.ratio	minimum ratio of dimensions when auto-scale
v.aln	object for vertical alignment (when NULL, set to x)
h.scale	object for horizontal scaling (when NULL, set to x)
h.scale.proportional	when h.scale is provided, whether to make proportional to data

Value

a dimension to the left of x

ly *show layout*

Description

show layout

Usage

```
ly(x)
```

MapToContinuousColors *map data to continuous color*

Description

map data to continuous color

Usage

```
MapToContinuousColors(data, cmp = CMPar(), given.cm = NULL)
```

Arguments

data	numeric vector
cmp	an color map parameter object of class CMPar

Value

an object of ColorMap

MapToDiscreteColors *map data to discrete color*

Description

map data to discrete color

Usage

```
MapToDiscreteColors(data, cmp = CMPar(), given.cm = NULL)
```

Arguments

data	numeric vector
cmp	an color map parameter object of class CMPar

Value

an object of ColorMap

plot.WHeatmap *plot WHeatmap*

Description

plot WHeatmap

Usage

```
## S3 method for class 'WHeatmap'
plot(hm, cex = 1, layout.only = FALSE,
  stand.alone = TRUE)
```

Arguments

hm	heatmap to plot

print.WDendrogram *print a dendrogram*

Description

print a dendrogram

Usage

```
## S3 method for class 'WDendrogram'
print(dend, stand.alone = TRUE, layout.only = FALSE,
  cex = 1)
```

Arguments

dend	a dendrogram
stand.alone	whether the re-scale should occur when the plot is stand alone
layout.only	plot layout only
cex	factor to scale texts

print.WGenerator *print a WGenerator*

Description

This calls WGenerator and creates a WGroup to enclose the produced object.

Usage

```
## S3 method for class 'WGenerator'
print(xg)
```

Arguments

xg	a WGenerator object

Value

the WGroup containing the plotting object

print.WGroup *Draw WGroup*

Description

Draw WGroup

Usage

```
## S3 method for class 'WGroup'
print(group, stand.alone = TRUE, cex = 1,
  layout.only = FALSE)
```

Arguments

group	plot to display
cex	for scale fonts

print.WHeatmap *plot WHeatmap*

Description

plot WHeatmap

Usage

```
## S3 method for class 'WHeatmap'
print(hm, cex = 1, layout.only = FALSE,
  stand.alone = TRUE)
```

Arguments

hm	an object of class WHeatmap

Value

NULL

RightOf *RightOf*

Description

Generate dimension to the right of another object

Usage

```
RightOf(x = NULL, width = NULL, pad = 0.01, min.ratio = 0.02,
  v.aln = NULL, h.scale = NULL, h.scale.proportional = FALSE)
```

Arguments

x	an object with dimension
width	the width of the new object (when NULL, set proportional to data)
pad	padding between the target and current
min.ratio	minimum ratio of dimensions when auto-scale
v.aln	object for vertical alignment (when NULL, set to x)
h.scale	object for horizontal scaling (when NULL, set to x)
h.scale.proportional	when h.scale is provided, whether to make proportional to data

Value

a dimension to the right of x

row.cluster *row cluster a matrix*

Description

row cluster a matrix

Usage

```
row.cluster(mat, hc.method = "ward.D2")
```

Arguments

mat	input matrix
hc.method	method to use in hclust

Value

a list of clustered row, column and matrix

ScaleGroup *Scale group*

Description

Scale group to incorporate text on margins

Usage

```
ScaleGroup(group.obj)
```

Arguments

group.obj	group object that needs to be scaled

Value

scaled group obj

text.width *font width and scale to specified font size*

Description

font width and scale to specified font size

Usage

```
## S3 method for class 'width'
text(txt, fontsize = NULL)
```

ToAffine *Convert from absolute coordinates to affine coordinates*

Description

Convert from absolute coordinates to affine coordinates

Usage

```
ToAffine(dm, dm.sys)
```

Arguments

dm	dimension on the same coordinate system as the affine system (absolute coordinates)
dm.sys	dimension of the affine system

Value

dimension on affine coordinates (relative coordinates)

TopLeftOf *Top left of*

Description

Place a new object to the top left corner of another.

Usage

```
TopLeftOf(x = NULL, just = c("bottomright", "topright", "bottomleft",
  "topleft"), v.pad = 0, h.pad = 0)
```

Arguments

x	target object, either a name, a object or NULL which refers to the last plotting object
just	the part from the new object that should be attached to from c(bottomright, topright, bottomleft, topleft)
v.pad	vertical translational padding [0.0]
h.pad	horizontal translational padding [0.0]

Value

a WDimGenerator

TopOf *Top of*

Description

Generate dimension top of another object

Usage

```
TopOf(x = NULL, height = NULL, pad = 0.01, min.ratio = 0.02,
  h.aln = NULL, v.scale = NULL, v.scale.proportional = FALSE)
```

Arguments

x	an object with dimension
height	the height of the new object (when NULL, set to proportional to data)
pad	padding between the target and current
min.ratio	minimum ratio of dimensions when auto-scale
h.aln	object for horizontal alignment (when NULL, set to x)
v.scale	object for vertical scaling (when NULL, set to x)
v.scale.proportional	when v.scale is provided, whether to make proportional to data

Value

a dimension generator on top of x

TopRightOf *Top right of*

Description

Place a new object to the top right corner of another.

Usage

```
TopRightOf(x = NULL, just = c("bottomleft", "topleft", "bottomright",
  "topright"), v.pad = 0, h.pad = 0)
```

Arguments

x	target object, either a name, a object or NULL which refers to the last plotting object
just	the part from the new object that should be attached to from c(bottomright, topright, bottomleft, topleft)
v.pad	vertical translational padding [0.0]
h.pad	horizontal translational padding [0.0]

Value

a WDimGenerator

`WColorBarH` *WColorBarH*

Description

a horizontal color bar

Usage

```
WColorBarH(data, ...)
```

Arguments

`data`	numeric vector

Value

an object of class WColorBarH

`WColorBarV` *WColorBarV*

Description

a vertical color bar

Usage

```
WColorBarV(data, ...)
```

Arguments

`data`	numeric vector

Value

an object of class WColorBarV

WColumnBind *column bind non-overlapping objects*

Description

column bind non-overlapping objects

Usage

```
WColumnBind(..., nr = NULL, nc = NULL)
```

Arguments

...	plotting objects
nr	number of rows
nc	number of columns

Value

an object of class WDim

WCustomize *Customize an existing plot*

Description

Customize an existing plot

Usage

```
WCustomize(mar.left = NULL, mar.right = NULL, mar.top = NULL,
  mar.bottom = NULL)
```

Arguments

mar.left	left margin [0.03]
mar.right	right margin [0.03]
mar.top	top margin [0.03]
mar.bottom	bottom margin [0.03]

Value

an object of class WCustomize

Examples

```
WHeatmap(matrix(c('fred','frank','brad',
                'frank','fred','frank'), ncol=2)) +
  WLegendV(NULL, RightOf(), label.fontsize = 20) +
  WCustomize(mar.right=0.1)
```

WDendrogram *WDendrogram class*

Description

WDendrogram class

Usage

```
WDendrogram(clust = NULL, dm = WDim(0, 0, 1, 1), name = "",
  facing = c("bottom", "top", "left", "right"))
```

Arguments

clust	hclust object
dm	plotting dimension
name	name of the dendrogram plot
facing	direction of the dendrogram plot

Value

an object of class WDendrogram

WDim *class WDim*

Description

class WDim

Usage

```
WDim(left = 0, bottom = 0, width = 1, height = 1, nr = 1, nc = 1,
  column.split = NULL, row.split = NULL)
```

Arguments

left	left coordinate
bottom	bottom coordinate
width	width
height	height
column.split	a list of WDim objects for column split
row.split	a list of WDim objects for row split

WGroup *Construct a WGroup*

Description

Construct a WGroup

Usage

```
WGroup(..., name = "", group.dm = WDim(), mar = WMar(), affine = FALSE,
  nr = NULL, nc = NULL)
```

Arguments

...	plotting objects to be grouped
name	name of the group
group.dm	group dimension
affine	whether the group members are on affine coordinates already
nr	number of rows
nc	number of columns

Value

an object of class WGroup

WHeatmap *WHeatmap object*

Description

Create a heatmap

Usage

```
WHeatmap(data = NULL, dm = NULL, name = "", continuous = NULL,
  cmp = CMPar(), cm = NULL, title = NULL, title.fontsize = 12,
  title.pad = 0.005, title.side = "l", xticklabels = NULL,
  xticklabels.n = NULL, xticklabel.side = "b", xticklabel.fontsize = 12,
  xticklabel.rotat = 90, xticklabel.pad = 0.005, yticklabels = NULL,
  yticklabels.n = NULL, yticklabel.side = "l", yticklabel.fontsize = 12,
  yticklabel.pad = 0.005, alpha = 1, sub.name = NULL, gp = NULL)
```

Arguments

data	data matrix
dm	plotting dimension (a WDim or a WDimGenerator object)
name	name of the plot
continuous	whether the data should be treated as continuous or discrete
cmp	a CMPar object, for tunning color mapping parameters
cm	a given color map
xticklabels	xtick label
xticklabels.n	number of xtick labels to plot (resample for aethetics by default)

Value

one or a list of heatmaps (depends on whether dimension is split)

WLegendH *WLegendH*

Description

a horizontal legend

Usage

```
WLegendH(x = NULL, dm = NULL, name = "", n.stops = 20, n.text = 5,
  label.fontsize = 12, width = 0.1, height = 0.1, ...)
```

Arguments

x	a name or a plotting object, if NULL use the last plotting object
dm	position
name	name of the plotted legend
n.stops	number of stops in computing continuous legend
n.text	number of text labels in continuous legend
label.fontsize	label font size
width	width of each unit in plotted legend
height	height of each unit in plotted legend

Value

an object of class WLegendH

Examples

```
WHeatmap(matrix(1:4,nrow=2))+WLegendH(NULL, Beneath())
```

WLegendV *WLegendV*

Description

a vertical legend

Usage

```
WLegendV(x = NULL, dm = NULL, name = "", n.stops = 20, n.text = 5,
  label.fontsize = 12, width = 0.1, height = 0.1, ...)
```

Arguments

x	a name or a plotting object, if NULL use the last plotting object
dm	position
name	name of the plotted legend
n.stops	number of stops in computing continuous legend
n.text	number of text labels in continuous legend
label.fontsize	label font size
width	width of each unit in plotted legend
height	height of each unit in plotted legend

Value

an object of class WLegendV

Examples

```
WHeatmap(matrix(1:4,nrow=2))+WLegendV(NULL, RightOf())
```

WRowBind *row bind non-overlapping objects*

Description

row bind non-overlapping objects

Usage

```
WRowBind(..., nr = NULL, nc = NULL)
```

Arguments

...	plotting objects
nr	number of rows
nc	number of columns

Value

an object of class WDim

`[.WGroup` *subset WGroup*

Description

subset WGroup

Usage

```
## S3 method for class 'WGroup'
x[i]
```

Arguments

`i`	integer indexing element

Index