

Package ‘prettyR’

November 11, 2009

Version 1.7

Title Pretty descriptive stats.

Date 2009-11-11

Author Jim Lemon <jim@bitwrit.com.au>, Philippe Grosjean
<phgrosjean@sciviews.org>

Maintainer Jim Lemon <jim@bitwrit.com.au>

Description Functions for conventionally formatted descriptive stats, and to format R output as HTML.

Depends

License GPL (>= 2)

Repository CRAN

Date/Publication 2009-11-11 13:10:24

R topics documented:

add.value.labels	2
AddNav	3
AddNavItem	3
BeginNav	4
brkdn	5
calculate.xtab	6
CreateIndexFile	7
createIndxFile	7
delim.table	8
describe	10
describe.factor	11
describe.logical	12
describe.numeric	13
EndHTML	14

freq	15
HTMLgraph	16
htmlize	17
Mode	19
print.freq	20
print.xtab	21
R2html	22
StartList	23
StartNav	24
toNA	25
valid.n	25
xtab	26
Index	28

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

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

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

`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

<code>navcon</code>	The connection for writing to the navigation file.
<code>bgcolor</code>	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"),
      vname.width=NULL, width=10, round.n=2)
```

Arguments

<code>formula</code>	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
<code>data</code>	the data frame from which to select the variables
<code>maxlevels</code>	the maximum number of levels in any subgroup
<code>num.desc</code>	names of the summary functions to apply to the variable on the left side of the formula
<code>vname.width</code>	The number of characters to allow for printing the variable name. If NULL, the function will calculate this.
<code>width</code>	The number of characters to allow for each column in the summary output.
<code>round.n</code>	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

`createIdxFile` 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

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,
  show.rownames=TRUE, leading.delim=TRUE, show.all=FALSE, 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.

<code>trailer</code>	A character string to be written at the end of the output file.
<code>show.rownames</code>	Whether to display the rownames of a table.
<code>leading.delim</code>	Whether to add an extra delimiter at the beginning of the table. See Details.
<code>show.all</code>	Whether to show all the components of a list. The default FALSE is to show only those components that look like 2D tables.
<code>con</code>	A connection to which the output will be written. If a filename is passed, it will be ignored if <code>con</code> is not missing.
<code>open.con</code>	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. 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 comma 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.

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
delim.table(testglm,"testglm.html",delim="<td>",tabegin="<table border=1>",
  bor="<tr><td>",tablend="</table>",header="<html><body>",
  trailer="</body></html>")
# to clean up, delete the files "testglm.csv" and "testglm.html"
```

 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,
  maxfac=8,show.pc=TRUE)
```

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).
maxfac	The maximum number of factors for <code>describe.factor</code> to display.
show.pc	Whether to show percentages of factor and logical values.

Details

`describe` displays a table of descriptive statistics for numeric and factor 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.

Value

A list with three components:

Numeric	A matrix of the values returned from <code>describe.numeric</code> for each column described.
Factor	A matrix of the modal value and number of missing values for each column described.
Logical	A matrix of the counts of FALSE, TRUE and missing values 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="", vname.space=10, maxfac=10, show.pc=TRUE)
```

Arguments

<code>x</code>	A factor.
<code>varname</code>	A name for the variable. <code>describe</code> always passes the name.
<code>vname.space</code>	The width of the variable name display.
<code>maxfac</code>	The maximum number of factors to display.
<code>show.pc</code>	Whether to show the percentages for each level.

Details

`describe.factor` displays the name of the factor, a table of its values with up to `maxfac` entries, the modal value of the factor, the number of valid (not NA) values and the number of values not displayed if it contained more than `maxfac` levels.

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="", vname.space=10, show.pc=TRUE)
```

Arguments

<code>x</code>	A logical.
<code>varname</code>	An optional name for the variable. <code>describe</code> always passes the name of the variable.
<code>vname.space</code>	The width of the variable name display.
<code>show.pc</code>	Whether to show the percentage of TRUE values.

Details

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

Value

nil

Author(s)

Jim Lemon

See Also

[table](#)

`describe.numeric` *Description of numeric variables*

Description

Describes a numeric variable.

Usage

```
describe.numeric(x, num.desc=c("mean", "median", "var", "sd", "valid.n"),  
varname="", vname.space=10, fname.space=10)
```

Arguments

<code>x</code>	A numeric vector.
<code>num.desc</code>	The names of the functions to apply to the vector.
<code>varname</code>	The variable name to display.
<code>vname.space</code>	The width of the variable name display.
<code>fname.space</code>	The width of the descriptive columns.

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

<code>con</code>	The connection for writing to the HTML file.
<code>ending</code>	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)
```

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.

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

<code>listfile</code>	The name of the HTMLize listing file.
<code>graphfile</code>	The name for the graphic file (see Details).
<code>type</code>	The graphic format in which to write the image.
<code>...</code>	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

<code>Rfile</code>	The R script file from which to read the commands.
<code>HTMLbase</code>	The base name for the HTML files (see Details).
<code>HTMLdir</code>	The directory in which to write the HTML output.
<code>title</code>	The title of the HTML page and the headings for the frames. See Details for including the title in the R script.
<code>bgcolor</code>	The background color for the frames.
<code>echo</code>	Whether to include ("echo") the commands in the listing.
<code>do.nav</code>	Whether to have a navigation window.
<code>useCSS</code>	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.
<code>...</code>	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 displayed 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 Phillippe 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)
```

Arguments

<code>x</code>	An object, usually a factor.
<code>na.rm</code>	A dummy argument to make it compatible with calls to <code>mean</code> , 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 character.

Note

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

Author(s)

Jim Lemon

See Also

[describe](#)

print.freq	<i>Display frequency table(s)</i>
------------	-----------------------------------

Description

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

Usage

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

Arguments

<code>x</code>	a frequency table produced by <code>freq</code>
<code>show.pc</code>	whether to display percentages as well as counts.
<code>...</code>	additional arguments passed to <code>print</code> .

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.

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)
```

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=FALSE, phi=FALSE, ...)
```

Arguments

x	The list returned by <code>calculate.xtab</code> .
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 <code>chisq.test</code> and display the result.
phi	Whether to calculate and display the phi coefficient (only for 2x2 tables).
...	additional arguments passed to <code>chisq.test</code> .

Details

`print.xtab` displays a crosstabulation in a fairly conventional style with row, column and marginal percentages.

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 relative risk of the column variable given the <u>second</u> 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="#d4d4d4", ...)
```

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 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 displayed at the left side of the HTML output as a "navigation" list using the commands as names. The third file is named by concatenating `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 testR2html_list.html hista.png")
```

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**

<code>navcon</code>	The connection for writing to the navigation file.
<code>title</code>	Title for the navigation window.
<code>bgcolor</code>	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

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 codes to NA.

Value

The object with NAs replacing all specified values.

Author(s)

Jim Lemon

See Also

[%in%](#)

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

Description

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

Usage

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

Arguments

`x` An object.
`na.rm` A dummy argument to make it compatible with calls to `mean`, etc.

Details

`valid.n` finds the number of valid values of the object.

Value

The number of valid values 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)
```

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

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`.

Value

The result of `calculate.xtab` if there is only one table to display, otherwise `nil`.

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)
```

Index

*Topic **misc**

add.value.labels, 1
AddNav, 2
AddNavItem, 3
BeginNav, 3
brkdn, 4
calculate.xtab, 5
CreateIndexFile, 6
createIndxFile, 7
delim.table, 7
describe, 9
describe.factor, 11
describe.logical, 12
describe.numeric, 13
EndHTML, 14
freq, 14
HTMLgraph, 15
htmlize, 16
Mode, 18
print.freq, 19
print.xtab, 20
R2html, 21
StartList, 22
StartNav, 23
toNA, 24
valid.n, 24
xtab, 25

%in%, 24

add.value.labels, 1
AddNav, 2
AddNavItem, 3

BeginNav, 3
brkdn, 4

calculate.xtab, 5, 21, 26
CreateIndexFile, 6
createIndxFile, 7

delim.table, 7

describe, 9, 13, 19, 25
describe.factor, 10, 11
describe.logical, 12
describe.numeric, 5, 10, 13

EndHTML, 14

freq, 14, 19, 20

HTMLgraph, 15
htmlize, 9, 16

Mode, 10, 11, 18

print.freq, 15, 19
print.xtab, 6, 20, 26

R2html, 21

sink, 21
StartList, 22
StartNav, 23

table, 6, 11, 12, 26
toNA, 24

valid.n, 10, 11, 13, 24

xtab, 21, 25