Package ‘ztable’

September 28, 2021

Title Zebra-Striped Tables in LaTeX and HTML Formats

Version 0.2.3

Description Makes zebra-striped tables (tables with alternating row colors) in LaTeX and HTML formats easily from a data.frame, matrix, lm, aov, anova, glm, coxph, nls, fitdistr, mytable and cbind.mytable objects.

Depends R (>= 3.1.2)

License GPL-2

URL https://github.com/cardiomoon/ztable

LazyData true

Encoding UTF-8

Imports stringr, magrittr, RColorBrewer, flextable, officer, rstudioapi, scales

Suggests MASS, moonBook, survival, testthat, knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.1.1

NeedsCompilation no

Author Keon-Woong Moon [aut, cre]

Maintainer Keon-Woong Moon <cardiomoon@gmail.com>

Repository CRAN

Date/Publication 2021-09-28 04:50:02 UTC

R topics documented:

.onAttach .................................................. 3
addCellColor ............................................. 4
addcgroup ............................................... 5
addColColor ............................................. 5
addFrontColor .......................................... 6
addrgroup ................................................ 6
addRowColor ........................................... 7
R topics documented:

- addSigColor .................................................. 8
- addSubColNames ................................................. 8
- align2html ..................................................... 9
- align2lines ................................................... 9
- align2nd .......................................................... 10
- alignCheck ..................................................... 10
- alignCount .................................................... 10
- caption2minipage .............................................. 11
- cgroup2df .................................................... 11
- cGroupSpan ................................................... 12
- colGroupCount ............................................... 12
- color2hex ...................................................... 13
- data2table ..................................................... 13
- define_colors .................................................. 14
- getNewAlign ................................................... 14
- getNewSpanCol ................................................ 14
- getNewSpanRow ............................................... 15
- getspanRowData .............................................. 15
- getspanRowLength ............................................ 16
- gradientColor ................................................ 16
- hlines ............................................................ 17
- isGroupCol .................................................... 17
- isspanCol ....................................................... 18
- isspanRow ...................................................... 18
- make.cell.color .............................................. 19
- make.frontcolor ............................................. 20
- makeHeatmap .................................................. 20
- make_align .................................................... 21
- myhtmlStyle ................................................... 22
- name2rgb ....................................................... 22
- normalize2 ..................................................... 23
- palette2colors ............................................... 23
- parallelTables ............................................... 24
- parallelTablesHTML ......................................... 24
- parallelTablesLatex ........................................... 25
- print.ztable .................................................. 25
- printHTMLHead ............................................... 26
- printLatexHead ............................................... 26
- printRowGroup ............................................... 26
- print_ztable .................................................. 27
- repColor ....................................................... 27
- roundDf .......................................................... 27
- spanCol .......................................................... 28
- spanColWidth .................................................. 28
- spanRow .......................................................... 29
- tableLength .................................................... 29
- totalCol ........................................................ 30
- totalLeft ....................................................... 30
Description

Functions to be called when loaded, attached, detached or unloaded

Usage

.onAttach(libname, pkgname)

Arguments

libname a character string giving the library directory where
pkgname a character string giving the name of the package.
addCellColor

Add column colors of an object of ztable

Description

Add column colors of an object of ztable

Usage

addCellColor(
  z,
  rows = NULL,
  cols = NULL,
  bg = NULL,
  color = NULL,
  condition = NULL
)

Arguments

z  An object of ztable
rows  An integer vector indicating specific rows
cols  An integer vector indicating specific columns
bg  A character vector indicating background color
color  A character vector indicating color
condition  Logical expression to select rows

Examples

## Not run:
z=ztable(head(iris))
z=addRowColor(z,c(1,3),color="platinum")
z=addColColor(z,2,color="cyan")
z=addCellColor(z,cols=c(5,4),rows=5,color="red")
z

## End(Not run)
addcgroup

Add column groups of an object of ztable

Description
Add column groups of an object of ztable

Usage
addcgroup(z, cgroup, n.cgroup, color = "black", bg = "white", top = FALSE)

Arguments
- z: An object of ztable
- cgroup: A character vector or matrix indicating names of column group. Default value is NULL
- n.cgroup: A integer vector or matrix indicating the numbers of columns included in each cgroup. Default value is NULL
- color: A character vector indicating the font color of each cells.
- bg: A character vector indicating the background color of each cells.
- top: Logical. Whether or not cgroup be placed at top.

addColColor

Add column colors of an object of ztable

Description
Add column colors of an object of ztable

Usage
addColColor(z, cols = NULL, bg = NULL, color = NULL)

Arguments
- z: An object of ztable
- cols: An integer vector indicating specific columns
- bg: A character vector indicating background color
- color: A character vector indicating color

Examples
z=ztable(head(iris))
z=addColColor(z,c(1,3),color="platinum")
z
addFrontColor  Add column colors of an object of ztable

Description
Add column colors of an object of ztable

Usage
addFrontColor(z, rows, cols, color)

Arguments
- **z**: An object of ztable
- **rows**: An integer vector indicating specific rows
- **cols**: An integer vector indicating specific columns
- **color**: A character vector indicating color

Examples
z=zttable(head(iris))
z=addFrontColor(z, rows=2:4, cols=c(2,4,6), color=c("red","green","blue"))
z

addrgroup  Add row groups of an object of ztable

Description
Add row groups of an object of ztable

Usage
addrgroup(
  z,
  rgroup,
  n.rgroup,
  cspan.rgroup = NULL,
  color = "black",
  bg = "white"
)
addRowColor

Arguments

z An object of ztable
rgroup A character vector indicating names of row group. Default value is NULL
n.rgroup A integer vector indicating the numbers of rows included in each rgroup Default value is NULL
cspan.rgroup An integer indicating the column span of rgroup
color A character vector indicating the font color of rgroup.
bg A character vector indicating the background color of rgroup.

Description

Add row colors of an object of ztable

Usage

addRowColor(z, rows = NULL, bg = NULL, color = NULL, condition = NULL)

Arguments

z An object of ztable
rows An integer vector indicating specific rows
bg A character vector indicating background color
color A character vector indicating color
condition Logical expression to select rows

Examples

z=ztable(head(iris))
z=addRowColor(z,c(1,3),color="platinum")
z
addSubColNames

Description

Add a adjunctive name below column name in a ztable

Usage

addSubColNames(z, subcolnames)

Arguments

z An object of ztable
subcolnames A character vector

addSigColor

Add row color or cellcolor for rows or cells of p-value less than sigp in a ztable

Description

Add row color or cellcolor for rows or cells of p-value less than sigp in a ztable

Usage

addSigColor(z, level = 0.05, bg = "lightcyan", color = "black")

Arguments

z An object of ztable
level A p-value
bg A character indicating background color
color A character indicating color
**align2html**  
*Convert the align in Latex format to html format*

**Description**  
Convert the align in Latex format to html format

**Usage**  
`align2html(align)`

**Arguments**  
- **align**  
  A character of align in Latex format

**align2lines**  
*count the vertical column lines from align of Latex format*

**Description**  
Count the vertical column lines from align of Latex format

**Usage**  
`align2lines(align)`

**Arguments**  
- **align**  
  A string of align Latex format

**Value**  
a numeric vector consists of vertical lines of each column
align2nd  
\textit{Delete first components of align}

\textbf{Description}

Delete first components of align

\textbf{Usage}

align2nd(align)

\textbf{Arguments}

- \texttt{align} \quad A character for define the align of column in Latex format

alignCheck  
\textit{Check the validity of align}

\textbf{Description}

Check the validity of align

\textbf{Usage}

alignCheck(align, ncount, addrow)

\textbf{Arguments}

- \texttt{align} \quad A character for define the align of column in Latex format
- \texttt{ncount} \quad An integer equals of ncol function
- \texttt{addrow} \quad An integer

alignCount  
\textit{Count the number of align}

\textbf{Description}

Count the number of align

\textbf{Usage}

alignCount(align)

\textbf{Arguments}

- \texttt{align} \quad A character for define the align of column in Latex format
caption2minipage

Convert long caption to minipage

Description

Convert long caption to minipage

Usage

caption2minipage(z, caption)

Arguments

z An object of ztable
caption A character vector to convert

cgroup2df

Convert cgroup of ztable into data.frame

Description

Convert cgroup of ztable into data.frame

Usage

cgroup2df(z)

Arguments

z An object of ztable

Value

A data.frame
### cGroupSpan

**Count the colspan of each colgroup**

**Description**

Count the colspan of each colgroup

**Usage**

cGroupSpan(z)

**Arguments**

- **z**: An object of ztable

**Value**

A matrix indicating the column span occupied by each colgroup

---

### colGroupCount

**Count the colgroup of an object of ztable**

**Description**

Count the colgroup of an object of ztable

**Usage**

colGroupCount(z)

**Arguments**

- **z**: An object of class ztable

**Value**

A vector indicating the position of colgroup
color2hex

**Description**

Convert a named color into a hexadecimal color with rgb value

**Usage**

color2hex(color)

**Arguments**

color  A named color

**Value**

A hexadecimal color

**Examples**

color2hex("green")
color2hex("red")

data2table

**Description**

Convert data to formatted data for table

**Usage**

data2table(z)

**Arguments**

z  An object of class "ztable"
**define_colors**  
*Define colors*

**Description**  
Define colors of mycolors

**Usage**  
```r  
define_colors(mycolors, no = 1)  
```

**Arguments**

- `mycolors` characters vectors of color names
- `no` An integer indicating start number

---

**getNewAlign**  
*Make a character string indicating the alignment of components of table.*

**Description**  
Make a character string indicating the alignment of components of table.

**Usage**  
```r  
getNewAlign(z)  
```

**Arguments**

- `z` An object of ztable

---

**getNewSpanCol**  
*Calculating new spanCol with spanCol plus space made by column group*

**Description**  
Calculating new spanCol with spanCol plus space made by column group

**Usage**  
```r  
getNewSpanCol(z)  
```

**Arguments**

- `z` An object of ztable
getNewSpanRow

**Calculating new spanRow with spanRow plus space made by row group**

**Description**
Calculating new spanRow with spanRow plus space made by row group

**Usage**
```r
getNewSpanRow(z)
```

**Arguments**
- `z`: An object of ztable

getspanRowData

**Gets the spanRow start column**

**Description**
Gets the spanRow start column

**Usage**
```r
getspanRowData(z, i, j)
```

**Arguments**
- `z`: An object of ztable
- `i`: An integer indicating the row of specific cell
- `j`: An integer indicating the column of specific cell

**Value**
An integer indicating column where spanRow start. This function is for latex multirow
getspanRowLength  

**Description**

Gets spanRow length

**Usage**

`getspanRowLength(z, i, j)`

**Arguments**

- `z`  
  An object of ztable
- `i`  
  An integer indicating the row of specific cell
- `j`  
  An integer indicating the column of specific cell

**Value**

row count when spanRow starts, 0 when column spans.

---

gradientColor  

**Description**

Make Sequential colour gradient palette

**Usage**

`gradientColor(high = "red", low = "white", mid = NULL, n = 20, plot = FALSE)`

**Arguments**

- `high`  
  colour for high end of gradient.
- `low`  
  colour for low end of gradient.
- `mid`  
  colour for middle of gradient.
- `n`  
  the number of colors in palette
- `plot`  
  Logical. Whether or not draw plot
### hlines

Add or delete horizontal lines in a ztable

**Description**

Add or delete horizontal lines in a ztable

**Usage**

```
hlines(z, type = NULL, add = NULL, del = NULL)
```

**Arguments**

- **z**: An object of ztable
- **type**: An integer or one of c("none","all")
- **add**: An integer vector indicating rows where the horizontal lines added
- **del**: An integer vector indicating rows where the horizontal lines deleted

### isGroupCol

Returns whether or not column with position start plus length is group column

**Description**

Returns whether or not column with position start plus length is group column

**Usage**

```
isGroupCol(start, length, colCount)
```

**Arguments**

- **start**: An integer indicating start column position
- **length**: An integer indicating spanCol length
- **colCount**: An integer vector calculating from colGroupCount()
isspanCol

**Description**
Identify the spanCol status of a cell

**Usage**
isspanCol(z, i, j)

**Arguments**
- `z`: An object of ztable
- `i`: An integer indicating the row of specific cell
- `j`: An integer indicating the column of specific cell

**Value**
column plus space count when spanCol starts, 0 when column spans, minus value when spanCol ends, NULL when no span.

isspanRow

**Description**
Identify the spanRow status of a cell

**Usage**
isspanRow(z, i, j)

**Arguments**
- `z`: An object of ztable
- `i`: An integer indicating the row of specific cell
- `j`: An integer indicating the column of specific cell

**Value**
columns count plus spaces by rgroup when spanRow starts, 0 when row spans, minus value when spanRow ends, NULL when no span.
make.cell.color

Description

Make a data.frame named "cellcolor" from ztable call

Usage

make.cell.color(
  x,
  zebra,
  zebra.color,
  zebra.type,
  zebra.list,
  zebra.colnames,
  zebra.rownames
)

Arguments

x a data.frame
zebra Null or an integer of 0 or 1 or 2. The arguments zebra and zebra.color are used to make a Zebra striping table (table with alternating background colors) easily. A value of 1 sets background color of all odd rows/columns with specified with zebra.color. A value of 2 sets all even rows/columns. A value of 0 sets background colors of all rows/columns with colors specified with zebra.color. When zebra is 1 or 2, the parameters of prefix.rows and commands ignored. Default is NULL.

zebra.color A color name or a numeric value indicating pre-defined color. When parameter zebra is 0 or 1 or 2 and zebra.color is NULL, then zebra.color is set to "platinnum". Numeric values between 1 to 13 is converted to predefined color names. The predefined color names are c("peach", "peach-orange", "peachpuff", "peach-yellow", "pear", "pearl", "peridot", "periwinkle", "pastelred", "pastelgray"). Default is NULL.

zebra.type An integer of 0 or 1 or 2 or 3. A value of 1 sets background colors by row. A value of 2 sets background colors by column. A value of 0 sets background colors of all cells. A value of 3 sets background colors of cells specified with zebra.list. Default value is 1.

zebra.list A list consists of y,x,color. zebra.list is used only when zebra.type=3. zebra.list sets the cells specified with rows of vector "y" and columns of vector "x" with "color". The y and x are integer vector indicating rows or columns. NA value of y or x indicating all columns or rows. The color is an character vector consists of names of color.

zebra.colnames whether or not use background colors in column names row. Default value is FALSE
zebra.rownames whether or not use background colors in row names column, Default value is TRUE

make.frontcolor Make a data.frame named "cellcolor" from ztable call

Description
Make a data.frame named "cellcolor" from ztable call

Usage
make.frontcolor(x, color = "black")

Arguments
x A data.frame
color A character string

makeHeatmap Add gradient background color to ztable

Description
Add gradient background color to ztable

Usage
makeHeatmap(
  z,
  palette = "Reds",
  mycolor = NULL,
  rows = NULL,
  cols = NULL,
  changeColor = TRUE,
  reverse = FALSE,
  margin = 0
)

make Align

Arguments

z An object of class ztable
palette Name of color palette
mycolor user defined color vectors
rows columns to make heatmap
cols columns to make heatmap
changeColor Logical. Whether of not change font color automatically
reverse If true, reverse the font color
margin An integer. Choices are one of 0, 1 and 2. 0 (default), heatmap for all numeric data. 1: rowwise heatmap, 2: columnwise heatmap.

Examples

require(magrittr)
ztable(head(mtcars)) %>% makeHeatmap()
## Not run:
ztable(head(mtcars)) %>% makeHeatmap(palette="YlOrRd", cols=c(1,4,6), margin=2)
ztable(head(mtcars)) %>% makeHeatmap(rows=c(1,3,5), margin=1)
require(moonBook)
x=table(acs$smoking,acs$Dx)
ztable(x) %>% makeHeatmap
ztable(x) %>% makeHeatmap(palette="Blues")
ztable(x) %>% makeHeatmap(mycolor=gradientColor(low="yellow",mid="orange",high="red"))
## End(Not run)

make_align

Make align for an object of class ztable.mytable

Description

Make align for an object of class ztable.mytable

Usage

make_align(z)

Arguments

z An object of class ztable.mytable
myhtmlStyle  

**Description**

print html style

**Usage**

myhtmlStyle(z)

**Arguments**

z  
An object of ztable

name2rgb  

**Description**

Find rgb value from color name

**Usage**

name2rgb(name)

**Arguments**

name  
a valid color name

**Value**

rgb value
**normalize2**

*Convert numeric vector min to 0, max to maxvalue*

**Description**

Convert numeric vector min to 0, max to maxvalue

**Usage**

`normalize2(x, maxvalue = 10)`

**Arguments**

- `x` A vector
- `maxvalue` maximal value

**palette2colors**

*Extract hexadecimal colors from a color palette*

**Description**

Extract hexadecimal colors from a color palette

**Usage**

`palette2colors(name, reverse = FALSE)`

**Arguments**

- `name` The name of color palette from RColorBrewer package
- `reverse` Whether or not reverse the order of colors

**Value**

hexadecimal colors

**Examples**

```r
require(RColorBrewer)
require(magrittr)
palette2colors("Reds")
ztable(head(mtcars,10)) %>%
  addColColor(cols=1:12,bg=palette2colors("Set3"))
```
**parallelTables**  
*Place two or more ztables or figures side by side in Latex or HTML format*

**Description**

Place two or more ztables or figures side by side in Latex or HTML format. Requires Latex "boxed-minipage" package in preamble. The ztable for this purpose should be made by function ztable with tabular="TRUE".

**Usage**

```r
parallelTables(width, listTables, type = "latex")
```

**Arguments**

- `width` a numeric vector specifies the width to which the tables or figures should be scaled  
- `listTables` a list consists of object of "ztable" or valid figure name  
- `type` Type of table to produce. Possible values for type are "latex" or "html". Default value is "latex".

**Examples**

```r
require(ztable)
z=ztable(head(mtcars[1:3]),tabular=TRUE)
parallelTables(c(0.4,0.3),list(z,z))
parallelTables(c(0.5,0.5),list(z,z))
parallelTables(c(0.5,0.5),list(z,z,type="html"))
z1=ztable(head(iris[1:3]),turn=TRUE,angle=10,zebra=1)
z2=ztable(head(iris[1:3]),turn=TRUE,angle=-10,zebra=2)
parallelTables(c(0.5,0.5),list(z1,z2))
```

---

**parallelTablesHTML**  
*Place two or more ztables or figures side by side in HTML format*

**Description**

Place two or more ztables or figures side by side in HTML format. The ztable for this purpose should be made by function ztable with tabular="TRUE".

**Usage**

```r
parallelTablesHTML(width, listTables)
```
parallelTablesLatex

Arguments

width a numeric vector specifies the width to which the tables or figures should be scaled
listTables a list consists of object of "ztable" or valid figure name

parallelTablesLatex Place two or more ztables or figures side by side in Latex format

Description

Place two or more ztables or figures side by side in HTML format. The ztable for this purpose should be made by function ztable with tabular="TRUE".

Usage

parallelTablesLatex(width, listTables)

Arguments

width a numeric vector specifies the width to which the tables or figures should be scaled
listTables a list consists of object of "ztable" or valid figure name

print.ztable Print an object of class "ztable"

Description

Print an object of class "ztable"

Usage

## S3 method for class 'ztable'
print(x, ...)

Arguments

x An object of class "ztable"
... further argument passed to other function
### printHTMLHead

*Print HTML head if ztable object `a` has a colgroup*

**Description**

Print HTML head if ztable object `a` has a colgroup

**Usage**

```r
printHTMLHead(z)
```

**Arguments**

- `z` An object of `ztable`

### printLatexHead

*Print the head of latex table if the object of ztable has a colgroup*

**Description**

Print the head of latex table if the object of ztable has a colgroup

**Usage**

```r
printLatexHead(z)
```

**Arguments**

- `z` An object of `ztable`

### printRowGroup

*Print Row Groups in a latex table*

**Description**

Print Row Groups in a latex table

**Usage**

```r
printRowGroup(z, i)
```

**Arguments**

- `z` An object of class `ztable`
- `i` An integer indicating row
print_ztable

Print an object of class "ztable"

Description
Print an object of class "ztable"

Usage
print_ztable(z)

Arguments
z An object of class "ztable"

repColor
Make vector x from vector color

Description
Internal function of make.cell.color

Usage
repColor(x, color)

Arguments
x A destination vector
color A character vector consists of color names

roundDf
Round the numbers of a data.frame

Description
Round the numbers of a data.frame

Usage
roundDf(df, digits = 2)
Arguments

- \texttt{df} \quad A \texttt{data.frame}
- \texttt{digits} \quad A vector of integer indicating the number of decimal places

Value

- a rounded data.frame

spanCol \quad \textit{Merging data cells of ztable object in columns}

Description

Merging data cells of ztable object in columns

Usage

\begin{verbatim}
spanCol(z, row, from, to, bg = NULL, color = NULL)
\end{verbatim}

Arguments

- \texttt{z} \quad An object of ztable
- \texttt{row} \quad An integer indicating the row of merging data cell
- \texttt{from} \quad An integer indicating start column of merging data cell
- \texttt{to} \quad An integer indicating end column of merging data cell
- \texttt{bg} \quad An optional character indicating the background color of merging cell
- \texttt{color} \quad An optional character indicating the font color of merging cell

spanColWidth \quad \textit{Calculate the spanColWidth when spanCol start}

Description

Calculate the spanColWidth when spanCol start

Usage

\begin{verbatim}
spanColWidth(z, i, j)
\end{verbatim}

Arguments

- \texttt{z} \quad An object of ztable
- \texttt{i} \quad An integer indicating the row of specific cell
- \texttt{j} \quad An integer indicating the column of specific cell
spanRow

**Value**

column count when spanCol start

---

spanRow

*Merging data cells of ztable object in rows*

---

**Description**

Merging data cells of ztable object in rows

**Usage**

spanRow(z, col, from, to, bg = NULL, color = NULL)

**Arguments**

- **z**
  - An object of ztable
- **col**
  - An integer indicating the column of merging data cell
- **from**
  - An integer indicating start row of merging data cell
- **to**
  - An integer indicating end row of merging data cell
- **bg**
  - An optional character indicating the background color of merging cell
- **color**
  - An optional character indicating the font color of merging cell

---

tableLength

*Convert data to formatted data for table*

---

**Description**

Convert data to formatted data for table

**Usage**

tableLength(z)

**Arguments**

- **z**
  - An object of class "ztable"
### totalCol

*Calculating total columns of ztable*

**Description**

Calculating total columns of ztable

**Usage**

```r
totalCol(z)
```

**Arguments**

- `z`  
  An object of ztable

### totalLeft

*Arrange total column to the left*

**Description**

Arrange total column to the left

**Usage**

```r
totalLeft(z)
```

**Arguments**

- `z`  
  An object of class ztable.mytable or ztable.cbind.mytable

**Examples**

```r
require(moonBook)
require(ztable)
require(magrittr)
mytable(sex~.,data=acs,show.total=TRUE) %>% ztable() %>% totalLeft()
## Not run:
mytable(sex+Dx~.,data=acs,show.total=TRUE) %>% ztable %>% totalLeft
## End(Not run)
```
**tr**  
*Subfunction used in ztable2latex*

**Description**
Subfunction used in ztable2latex

**Usage**
tr(string)

**Arguments**
- string: a character vector

---

**tr2**  
*Subfunction used in ztable2html*

**Description**
Subfunction used in ztable2html

**Usage**
tr2(string)

**Arguments**
- string: a character vector

---

**trim.ztable**  
*Make align and edit p value column for an object of class ztable.mytable*

**Description**
Make align and edit p value column for an object of class ztable.mytable

**Usage**
trim.ztable(z)

**Arguments**
- z: An object of class ztable.mytable
Description

Update options of ztable before print

Usage

update_ztable(
  z,
  family = NULL,
  size = NULL,
  color = NULL,
  tablewidth = NULL,
  type = NULL,
  include.rownames = NULL,
  placement = NULL,
  position = NULL,
  show.heading = NULL,
  show.footer = NULL,
  caption = NULL,
  caption.placement = NULL,
  caption.position = NULL,
  caption.bold = NULL,
  align = NULL,
  digits = NULL,
  display = NULL,
  sidewaystable = NULL,
  longtable = NULL,
  rotate = NULL,
  turn = NULL,
  angle = NULL,
  wraptable = NULL,
  wraptablewidth = NULL,
  tabular = NULL,
  label = NULL,
  hline.after = NULL,
  booktabs = NULL,
  prefix.rows = NULL,
  commands = NULL,
  top.command = NULL,
  zebra = NULL,
  zebra.color = NULL,
  zebra.type = NULL,
  zebra.list = NULL,
  zebra.colnames = NULL,
Arguments

z
An object of class "ztable"

family
Font family. Default value is NULL. Possible value is one of the c("serif","times").

size
An integer from 1 to 10 indicating font size= c("tiny","scriptsize","footnotesize","small"," normalsize","large","Large","LARGE","huge","Huge") respectively.

color
A character indicating color of ztable

tablewidth
A numeric indicating desired table width as a ratio to linewidth. Default value is 0.3.

type
character indicating formats of ztable, either "html" or "latex".

include.rownames
A logical value whether or not include rownames in the table

placement
The table will have placement given by placement where placement must be NULL or contain only elements of "h","t","b","p","!","H".

position
The table will be have placed at the center of the paper if position is "center" or "c", and at the left side of the paper if it equals "left" or "l", and at the right side of the paper if it equals "right" or "r". The position is translated to specific latex environments such as "flushright" or "flushleft" or "center" (provided as a character vector) will enclose the tabular environment.

show.heading
A logical value whether or not include headings in the table.

show.footer
A logical value whether or not include headings in the table.

caption
A character

caption.placement
The caption will be have placed at the top of the table if caption.placement is "top" and at the bottom of the table if it equals "bottom".

caption.position
The caption will be have placed at the center of the table if caption.position is "center" or "c", and at the left side of the table if it equals "left" or "l", and at the right side of the table if it equals "right" or "r".

caption.bold
whether or not use bold font for caption

align
Character vector : nchar equal to the number of columns of the resulting table indicating the alignment of the corresponding columns.

digits
Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table.
display  | Character vector of length equal to the number of columns of the resulting table indicating the format for the corresponding columns. Since the row names are printed in the first column, the length of display is one greater than ncol(x) if x is a data.frame. These values are passed to the formatC function. Use "d" (for integers), "f", "e", "E", "g", "G", "fg" (for reals), or "s" (for strings). "f" gives numbers in the usual xxx.xxx format; "e" and "E" give n.ddde+nn or n.dddE+nn (scientific format); "g" and "G" put x[i] into scientific format only if it saves space to do so. "fg" uses fixed format as "f", but digits as number of significant digits. Note that this can lead to quite long result strings.

sidewastable  | Logical value whether or not set the tabular environment= "sidewaystable". Requires Latex "rotating" package in preamble.

longtable | Logical value whether or not set the tabular environment= "longtable". Requires Latex "longtable" package in preamble.

rotate | Logical value whether or not set the tabular environment= "rotate". No special arrangement is made to find space for the result. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle(counterclockwise).

turn | Logical value whether or not set the tabular environment= "turn". In this environment, Latex leaves space for the rotated table. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle.

angle | An integer indicate the angle to rotate(degree); range -180 to 180.

wrapstable  | Logical value whether or not set the tabular environment= "wrapstable". Requires Latex "wrapfig" package in preamble.

wrapstablewidth | A integer indicate wrapstable width in centimeter.

tabular | Logical value whether or not set the tabular environment. If TRUE, no tabular environment is set.

label | Character vector of length 1 containing the LaTeX label or HTML anchor. Set to NULL to suppress the label.

hline.after | A vector of numbers between -1 and "nrow(x)", inclusive, indicating the rows after which a horizontal line should appear. If NULL is used no lines are produced. Default value is c(-1,0,nrow(x)) which means draw a line before and after the columns names and at the end of the table. Repeated values are allowed.

booktabs | Logical value. If TRUE, the toprule, midrule and bottomrule tags from the Latex "booktabs" package are used rather than hline for the horizontal line tags. Requires Latex "booktabs" package in preamble.

prefix.rows | A numeric vector contains the position of rows on which extra Latex commands should be added as a prefix.

commands | A character vector of the length 1 or same length of the nrow of data.frame which contains the command that should be added as a prefix at the specified rows.

top.command | A character vector of the length 1 which contains the command that should be added as a prefix at the colnames row.

zebra | Null or a integer of 1 or 2. The arguments zebra and zebra.color are used to make a Zebra striping table(table with alternating background colors) easily. A value of 1 sets background color of all odd rows with specified with zebra.color. A
value of 2 sets all even rows. when zebra is 1 or 2, the parameters of prefix.rows and commands ignored.

zebra.color  A color name or a numeric value indicating pre-defined color. When parameter zebra is 0 or 1 or 2 and zebra.color is NULL, then zebra.color is set to "platinum". Numeric values between 1 to 13 is converted to predefined color names. The predefined color names are c("peach","peach-orange","peachpuff","peachyellow","pear","pearl","peridot","periwinkle","pastelred","pastelgray").

zebra.type  An integer of 0 or 1 or 2 or 3. A value of 1 sets background colors by row. A value of 2 sets background colors by column. A value of 0 sets background colors of all cells. A value of 3 sets background colors of cells specified with zebra.list. Default value is 1.

zebra.list  A list consists of y,x,color. zebra.list is used only when zebra.type=3. zebra.list sets the cells specified with cells[y,x] with "color". The y and x are integer indicating rows and columns. NA value of y or x indicating all columns or rows.

zebra.colnames  whether or not use background colors in column names row, Default value is FALSE

zebra.rownames  whether or not use background colors in row names column, Default value is TRUE

colnames.bold  whether or not use bold font for column names.

include.colnames  Logical. If TRUE the column names is printed.

cgroup  A character vector or matrix indicating names of column group. Default value is NULL

n.cgroup  A integer vector or matrix indicating the numbers of columns included in each cgroup. Default value is NULL

rgroup  A character vector indicating names of row group. Default value is NULL

n.rgroup  A integer vector indicating the numbers of rows included in each rgroup. Default value is NULL

cspan.rgroup  The number of columns that an rgroup should span. It spans by default all columns but you may want to limit this if you have column colors that you want to retain.

col  number of column displaying p value

validColor  Find valid color name

Description

Find valid color name

Usage

validColor(a, mycolor)
Arguments
a An integer or a character
mycolor predefined color names

Value
a valid Latex color name

validColor2 Find valid color name

Description
Find valid color name

Usage
validColor2(a)

Arguments
a An integer or a character

Value
a valid Latex color name

vline2align Make a latex "align" from a string and vertical line specifier

Description
Make a latex "align" from a string and vertical line specifier

Usage
vline2align(align, vlines)

Arguments
align A character string indicating align of latex table
vlines An integer vector indicating vertical line position
Add or delete vertical lines in a ztable

Usage

vlines(z, type = NULL, add = NULL, del = NULL)

Arguments

z An object of ztable
type An integer or one of c("none","all")
add An integer vector indicating columns where the width of vertical lines added
del An integer vector indicating columns where the width of vertical lines subtracted

zcolors Definition of Latex Color

Description

A dataset containing the name of color and Hex-triplets and latex definition

Usage

zcolors

Format

A data frame with 749 rows and 3 variables:

name Color name
rgb Hex triplet of color
definition Latex command of color definition

Details

To use this color definition, a latex package "color" should be included in your preamble.
Description

Make ztable from object cbind.mytable

Usage

```r
## S3 method for class 'cbind.mytable'
ztable(x, digits = NULL, ...)
```

Arguments

- `x`: An object of cbind.mytable
- `digits`: Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table
- `...`: arguments to be passed to `ztable_sub`

Examples

```r
require(moonBook)
res=mytable(sex+DM~., data=acs)
z=ztable(res)
z
```

Description

Make ztable from object mytable

Usage

```r
## S3 method for class 'mytable'
ztable(x, digits = NULL, ...)
```

Arguments

- `x`: An object of mytable
- `digits`: Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table
- `...`: arguments to be passed to `ztable_sub`
Examples

```r
require(moonBook)
res=mytable(sex~.,data=acs)
z=ztable(res)
z
```

Exporting a R object to an object of class "ztable"

Description

Exporting a R object to an object of class "ztable"

Usage

```r
## S3 method for class 'table'
ztable(x, digits = NULL, ...)

## Default S3 method:
ztable(x, digits = NULL, ...)

## S3 method for class 'data.frame'
ztable(x, digits = NULL, ...)

## S3 method for class 'matrix'
ztable(x, digits = NULL, ...)

## S3 method for class 'lm'
ztable(x, digits = NULL, ...)

## S3 method for class 'fitdistr'
ztable(x, digits = NULL, ...)

## S3 method for class 'nls'
ztable(x, digits = NULL, ...)

## S3 method for class 'aov'
ztable(x, digits = NULL, ...)

## S3 method for class 'anova'
ztable(x, digits = NULL, ...)

## S3 method for class 'glm'
ztable(x, digits = NULL, ...)
```
## S3 method for class 'coxph'
ztable(x, digits = NULL, ...)

## S3 method for class 'prcomp'
ztable(x, digits = NULL, ...)

## S3 method for class 'summary.prcomp'
ztable(x, digits = NULL, ...)

### Arguments

**x**
An R object, mainly data.frame

digits
Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table

... arguments to be passed to \texttt{ztable\_sub}

### Methods (by class)

- **table**: Makes a ztable for class table
- **default**: Default method of ztable
- **data.frame**: Makes a ztable for class 'data.frame'
- **matrix**: Makes a ztable for class matrix
- **lm**: Makes a ztable for class 'lm'
- **fitdistr**: Makes a ztable for class 'fitdistr'
- **nls**: Makes a ztable for class 'nls'
- **aov**: Makes a ztable for class 'aov'
- **anova**: Makes a ztable for class 'anova'
- **glm**: Makes a ztable for class 'glm'
- **coxph**: Makes a ztable for class 'coxph'
- **prcomp**: Makes a ztable for class 'prcomp'
- **summary.prcomp**: Makes a ztable for class 'summary.prcomp'

---

\texttt{ztable2flextable} \hspace{1cm} \textit{Convert an object of ztable into an object of flextable}

### Description

Convert an object of ztable into an object of flextable

### Usage

\texttt{ztable2flextable(z)}
ztable2html

Arguments
z An object of class ztable

Value
An object of class flextable

Examples
z=ztable(head(mtcars))
ztable2flextable(z)

ztable2html  Print an object of class "ztable" to html table

Description
Print an object of class "ztable" to html table

Usage
ztable2html(z, xdata)

Arguments
z  An object of class "ztable"
xdata  A formatted data.frame

ztable2latex  Print an object of class "ztable" to Latex table

Description
Print an object of class "ztable" to Latex table

Usage
ztable2latex(z, xdata)

Arguments
z  An object of class "ztable"
xdata  A formatted data.frame
ztable2viewer

Print an object of ztable via rstudioapi::viewer

Description

Print an object of ztable via rstudioapi::viewer

Usage

ztable2viewer(z)

Arguments

z An object of ztable

ztable_sub

Exporting "data.frame" to an object of class "ztable"

Description

Exporting "data.frame" to an object of class "ztable"

Usage

ztable_sub(
x,
  family = NULL,
  size = 5,
  color = getOption("ztable.color", "black"),
  tablewidth = 0.3,
  type = getOption("ztable.type", "latex"),
  include.rownames = getOption("ztable.include.rownames", TRUE),
  placement = "!hbt",
  position = "c",
  show.heading = getOption("ztable.show.heading", TRUE),
  show.footer = getOption("ztable.show.footer", TRUE),
  caption = NULL,
  caption.placement = getOption("ztable.caption.placement", "top"),
  caption.position = getOption("ztable.caption.position", "c"),
  caption.bold = getOption("ztable.caption.bold", FALSE),
  align = NULL,
  digits = NULL,
  display = NULL,
  sidewaystable = FALSE,
  longtable = FALSE,
Arguments

x A data.frame
family Font family. Default value is NULL. Possible value is one of the c("serif","times").
size An integer from 1 to 10 indicating font size= c("tiny","scriptsize", "footnotesize","small","normalsize","large","Large","LARGE","huge","Huge") respectively. Defaulting is 5(= "normalsize").
color A character indicating color of ztable
tablewidth A numeric value indicating desired table width as a ratio to linewidth. This value is only useful when caption is longer than table length. Default value is 0.3.
type character indicating formats of ztable, either "html" or "latex". Default value is "latex"
include.rownames A logical value whether or not include rownames in the table. Default value is TRUE.
placement The table will have placement given by placement where placement must be NULL or contain only elements of "h","t","b","p","!","H". Default value is NULL.
**position**
The table will be have placed at the center of the paper if position is "center" or "c", and at the left side of the paper if it equals "left" or "l", and at the right side of the paper if it equals "right" or "r". The position is translated to specific latex environments such as "flushright" or "flushleft" or "center" (provided as a character vector) will enclose the tabular environment. Default value is "center".

**show.heading**
A logical value whether or not include headings in the table. Default value is TRUE.

**show.footer**
A logical value whether or not include headings in the table. Default value is TRUE.

**caption**
A character

**caption.placement**
The caption will be have placed at the top of the table if caption.placement is "top" and at the bottom of the table if it equals "bottom". Default value is "top".

**caption.position**
The caption will be have placed at the center of the table if caption.position is "center" or "c", and at the left side of the table if it equals "left" or "l", and at the right side of the table if it equals "right" or "r". Default value is "center".

**caption.bold**
whether or not use bold font for caption

**align**
Character vector : nchar equal to the number of columns of the resulting table indicating the alignment of the corresponding columns.

**digits**
Numeric vector of length equal to one (in which case it will be replicated as necessary) or to the number of columns of the resulting table

**display**
Character vector of length equal to the number of columns of the resulting table indicating the format for the corresponding columns. Since the row names are printed in the first column, the length of display is one greater than ncol(x) if x is a data.frame. These values are passed to the formatC function. Use "d" (for integers), "f", "e", "E", "g", "G", "fg" (for reals), or "s" (for strings). "f" gives numbers in the usual xxx.xxx format; "e" and "E" give n.ddde+nn or n.dddE+nn (scientific format); "g" and "G" put x[i] into scientific format only if it saves space to do so. "fg" uses fixed format as "f", but digits as number of significant digits. Note that this can lead to quite long result strings. Default value is NULL. the class of x.

**sidewaystable**
Logical value whether or not set the tabular environment= "sidewaystable". Requires Latex "rotating" package in preamble. Default value is FALSE.

**longtable**
Logical value whether or not set the tabular environment= "longtable". Requires Latex "longtable" package in preamble. Default value is FALSE.

**rotate**
Logical value whether or not set the tabular environment= "rotate". No special arrangement is made to find space for the result. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle(counterclockwise). Default value is FALSE.

**turn**
Logical value whether or not set the tabular environment= "turn". In this environment, Latex leaves space for the rotated table. Requires Latex "rotating" package in preamble. If TRUE, requires the rotate angle. Default value is FALSE.
angle An integer indicate the angle to rotate (degree); range -180 to 180. Default value is 0.

wraptable Logical value whether or not set the tabular environment= "wraptable". Requires Latex "wrapfig" package in preamble. Default value is FALSE.

wraptablewidth A integer indicate wraptable width in centimeter. Default=12.

tabular Logical value whether or not set the tabular environment. If TRUE, no tabular environment is set. Default value is FALSE.

label Character vector of length 1 containing the LaTeX label or HTML anchor. Set to NULL to suppress the label. Default value is NULL.

hline.after A vector of numbers between -1 and "nrow(x)", inclusive, indicating the rows after which a horizontal line should appear. If NULL is used no lines are produced. Default value is c(-1,0,nrow(x)) which means draw a line before and after the columns names and at the end of the table. Repeated values are allowed.

booktabs Logical value. If TRUE, the toprule, midrule and bottomrule tags from the Latex "booktabs" package are used rather than hline for the horizontal line tags. Requires Latex "booktabs" package in preamble. Default value is TRUE.

prefix.rows A numeric vector contains the position of rows on which extra Latex commands should be added as a prefix.

commands A character vector of the length 1 or same length of the nrow of data.frame which contains the command that should be added as a prefix at the specified rows. Default value is NULL, i.e. do not add commands.

top.command A character vector of the length 1 which contains the command that should be added as a prefix at the colnames row.

zebra Null or an integer of 0 or 1 or 2 or 3. The arguments zebra and zebra.color are used to make a Zebra striping table(table with alternating background colors) easily. A value of 1 sets background color of all odd rows/columns with specified with zebra.color. A value of 2 sets all even rows/columns. A value of 0 sets background colors of all rows/columns with colors specified with zebra.color. When zebra is 1 or 2, the parameters of prefix.rows and commands ignored. When zebra=3, the background colors can be defined by addRowColor, addColColor and addCellColor functions. Default is NULL.

zebra.color A color name or a numeric value indicating pre-defined color. When parameter zebra is 0 or 1 or 2 and zebra.color is NULL, then zebra.color is set to "platinum". Numeric values between 1 to 13 is converted to predefined color names. The predefined color names are c("peach","peach-orange","peachpuff","peach-yellow","pear","pearl","peridot","periwinkle","pastelred","pastelgray"). Default is NULL.

zebra.type An integer of 0 or 1 or 2 or 3. A value of 1 sets background colors by row. A value of 2 sets background colors by column. A value of 0 sets background colors of all cells. A value of 3 sets background colors of cells specified with zebra.list. Default value is 1.

zebra.colnames whether or not use background colors in column names row. Default value is FALSE

zebra.rownames whether or not use background colors in row names column. Default value is TRUE
zebra.list  A list consists of y, x, color. zebra.list is used only when zebra.type=3. zebra.list sets the cells specified with rows of vector "y" and columns of vector "x" with "color". The y and x are integer vector indicating rows and columns. NA value of y or x indicating all columns or rows. The color is an character vector consists of names of color.

colnames.bold  whether or not use bold font for column names, Default value is FALSE
include.colnames  Logical. If TRUE the column names is printed. Default value is TRUE.
cggroup  A character vector or matrix indicating names of column group. Default value is NULL
n.cggroup  A integer vector or matrix indicating the numbers of columns included in each cggroup Default value is NULL
rgroup  A character vector indicating names of row group. Default value is NULL
n.rgroup  A integer vector indicating the numbers of rows included in each rgroup Default value is NULL
cspan.rgroup  The number of columns that an rgroup should span. It spans by default all columns but you may want to limit this if you have column colors that you want to retain.
pcol  number of column displaying p value

Examples

```r
require(ztable)
x=head(iris)
ztbl(x)
## Not run:
ztbl(x,size=3,caption="Table 1. ztable Test")
ztbl(x,size=7,caption="Table 1. ztable Test",caption.position="l")
ztbl(x,size=7,caption="Table 1. ztable Test",caption.placement="bottom", caption.position="l")
fit=lm(mpg~.,data=mtcars)
ztbl(fit)
data(USArrests)
pr1 <- prcomp(USArrests)
ztbl(pr1)
ztbl(summary(pr1))
require(survival)
data(colon)
attach(colon)
out <- glm(status ~ rx+obstruct+adhere+nodes+extent, data=colon, family=binomial)
ztbl(out)
colon$TS = Surv(time,status==1)
out1=coxph(TS~rx+obstruct+adhere+differ+extent+surg+node4,data=colon)
ztbl(out1)
ztbl(head(mtcars),zebra=1)
ztbl(head(mtcars),zebra=1,zebra.type=2)

## End(Not run)
```
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>* datasets</td>
</tr>
<tr>
<td>- zcolors, 37</td>
</tr>
<tr>
<td>.onAttach, 3</td>
</tr>
<tr>
<td>addCellColor, 4</td>
</tr>
<tr>
<td>addcgroup, 5</td>
</tr>
<tr>
<td>addColColor, 5</td>
</tr>
<tr>
<td>addFrontColor, 6</td>
</tr>
<tr>
<td>addgroup, 6</td>
</tr>
<tr>
<td>addRowColor, 7</td>
</tr>
<tr>
<td>addSigColor, 8</td>
</tr>
<tr>
<td>addSubColNames, 8</td>
</tr>
<tr>
<td>align2html, 9</td>
</tr>
<tr>
<td>align2lines, 9</td>
</tr>
<tr>
<td>align2nd, 10</td>
</tr>
<tr>
<td>alignCheck, 10</td>
</tr>
<tr>
<td>alignCount, 10</td>
</tr>
<tr>
<td>caption2minipage, 11</td>
</tr>
<tr>
<td>cgroup2df, 11</td>
</tr>
<tr>
<td>colGroupSpan, 12</td>
</tr>
<tr>
<td>colGroupCount, 12</td>
</tr>
<tr>
<td>color2hex, 13</td>
</tr>
<tr>
<td>data2table, 13</td>
</tr>
<tr>
<td>define_colors, 14</td>
</tr>
<tr>
<td>getNewAlign, 14</td>
</tr>
<tr>
<td>getNewSpanCol, 14</td>
</tr>
<tr>
<td>getNewSpanRow, 15</td>
</tr>
<tr>
<td>getspanRowData, 15</td>
</tr>
<tr>
<td>getspanRowLength, 16</td>
</tr>
<tr>
<td>gradientColor, 16</td>
</tr>
<tr>
<td>hlines, 17</td>
</tr>
<tr>
<td>isGroupCol, 17</td>
</tr>
<tr>
<td>isspanCol, 18</td>
</tr>
<tr>
<td>isspanRow, 18</td>
</tr>
<tr>
<td>make.cell.color, 19</td>
</tr>
<tr>
<td>make.frontcolor, 20</td>
</tr>
<tr>
<td>make_align, 21</td>
</tr>
<tr>
<td>makeHeatmap, 20</td>
</tr>
<tr>
<td>myhtmlStyle, 22</td>
</tr>
<tr>
<td>name2rgb, 22</td>
</tr>
<tr>
<td>normalize2, 23</td>
</tr>
<tr>
<td>palette2colors, 23</td>
</tr>
<tr>
<td>parallelTables, 24</td>
</tr>
<tr>
<td>parallelTablesHTML, 24</td>
</tr>
<tr>
<td>parallelTablesLatex, 25</td>
</tr>
<tr>
<td>print.ztable, 25</td>
</tr>
<tr>
<td>print_ztable, 27</td>
</tr>
<tr>
<td>printHTMLHead, 26</td>
</tr>
<tr>
<td>printLatexHead, 26</td>
</tr>
<tr>
<td>printRowGroup, 26</td>
</tr>
<tr>
<td>repColor, 27</td>
</tr>
<tr>
<td>roundDf, 27</td>
</tr>
<tr>
<td>spanCol, 28</td>
</tr>
<tr>
<td>spanColWidth, 28</td>
</tr>
<tr>
<td>spanRow, 29</td>
</tr>
<tr>
<td>tableLength, 29</td>
</tr>
<tr>
<td>totalCol, 30</td>
</tr>
<tr>
<td>totalLeft, 30</td>
</tr>
<tr>
<td>tr, 31</td>
</tr>
<tr>
<td>tr2, 31</td>
</tr>
<tr>
<td>trim.ztable, 31</td>
</tr>
<tr>
<td>update_ztable, 32</td>
</tr>
<tr>
<td>validColor, 35</td>
</tr>
<tr>
<td>validColor2, 36</td>
</tr>
<tr>
<td>vline2align, 36</td>
</tr>
<tr>
<td>vlines, 37</td>
</tr>
<tr>
<td>zcolors, 37</td>
</tr>
<tr>
<td>ztable (ztable.table), 39</td>
</tr>
</tbody>
</table>
ztable.cbind.mytable, 38
ztable.mytable, 38
ztable.table, 39
ztable2flextable, 40
ztable2html, 41
ztable2latex, 41
ztable2viewer, 42
ztable_sub, 38, 40, 42