Package: prettyR (via r-universe)

November 13, 2026

Version 2.2.3.9002

Title Pretty Descriptive Stats

Date 2019-04-08

Author Jim Lemon <drjimlemon@gmail.com>, Philippe Grosjean <phgrosjean@sciviews.org>

Maintainer Jim Lemon <drjimlemon@gmail.com>

Description Functions for conventionally formatting descriptive stats, reshaping data frames and formatting R output as HTML. Compatibility updates by Renova.

Imports grDevices, stats, utils

License GPL (>= 2)

NeedsCompilation no

Repository https://renova-r.r-universe.dev

Date/Publication 2026-11-13 04:57:41 UTC

RemoteUrl https://github.com/renova-r/prettyR

RemoteRef renova-2.2.3.9002

RemoteSha dd8f1eb1f292869091e57b1cfb770121d6cb1830

Contents

add.value.labels 2
AddNav 3
AddNavItem 4
addxtabs 4
BeginNav 5
brkdn 6
calculate.xtab 7
CreateIndexFile 8
createIndxFile 9
decimal.align 9
delim.table 10

delim.xtab . . . 12
describe . . . 14
describe.factor . . . 15
describe.logical . . . 16
describe.numeric . . . 17
EndHTML . . . 18
freq . . . 18
HTMLgraph . . . 20
htmlize . . . 20
Mode . . . 22
print.desc . . . 23
print.freq . . . 24
print.xtab . . . 25
R2html . . . 26
rep_n_stack . . . 28
skew . . . 29
StartList . . . 30
StartNav . . . 30
stretch_df . . . 31
toNA . . . 32
truncString . . . 33
valid.n . . . 34
xtab . . . 34

Index **36**

`add.value.labels` *Add value labels*

Description

Adds value labels to a variable.

Usage

```
add.value.labels(x,value.labels)
```

Arguments

`x`	The variable to add the labels.
`value.labels`	The labels.

Details

'`add.value.labels`' adds value labels like those from an SPSS .sav file. It makes it a bit easier to stick on value labels that have been lost or were not there in the first place.

Value

The variable with the labels added.

Author(s)

Jim Lemon

Examples

```
fgh<-data.frame(sex=sample(1:2,20,TRUE),viviality=sample(1:3,20,TRUE))
fgh$sex<-add.value.labels(fgh$sex,c("Female","Male"))
fgh$viviality<-add.value.labels(fgh$viviality,c("Alive","Dead","Zombie"))
```

AddNav *Add a navigation item (R2html).*

Description

Add an item to the HTML navigation file.

Usage

```
AddNav(navcon,Rcommand,listname)
```

Arguments

navcon	The connection for writing to the navigation file.
Rcommand	The current R command in the script file.
listname	The name of the listing file.

Details

'AddNav' adds an entry to the navigation file, generates a name tag for that entry and returns it to be inserted into the listing file. If the R command is longer than 20 characters, it is truncated to 18 characters and an ellipsis appended.

Value

The name tag to link the listing to the navigation entry.

Author(s)

Philippe Grosjean

AddNavItem *Add a navigation item (htmlize).*

Description

Add an item to the HTML navigation file.

Usage

```
AddNavItem(Rcommand,navcon,listname,navindex)
```

Arguments

Rcommand	The current R command in the script file.
navcon	The connection for writing to the navigation file.
listname	The name of the listing file.
navindex	The number used to generate the name tag.

Details

'AddNavItem' adds an entry to the navigation file, generates a name tag for that entry and returns it to be inserted into the listing file. If the R command is longer than 20 characters, it is truncated to 18 characters and an ellipsis appended.

Value

The name tag to link the listing to the navigation entry.

Author(s)

Jim Lemon

addxtabs *Add the cells of two or more xtab objects*

Description

Adds the counts in corresponding cells of xtab objects

Usage

```
addxtabs(x)
```

Arguments

x	A list containing two or more return values from 'xtab'.

Details

'addxtabs' adds the counts in the cells of two or more 'xtab' objects by matching the row and column names. All of the 'xtab' objects in the list will usually contain counts of the same response options, although some options may be missing in some tables. This often occurs when respondents do not have to complete all of the responses (e.g. "for up to three occasions ...").

This function facilitates calculating the total counts in crosstabulations that arise from multiple responses for the options. See the example.

Value

The number of valid values or the length of the object.

Author(s)

Jim Lemon

See Also

xtab

Examples

```
# Assume that respondents are asked to record the location and quantity for
# three occasions of drinking, and for each occasion the fields are named
# "drinks" and "loc"
drinkloc<-data.frame(drinks1=sample(c("1-2","3-5","6+"),100,TRUE),
 loc1=sample(c("Meal at home","Restaurant","Party","Pub"),100,TRUE),
 drinks2=sample(c("1-2","3-5","6+"),100,TRUE),
 loc2=sample(c("Meal at home","Restaurant","Party","Pub"),100,TRUE),
 drinks3=sample(c("1-2","3-5"),100,TRUE),
 loc3=sample(c("Meal at home","Restaurant","Party"),100,TRUE))
# notice how two options have been left out in drink3 and loc3
# create the list of xtab objects
dltablist<-list()
dltablist[[1]]<-xtab(loc1~drinks1,drinkloc)
dltablist[[2]]<-xtab(loc2~drinks2,drinkloc)
dltablist[[3]]<-xtab(loc3~drinks3,drinkloc)
# now sum the corresponding cells
addxtabs(dltablist)
```

BeginNav *Write the header for the HTML navigation file (htmlize).*

Description

Initiate the navigation file for an R listing in HTML format.

Usage

```
BeginNav(navcon,bgcolor="#dddddd")
```

Arguments

navcon	The connection for writing to the navigation file.
bgcolor	The background color for the navigation frame.

Details

'BeginNav' sets up the file with the navigation frame information for the HTML listing for the 'htmlize' function.

Value

nil

Author(s)

Jim Lemon

---

brkdn *Breakdown of a numeric variable by grouping variable(s)*

Description

Calculates means, variances and ns for subgroups of numeric observations and displays the results.

Usage

```
brkdn(formula,data,maxlevels=10,num.desc=c("mean","var","valid.n"),
 width=10,round.n=2)
```

Arguments

formula	a formula with the variable to be broken down on the left and the names of one or more variables defining subgroups on the right
data	the data frame from which to select the variables
maxlevels	the maximum number of levels in any subgroup
num.desc	names of the summary functions to apply to the variable on the left side of the formula
width	The number of characters to allow for each column in the summary output.
round.n	The number of decimal places to round the output.

Details

'brkdn' will accept a formula referring to columns in a data frame. It calls 'describe.numeric' for the calculations and displays a table of results.

Value

The results of 'describe.numeric', or a multi-level list if more than one grouping variable is specified.

Author(s)

Jim Lemon

See Also

describe.numeric

Examples

```
test.df<-data.frame(Age=rnorm(100)+3*10,Sex=sample(c("M","F"),100,TRUE),
 Employ=sample(c("FT","PT","NO"),100,TRUE))
# add value labels for Employ in alphabetical order so they match
attr(test.df$Employ,"value.labels")<-c("Full time","None","Part time")
brkdn(Age~Sex+Employ,test.df)
```

---

calculate.xtab *Calculate a crosstabulation*

Description

Calculates the marginal totals and names for a crosstabulation.

Usage

```
calculate.xtab(v1,v2,varnames=NULL)
```

Arguments

v1, v2	The variables to be crosstabulated.
varnames	Labels for the variables (defaults to 'names(data)')

Details

'calculate.xtab' calls 'table' for the base table, calculates the marginal totals and returns a list with these and the names of the variables that will be used by 'print.xtab'.

Value

A list containing the value of 'table', the row and column marginals and the names of the variables.

Author(s)

Jim Lemon

See Also

table, print.xtab

`CreateIndexFile` *Write an index file for the current output (htmlize).*

Description

Write an index file for the current HTML output.

Usage

```
CreateIndexFile(HTMLbase,HTMLdir,title="R listing")
```

Arguments

`HTMLbase`	The base name for the HTML files.
`HTMLdir`	The directory in which to write the HTML files.
`title`	The title for the listing.

Details

'`CreateIndexFile`' opens a new HTML index file. If there is another file with the same name, it will be overwritten. This is intentional, as the user may want to run '`htmlize`' repeatedly without generating multiple sets of files. It then writes the frameset definition into the file and closes it.

Value

nil

Author(s)

Jim Lemon

createIndxFile *Write an index file for the current output (R2html).*

Description

Write an index file for the current R2html output.

Usage

```
createIndxFile(HTMLfile,navfile,listfile,title="R listing")
```

Arguments

HTMLfile	The file name for the HTML files.
navfile	The name for the HTML navigation frame file.
listfile	The name for the HTML listing file.
title	The title for the listing.

Details

'createIndxFile' opens a new HTML index file. If there is another file with the same name, it will be overwritten. This is intentional, as the user may want to run 'R2html' repeatedly without generating multiple sets of files. It then writes the frameset definition into the file and closes it.

Value

nil

Author(s)

Philippe Grosjean

---

decimal.align *Turn numbers into strings with aligned decimal points*

Description

Formats decimal numbers as strings with aligned decimal points.

Usage

```
decimal.align(x,dechar=".",nint=NA,ndec=NA,pad.left=TRUE)
```

Arguments

x	One or more decimal numbers.
dechar	The character used to separate the decimal part of a number.
nint	The number of characters to which the integer part of the numbers should be padded.
ndec	The number of characters to which the decimal part of the numbers should be padded.
pad.left	Whether the left (integer) side of the numbers should be padded as well as the right.

Details

'decimal.align' splits the incoming numbers at the decimal point and pads the decimal part and optionally the integer part so that when the numbers are displayed in a monospaced font the decimal points will be aligned. Note that if an integer or a decimal part without an integer is passed, the function will insert a zero for the missing part.

This is useful for displaying or storing aligned columns of decimal numbers when the user does not want to pad the decimal part with zeros as in the 'format' function.

Value

The original numbers as strings, padded with spaces.

Author(s)

Jim Lemon

See Also

sprintf

Examples

```
x<-c(1,2.3,44.55,666.777)
decimal.align(x)
```

---

delim.table *Format a 2D table*

Description

Format a 2D table with delimiters and other formatting commands

Usage

```
delim.table(x,filename="",delim=",",tabegin="",bor="",eor="\n",tablend="",
 label=deparse(substitute(x)),header=NULL,trailer=NULL,html=FALSE,
 show.rownames=TRUE,leading.delim=FALSE,show.all=FALSE,nsignif=4,
 con,open.con=FALSE)
```

Arguments

x	A list, matrix or data frame that is to be formatted.
filename	Name for a file to which the result will be written.
delim	The delimiter to place between entries in the table(s).
tabegin	Any formatting commands to be placed before the table.
bor	The formatting command for the beginning of a table row.
eor	The formatting command for the end of a table row.
tablend	Any formatting commands to be placed after the table.
label	A label to be displayed before the table.
header	A character string to be written at the beginning of the output file.
trailer	A character string to be written at the end of the output file.
html	If TRUE, all table formatting commands that are not explicitly specified will be altered to HTML tags.
show.rownames	Whether to display the rownames of a table.
leading.delim	Whether to add an extra delimiter at the beginning of the table. See Details.
show.all	Whether to show all the components of a list. The default FALSE is to show only those components that look like 2D tables.
nsignif	Number of significant digits for numeric display.
con	A connection to which the output will be written. If a filename is passed, it will be ignored if 'con' is not missing.
open.con	A flag for an open connection. This argument is used by the function to keep track of connections in recursive calls and should not be altered by the user.

Details

'delim.table' tries to format its first argument into one or more tables that will be displayed in another application. The most common use is to produce a CSV style file that can be imported into a spreadsheet. The default values for 'delim' and 'eor' should be adequate for this, and all the user has to do is to supply a filename as in the first example. When a filename is provided, the function attempts to open the file, write its output to it and close it again. In order to deal with the multilevel lists that are often produced by other functions, the function calls itself until it reaches the lowest level of the list, where it can successfully format the contents. Thus the function only passes the connection, not the filename, in recursive calls. If the user passes both a filename and a valid connection, the output will be written to the connection and the filename will not be used.

'delim.table' will fail if passed a table with more than two dimensions. However, the function will process 2D "slices" of such tables if called with one of the 'apply' family of functions or manually for each slice.

'delim.table' can also be used to format HTML tables as in the second example. If the user wants different tags from the defaults, pass these explicitly. In principle, any markup language that can produce a table using commands that include; commands to begin and end the table, a command to start and end a row, and a command to start a new cell.

'delim.table' is consistent with the default CSV arguments, adding an extra delimiter to the first line if there are row names. The user should set 'leading.delim' to FALSE for a table without the empty cell at the upper left. When 'delim.table' is used to format tables for 'htmlize', it should not attempt to open a new connection.

An unexpected use of 'delim.table' is producing tables that can be imported into OpenOffice Writer and most other word processors. While the tables in an HTML listing can be imported directly, the formatting is usually not suitable. If a table is output to an HTML or text document formatted with TAB characters as the delimiter as in the third example, the output can be copied and pasted into the word processor document and then converted to a table.

Value

nil

Author(s)

Jim Lemon

See Also

htmlize

Examples

```
testdf<-data.frame(a=sample(0:1,100,TRUE),b=rnorm(100),c=rnorm(100))
testglm<-summary(glm(a~b*c,testdf,family="binomial"))
# produce a CSV file to import into a spreadsheet, just the coefficients
delim.table(testglm$coefficients,"testglm.csv")
# now create an HTML file of the three tables in the result
# add a background color different from the default
delim.table(testglm,"testglm.html",header="<html><body bgcolor=\"#ffaaff\">",
 html=TRUE)
# these tables can be pasted into a word processor and converted
# using "Insert|Table" or similar commands
delim.table(testglm,"testglm.tab",delim="\t",leading.delim=FALSE)
# to clean up, delete the files "testglm.csv", "testglm.tab" and "testglm.html"
```

---

delim.xtab *Format a crosstabulation*

Description

Format a 2D crosstabulation from xtab

Usage

```
delim.xtab(x,pct=c("row","column","cell"),coltot=TRUE,rowtot=TRUE,
 ndec=1,delim="\t",interdigitate=TRUE,label=deparse(substitute(x)))
```

Arguments

x	An object of class 'xtab'.
pct	Whether and how to calculate percentages.
coltot, rowtot	Whether to add the marginal totals.
ndec	The number of decimal places for percentages.
delim	The delimiter to use between columns. Defaults to TAB.
interdigitate	Whether to place each column of percentages next to its row of counts.
label	A label to be displayed before the table.

Details

'delim.xtab' formats a crosstabulation in a manner similar to those produced by commercial spreadsheets, with a column of counts followed by a column of percentages. If 'interdigitate' is FALSE, the percentages will be displayed separately.

'delim.xtab' will only process one 2D xtab object at a time.

To format only the counts, set 'pct' to NA.

The intended use of 'delim.xtab' is producing tables that can be imported into most word processors. If a table is output to an HTML or text document formatted with TAB characters, the output can be copied and pasted into the word processor document and then converted to a table.

Value

nil

Author(s)

Jim Lemon

See Also

xtab

Examples

```
alpha1<-sample(LETTERS[1:3],50,TRUE)
alpha2<-sample(LETTERS[1:2],50,TRUE)
alphas<-data.frame(alpha1,alpha2)
alphatab<-xtab(alpha1~alpha2,alphas)
delim.xtab(alphatab,pct="row",interdigitate=TRUE)
delim.xtab(alphatab,pct="column",interdigitate=TRUE)
delim.xtab(alphatab,pct="cell",interdigitate=TRUE)
```

describe *Description of variables*

Description

Describes a vector or the columns in a matrix or data frame.

Usage

```
describe(x,num.desc=c("mean","median","var","sd","valid.n"),xname=NA,
 horizontal=FALSE)
```

Arguments

x	A vector, matrix or data frame.
num.desc	The names of the functions to apply to numeric data.
xname	A name for the object 'x', mostly where this would be a very long string describing its structure (e.g. if it was extracted by name from a data frame).
horizontal	Whether to display the results of 'describe.factor' across the page (TRUE) or down the page (FALSE).

Details

'describe' displays a table of descriptive statistics for numeric, factor and logical variables in the object 'x'. The summary measures for numeric variables can easily be altered with the argument 'num.desc'. Pass a character vector with the names of the desired summary measures and these will be displayed at the top of the numeric block with their results beneath them. If quantiles are desired, the user will have to write wrapper functions that call 'quantile' with the appropriate type or probabilities and use the names of the wrapper functions in 'num.desc'. Remember that any function called by 'describe' must have an 'na.rm' argument.

Percentages are now always displayed and returned in the tables for factor and logical variables.

Value

A list with three components:

Numeric	A list of the values returned from 'describe.numeric' for each column described.
Factor	A list of the tables for each column described.
Logical	A list of the tables for each column described.

Author(s)

Jim Lemon

See Also

Mode, valid.n, describe.numeric, describe.factor

Examples

```
sample.df<-data.frame(sex=sample(c("MALE","FEMALE"),100,TRUE),
 income=(rnorm(100)+2.5)*20000,suburb=factor(sample(1:4,100,TRUE)))
# include the maximum values
describe(sample.df,num.desc=c("mean","median","max","var","sd","valid.n"))
# make up a function
roughness<-function(x,na.rm=TRUE) {
 if(na.rm) x<-x[!is.na(x)]
 xspan<-diff(range(x))
 return(mean(abs(diff(x))/xspan,na.rm=TRUE))
}
# now use it
describe(sample.df$income,num.desc="roughness",xname="income")
```

---

describe.factor *Description of factor variables*

Description

Describes a factor variable.

Usage

```
describe.factor(x,varname="",horizontal=FALSE,decr.order=TRUE)
```

Arguments

x	A factor.
varname	A name for the variable. 'describe' always passes the name.
horizontal	Whether to display the results across the page (TRUE) or down the page (FALSE).
decr.order	Whether to order the rows by decreasing frequency.

Details

'describe.factor' displays the name of the factor, a table of its values, the modal value of the factor and the number of valid (not NA) values.

Value

nil

Author(s)

Jim Lemon

See Also

Mode, valid.n, table

`describe.logical` *Description of logical variables*

Description

Describes a logical variable.

Usage

```
describe.logical(x,varname="")
```

Arguments

`x`	A logical.
`varname`	An optional name for the variable. '`describe`' always passes the name of the variable.

Details

'`describe.logical`' displays the name of the logical, a table of counts of its two values (TRUE, FALSE) and the percentage of TRUE values.

Value

nil

Author(s)

Jim Lemon

See Also

table

`describe.numeric` *Description of numeric variables*

Description

Describes a numeric variable.

Usage

```
describe.numeric(x,varname="",
 num.desc=c("mean","median","var","sd","valid.n"))
```

Arguments

`x`	A numeric vector.
`varname`	The variable name to display.
`num.desc`	The names of the functions to apply to the vector.

Details

'`describe.numeric`' displays the name of the vector and the results of the functions whose names are passed in '`num.desc`'. Note that any functions that are called by '`describe.numeric`' must have an '`na.rm`' argument, even if it is a dummy.

Value

The vector of values returned from the functions in '`num.desc`'.

Author(s)

Jim Lemon

See Also

describe, valid.n

Examples

```
x<-rnorm(100)
# include a function that calculates the "smoothness" of a vector
# of numbers by calculating the mean of the absolute difference
# between each successive value - all values equal would be 0
smoothness<-function(x,na.rm=TRUE) {
 if(na.rm) x<-x[!is.na(x)]
 xspan<-diff(range(x))
 return(mean(abs(diff(x))/xspan,na.rm=TRUE))
}
describe(x,num.desc=c("mean","median","smoothness"),xname="x")
```

```
# now sort the values to make the vector "smoother"
describe(sort(x),num.desc=c("mean","median","smoothness"),xname="x")
```

EndHTML *End an HTML file (htmlize).*

Description

Append an ending to an HTML file.

Usage

```
EndHTML(con,ending=NULL)
```

Arguments

con	The connection for writing to the HTML file.
ending	Any "trailer" information to be added to the file before closing it.

Details

'EndHTML' appends a brief ending to an HTML file.

Value

nil

Author(s)

Jim Lemon

---

freq *Calculate a frequency table*

Description

Calculates one or more frequency table(s) from a vector, matrix or data frame.

Usage

```
freq(x,variable.labels=NULL,display.na=TRUE,decr.order=TRUE)
```

Arguments

x	a vector, matrix or data frame.
variable.labels	optional labels for the variables. The default is the name of the variable passed or the 'names' attribute if the variable has more than 1 dimension.
display.na	logical - whether to display counts of NAs.
decr.order	Whether to order each frequency table in decreasing order.

Details

'freq' calls 'table' to get the frequency counts and builds a list with one or more components containing the value labels and counts.

Value

A list with one or more components. Each component includes the values of the relevant variable as the names.

Note

The limit on the number of bins has been removed, so passing a numeric vector with many levels may produce a huge, useless "frequency" table.

Author(s)

Jim Lemon

See Also

print.freq

Examples

```
A<-sample(1:10,130,TRUE)
A[sample(1:130,6)]<-NA
C<-sample(LETTERS[1:14],130,TRUE)
C[sample(1:130,7)]<-NA
test.df<-data.frame(A,C)
freq(test.df)
```

HTMLgraph *Create a graphic in HTML output (R2html).*

Description

Creates a graphic file and links it to the HTML output.

Usage

```
HTMLgraph(listfile, graphfile = NULL,type = "png",...)
```

Arguments

listfile	The name of the HTMLize listing file.
graphfile	The name for the graphic file (see Details).
type	The graphic format in which to write the image.
...	Additional arguments - currently ignored.

Details

'HTMLgraph' sets up a graphic device to allow an R graphic to be written to a file and that file linked to the HTML listing. If no filename is passed, a name is constructed from 'fig' and a number that does not match any existing 'fignnn...' file. Only 'bmp, jpeg' and 'png' files are currently handled, defaulting to the last.

Value

nil

Author(s)

Philippe Grosjean

htmlize *Read an R script and write HTML output*

Description

Produces HTML output from an R script.

Usage

```
htmlize(Rfile,HTMLbase,HTMLdir,title,
 bgcolor="#dddddd",echo=FALSE,do.nav=FALSE,useCSS=NULL,...)
```

Arguments

Rfile	The R script file from which to read the commands.
HTMLbase	The base name for the HTML files (see Details).
HTMLdir	The directory in which to write the HTML output.
title	The title of the HTML page and the headings for the frames. See Details for including the title in the R script.
bgcolor	The background color for the frames.
echo	Whether to include ("echo") the commands in the listing.
do.nav	Whether to have a navigation window.
useCSS	The name of a CSS stylesheet that will define the appearance of components of the HTML display. If this is not NULL, the CSS file should exist.
...	Additional arguments - currently ignored.

Details

'htmlize' allows the user to produce a basic HTML listing from an existing R script. The script must already run correctly with 'source'. If the first line of the R script is a comment starting with '#title~' and the 'title' argument is missing, the rest of the first line will be used as the title of the HTML output.

If there is any graphic output, the script must contain the necessary commands to set up the graphic devices. Note that only TIFF, GIF, BMP, JPEG and PNG graphic images are generally viewable in HTML browsers. The last two are probably the most reliable, but see their help pages for more details. The graphic files will be linked to the HTML listing page so that they should be interleaved with text output and commands.

If 'do.nav' is TRUE, three files will be output. The first will be named 'HTMLbase.html', where 'HTMLbase' is whatever string has been passed as that argument. If that argument is missing, the function will attempt to munge the 'Rfile' argument into a base name. This file is an "index" file that sets up the HTML frameset. The second file will be named 'HTMLbase_nav.html' and will be dispayed at the left side of the HTML output as a "navigation" list using the commands as names. Commands longer than 20 characters will be truncated. The third file, named 'HTMLbase_list.html', contains the program listing. All three files will be written in 'HTMLdir'. If this is missing, the path of 'Rfile' will be used.

If 'do.nav' is FALSE, only one file will be written. It will have the same content as the 'HTMLbase_list.html' file except without the name tags for navigation and it will be named 'HTMLbase.html'.

Commands that create or alter connections, such as 'sink' are "forbidden", not evaluated and marked as comments in the listing. This prevents such commands from altering the connections necessary to write the HTML files.

If there is a function defined in the R script, 'htmlize' will run, but not write any output after the function definition. This has to do with the way that 'htmlize' reads command lines from the script file. This is a bug, so watch this space for a solution.

The ability to nominate a CSS stylesheet allows the user to customize the appearance of the HTML output. The most likely use of the 'useCSS' argument is for the user to specify whatever aspects of the HTML display are to be different from the default browser values in a stylesheet.

Value

nil

Note

As of version 1.1, both 'echo' and 'do.nav' are FALSE by default, as these defaults seem to be more popular.

The major differences between 'htmlize' and 'R2html' are: 'htmlize' will run with a minimum of one argument ('Rfile') and produces very basic HTML output. It requires the graphic devices to be set up as commands in the R script.

'R2html' does not require commands for graphic devices, just comments at the end of any line that requires graphic output to the HTML file. It color codes commands and output and is more careful about the content of lines it writes.

Author(s)

Jim Lemon with improvements by Phillipe Grosjean

Examples

```
rcon<-file("test.R","w")
cat("#title~This is the title\n")
cat("test.df<-data.frame(a=factor(sample(LETTERS[1:4],100,TRUE)),\n",
 file=rcon)
cat(" b=sample(1:4,100,TRUE),c=rnorm(100),d=rnorm(100))\n",file=rcon)
cat("describe(test.df)\n",file=rcon)
cat("print(freq(test.df$a))\n",file=rcon)
cat("xtab(a~b,test.df)\n",file=rcon)
cat("brkdn(c~b,test.df)\n",file=rcon)
cat("png(\"hista.png\")\nhist(test.df$b)\ndev.off()\n",file=rcon)
cat("png(\"plotcd.png\")\nplot(test.df$c,test.df$d)\ndev.off()\n",file=rcon)
close(rcon)
# call htmlize with minimal arguments
htmlize("test.R")
# if you want to see the output, use the following line
# system(paste(options("browser")," file:",getwd(),"/test.html",sep="",collapse=""))
# to clean up, use the following line
# system("rm test.R test.html hista.png plotcd.png")
```

Mode *Find the modal value*

Description

Finds the modal value of an object (usually a factor).

Usage

```
Mode(x,na.rm=FALSE)
```

Arguments

x	An object, usually a factor.
na.rm	A dummy argument to make it compatible with calls to 'mean', etc.

Details

'Mode' finds the modal value of the object. If there are multiple modal values, it returns an appropriate message. If 'Mode' is called with a continuous variable, it will not in general return a sensible answer. It does not attempt to estimate the density of the values and return an approximate modal value.

Value

The modal value of the object as a character string.

Note

This is not the same as 'mode' that determines the data mode of an object.

Author(s)

Jim Lemon

See Also

describe

---

print.desc *Display descriptive stats output*

Description

Displays a list of descriptive statistics produced by 'describe'.

Usage

```
 ## S3 method for class 'desc'
print(x,ndec=2,...)
```

Arguments

x	a list of descriptive statistics produced by 'describe'
ndec	The number of decimal places to display.
...	additional arguments passed to 'print'

Details

'print.desc' displays the list of descriptive statistics produced by the 'describe' function.

Value

nil

Author(s)

Jim Lemon

See Also

describe

Examples

```
test.df<-data.frame(A=c(sample(1:10,99,TRUE),NA),C=sample(LETTERS,100,TRUE))
test.desc<-describe(test.df)
print(test.desc)
```

---

print.freq *Display frequency table(s)*

Description

Displays one or more frequency tables produced by 'freq'.

Usage

```
 ## S3 method for class 'freq'
print(x,show.pc=TRUE,cum.pc=FALSE,show.total=FALSE,...)
```

Arguments

x	a frequency table produced by '\link{freq}'
show.pc	Whether to display percentages as well as counts.
cum.pc	Whether to display cumulative percentages.
show.total	Whether to display the total count of observations.
...	additional arguments passed to 'print'.

Details

'print.freq' displays frequency tables produced by 'freq'. If 'show.pc' is TRUE and there is a value in the frequency table with the label "NA", an additional set of percentages ignoring that value will also be displayed. If 'show.total' is TRUE, the total number of observations will be displayed.

Value

nil

Author(s)

Jim Lemon

See Also

freq

Examples

```
test.df<-data.frame(A=c(sample(1:10,99,TRUE),NA),C=sample(LETTERS,100,TRUE))
test.freq<-freq(test.df)
print(test.freq,show.total=TRUE)
```

---

print.xtab *Display a 2D crosstabulation*

Description

Displays a 2D crosstabulation with optional chi-squared test, odds ratio/relative risk and phi coefficient.

Usage

```
 ## S3 method for class 'xtab'
print(x,col.width=8,or=TRUE,chisq=TRUE,phi=TRUE,
  rowname.width=NA,html=FALSE,bgcol="lightgray",...)
```

Arguments

x	The list returned by 'calculate.xtab'.
col.width	Width of the columns in the display.
or	whether to calculate the odds ratio and relative risk (only for 2x2 tables).
chisq	Whether to call 'chisq.test' and display the result.
phi	Whether to calculate and display the phi coefficient (only for 2x2 tables).
rowname.width	Optional width for the rownames. Mostly useful for truncating very long rownames.
html	Whether to format the table with HTML tags.
bgcol	Background color for the table.
...	additional arguments passed to 'chisq.test'.

Details

'print.xtab' displays a crosstabulation in a fairly conventional style with row, column and marginal percentages. If 'html' is TRUE, the formatting will use HTML tags and will only be useful if viewed in an HTML browser. In order to get HTML formatting, save, the output of 'xtab' and print with the argument 'html=TRUE'.

If 'or' is 'TRUE' and the resulting table is 2x2, the odds ratio will be displayed below the table. If the function 'logical.names' within 'print.xtab' finds that the column margin names are one of FALSE/TRUE, 0/1 or NO/YES in those orders, the risk of the column variable for the second level of the row variable relative to the first row variable will be displayed as well.

Value

nil

Author(s)

Jim Lemon

See Also

calculate.xtab, xtab

R2html *Read an R script and write HTML output*

Description

Produces HTML output from an R script.

Usage

```
R2html(Rfile,HTMLfile,echo=TRUE,split=FALSE,browse=TRUE,
 title="R listing",bgcolor="#dddddd",...)
```

Arguments

Rfile	The R script file from which to read the commands.
HTMLfile	The name for the HTML index file (see Details).
echo	Whether to include ("echo") the commands in the listing.
split	Whether to split the output (see '\link{sink}')
.	
browse	Whether to automatically open the HTML output in the default browser when finished.
title	The title of the HTML page and the headings for the frames.
bgcolor	The background color for the frames.
...	Additional arguments - currently ignored.

Details

'R2html' allows the user to produce an HTML listing from an existing R script. The script must already run correctly and, if there is any graphic output, contain the necessary comments at the end of each graphic command to set up the graphic devices. The graphic files will be linked to the HTML listing page so that they should be interleaved with text output and commands.

Three files will be output. The first will be named 'HTMLfile' which must be a valid filename with the extension '.html'. This file is the "index" file that sets up the HTML frameset. The second file will be named by concatenating 'HTMLfile' without its extension and '_nav.html'. Its contents will be dispayed at the left side of the HTML output as a "navigation" list using the commands as names. The third file is named by concatentating 'HTMLfile' without its extension and '_list.html'. This contains the program listing. All three files will be written in whatever directory is specified by the path to 'HTMLfile'. If this is missing, everything will be written in the current R directory.

Commands that create or alter connections, such as 'sink' are "forbidden", not evaluated and marked as comments in the listing. This prevents such commands from altering the connections necessary to write the HTML files.

To include graphic output in the HTML file, place a comment at the end of any function that produces a graphic like this '#--FIG:filename.png--' and the appropriate graphic device is automatically set up. The filename may be left out, in which case a name will be generated.

Value

nil

Author(s)

Philippe Grosjean

Examples

```
rcon<-file("testR2html.R","w")
cat("test.df<-data.frame(a=factor(sample(LETTERS[1:4],100,TRUE)),\n",
 file=rcon)
cat(" b=sample(1:4,100,TRUE),c=rnorm(100),d=rnorm(100))\n",file=rcon)
cat("describe(test.df)\n",file=rcon)
cat("print(freq(test.df$a))\n",file=rcon)
cat("xtab(a~b,test.df)\n",file=rcon)
cat("brkdn(c~b,test.df)\n",file=rcon)
cat("hist(test.df$b)#--FIG:hista.png--\n",file=rcon)
cat("plot(test.df$c,test.df$d)#--FIG:plotcd.png--\n",file=rcon)
close(rcon)
# R2html("testR2html.R", "testR2html.html")
# if you want to see the output, use the following line
# system(paste(options("browser")," file:",getwd(),"/testR2html.html",sep="",collapse=""))
# to clean up, use the following line
# system("rm testR2html.R testR2html.html testR2html_nav.html")
# system("rm testR2html_list.html hista.png plotcd.png")
```

`rep_n_stack` *Replicate and stack columns of a data frame*

Description

Reshape a data frame by stacking two or more columns into one and adding a factor, while replicating the remaining columns and stacking them to match the number of rows

Usage

```
rep_n_stack(data,to.stack,stack.names=NULL)
```

Arguments

`data`	A data frame.
`to.stack`	Which columns are to be stacked together (see Details).
`stack.names`	Names for the new factor and stacked column.

Details

'`rep_n_stack`' takes two or more specified columns in a data frame and "stacks" them into a single column. It also creates a new factor composed of the replicated names of the columns that identifies from which column each value came. The remaining columns in the data frame are replicated to match the new number of rows.

If '`to.stack`' is a matrix of names or column numbers, '`rep_n_stack`' will stack each row into two new columns, allowing multiple related sets of values to be stacked in one operation.

A matrix or data frame of values can now be stacked so that the values can be displayed by a function like '`barNest`' in the **plotrix** package.

Value

The reshaped data frame.

Note

'`rep_n_stack`' only does what other reshaping functions can do, but may be more easy to understand.

Author(s)

Jim Lemon

See Also

reshape

Examples

```
wide.data<-data.frame(ID=1:10,Glup=sample(c("Montic","Subtic"),10,TRUE),
 Flimit1=runif(10,1,2),Flimit2=runif(10,1.5,2.5),Flimit3=runif(10,1.2,3),
 Glimit1=rnorm(10,mean=5),Glimit2=rnorm(10,mean=4),Glimit3=rnorm(10,mean=4.5))
# first just stack one set of related measures
rep_n_stack(wide.data[,1:5],to.stack=c("Flimit1","Flimit2","Flimit3"))
# now stack two sets of related measures and pass names for the stacks
rep_n_stack(wide.data,to.stack=matrix(3:8,nrow=2,byrow=TRUE),
 stack.names=c("Limit_F","Value_F","Limit_G","Value_G"))
# finally stack a matrix of means into a single column with the
# row and column names becoming "factor" variables
meanmat<-matrix(runif(16,10,20),nrow=4)
rownames(meanmat)<-c("Plunderers","Storers","Refusers","Jokers")
colnames(meanmat)<-c("Week1","Week2","Week3","Week4")
rep_n_stack(meanmat,to.stack=1:4,
 stack.names=c("Returns","Occasion","Strategy"))
```

---

skew *Calculate the skew of a distribution.*

Description

Calculates the skew of a distribution using a simple formula.

Usage

```
skew(x)
```

Arguments

x	A vector of numeric values, supposedly drawn from a particular distribution.

Details

'skew' calculates the asymmetry of the distribution of the values.

Value

The value calculated.

Author(s)

Jim Lemon

See Also

sd

StartList — *Write the header for the HTML listing file (htmlize).*

Description

Initiate the listing file for an R listing in HTML format.

Usage

```
StartList(listcon,title="R listing",bgcolor="#dddddd",useCSS=NULL)
```

Arguments

listcon	The connection for writing to the listing file.
title	The title and heading for the listing frame.
bgcolor	The background color for the listing frame.
useCSS	Path and filename of a CSS stylesheet.

Details

'StartList' sets up the file with the listing frame information for the HTML listing.

Value

nil

Author(s)

Jim Lemon

StartNav — *Write the header for the HTML navigation file (R2html).*

Description

Initiate the navigation file for an R listing in HTML format.

Usage

```
StartNav(navcon,title="R listing",bgcolor="#dddddd")
```

Arguments

navcon	The connection for writing to the navigation file.
title	Title for the navigation window.
bgcolor	The background color for the navigation frame.

Details

'StartNav' sets up the file with the navigation frame information for the HTML listing for 'R2html'.

Value

nil

Author(s)

Philippe Grosjean

---

stretch_df *Reshape a data frame from "long" to "wide" format*

Description

Reshape a data frame by reducing the multiple rows of repeated variables to a single row for each instance (usually a "case" or object) and stretching out the variables that are not repeated within each case.

Usage

```
stretch_df(x,idvar,to.stretch,ordervar=NA,include.ordervar=TRUE)
```

Arguments

x	A data frame.
idvar	A variable that identifies instances (cases or objects).
to.stretch	Which variables are to be stretched out in the single row.
ordervar	Variable that gives the order of the stretched variables.
include.ordervar	Include the ordering variable in the output.

Details

'stretch_df' takes a data frame in which at least some instances have multiple rows and reshapes it into a "wide" format with one row per instance. The variable passed as 'idvar' distinguishes the instances, and will be the first column in the new data frame. All other variables in the data frame except those named in 'to.stretch' and 'ordervar' will follow 'idvar'. The variables named in 'to.stretch' will follow the variables that are not repeated in the initial data frame, along with the order variable if 'include.ordervar' is TRUE.

Value

The reshaped data frame.

Note

'stretch_df' mostly does what other reshaping functions can do, but may be more easy to understand. It will stretch multiple variables, something that some reshaping functions will not do.

Author(s)

Jim Lemon

See Also

reshape

Examples

```
# create a data frame with two repeated variables
longdf<-data.frame(ID=c(rep(111,3),rep(222,4),rep(333,6),rep(444,3)),
 name=c(rep("Joe",3),rep("Bob",4),rep("Sue",6),rep("Bea",3)),
 score1=sample(1:10,16,TRUE),score2=sample(0:100,16),
 scoreorder=c(1,2,3,4,3,2,1,4,6,3,5,1,2,1,2,3))
stretch_df(longdf,"ID",c("score1","score2"),"scoreorder")
```

---

toNA *Set specified values in an object to NA*

Description

Sets all specified values in an object to NA.

Usage

```
toNA(x,values=NA)
```

Arguments

x	A vector, matrix or data frame (max 2D).
values	One or more values that are to be set to NA.

Details

'toNA' sets all entries in an object in values to NA. Useful for converting various missing value samps to NA.

Value

The object with NAs replacing all specified values.

Author(s)

Jim Lemon

See Also

%in%

`truncString` *Truncate strings and add ellipses if a string is truncated.*

Description

Truncates one or more strings to a specified length, adding an ellipsis (...) to those strings that have been truncated. The justification of the strings can be controlled by the '`justify`' argument as in format, which does the padding and justification.

Usage

```
truncString(x,maxlen=20,justify="left")
```

Arguments

`x`	A vector of strings.
`maxlen`	The maximum length of the returned strings.
`justify`	Justification for the new strings.

Value

The string(s) passed as '`x`' now with a maximum length of '`maxlen`'.

Author(s)

Jim Lemon

See Also

substr

valid.n *Find the number of valid (not NA) values*

Description

Finds the number of valid (not NA) or total values in an object.

Usage

```
valid.n(x,na.rm=TRUE)
```

Arguments

x	An object.
na.rm	Whether to count all values (FALSE) or only those not NA.

Details

'valid.n' finds the number of valid values of the object if 'na.rm=TRUE'.

Value

The number of valid values or the length of the object.

Author(s)

Jim Lemon

See Also

describe

xtab *Crosstabulate variables*

Description

Crosstabulates variables with small numbers of unique values.

Usage

```
xtab(formula,data,varnames=NULL,or=TRUE,chisq=FALSE,phi=FALSE,html=FALSE,
 bgcol="lightgray",lastone=TRUE)
```

Arguments

formula	a formula containing the variables to be crosstabulated
data	the data frame from which to select the variables
varnames	optional labels for the variables (defaults to 'names(data))'
or	whether to calculate the odds ratio (only for 2x2 tables).
chisq	logical - whether to display chi squared test(s) of the table(s)
phi	whether to calculate and display the phi coefficient of association - only for 2x2 tables
html	whether to format the resulting table with HTML tags.
bgcol	background color for the table if html=TRUE.
lastone	A flag that controls the names of the returned list.

Details

'xtab' will accept a formula referring to columns in a data frame or two explicit variable names. It calls 'calculate.xtab' for the calculations and displays one or more tables of results by calling 'print.xtab'. If 'html' is TRUE, the resulting table will be formatted with HTML tags. The default value of 'lastone' should not be changed.

Value

The result of 'calculate.xtab' if there is only one table to display, otherwise a nested list of tables.

Author(s)

Jim Lemon

See Also

table, calculate.xtab, print.xtab

Examples

```
test.df<-data.frame(sex=sample(c("MALE","FEMALE"),1000,TRUE),
suburb=sample(1:4,1000,TRUE),social.type=sample(LETTERS[1:4],1000,TRUE))
xtab(sex~suburb+social.type,test.df,chisq=TRUE)
# now add some value labels
attr(test.df$suburb,"value.labels")<-1:4
names(attr(test.df$suburb,"value.labels"))<-
 c("Upper","Middle","Working","Slum")
attr(test.df$social.type,"value.labels")<-LETTERS[1:4]
names(attr(test.df$social.type,"value.labels"))<-
 c("Gregarious","Mixer","Aloof","Hermit")
xtab(sex~suburb+social.type,test.df)
# now have some fun with ridiculously long factor labels
testxtab<-data.frame(firstbit=sample(c("Ecomaniacs","Redneck rogues"),50,TRUE),
 secondbit=sample(c("Macho bungy jumpers","Wimpy quiche munchers"),50,TRUE))
# and format the table in HTML and add some tests
xtab(secondbit~firstbit,testxtab,html=TRUE,chisq=TRUE,phi=TRUE)
```

Index

∗ **misc**
add.value.labels, 2
AddNav, 3
AddNavItem, 4
addxtabs, 4
BeginNav, 5
brkdn, 6
calculate.xtab, 7
CreateIndexFile, 8
createIndxFile, 9
decimal.align, 9
delim.table, 10
delim.xtab, 12
describe, 14
describe.factor, 15
describe.logical, 16
describe.numeric, 17
EndHTML, 18
freq, 18
HTMLgraph, 20
htmlize, 20
Mode, 22
print.desc, 23
print.freq, 24
print.xtab, 25
R2html, 26
rep_n_stack, 28
skew, 29
StartList, 30
StartNav, 30
stretch_df, 31
toNA, 32
truncString, 33
valid.n, 34
xtab, 34
%in%, *33*

add.value.labels, 2
AddNav, 3
AddNavItem, 4
addxtabs, 4

BeginNav, 5
brkdn, 6

calculate.xtab, 7, *26*, *35*
CreateIndexFile, 8
createIndxFile, 9

decimal.align, 9
delim.table, 10
delim.xtab, 12
describe, 14, *17*, *23*, *24*, *34*
describe.factor, 15, *15*
describe.logical, 16
describe.numeric, *7*, *15*, 17

EndHTML, 18

format, *33*
freq, 18, *25*

HTMLgraph, 20
htmlize, *12*, 20

Mode, *15*, *16*, 22

print.desc, 23
print.freq, *19*, 24
print.xtab, *8*, 25, *35*

R2html, 26
rep_n_stack, 28
reshape, *28*, *32*

sd, *29*
skew, 29
sprintf, *10*
StartList, 30
StartNav, 30
stretch_df, 31

substr, *33*

table, *8*, *16*, *35*
toNA, 32
truncString, 33

valid.n, *15–17*, 34

xtab, *5*, *13*, *26*, 34