Package 'BrailleR'

July 2, 2014

Type Package

Title Improved access for blind useRs

Version 0.9

Date 2014-06-27

Author A. Jonathan R. Godfrey [aut, cre]

Maintainer A. Jonathan R. Godfrey <a.j.godfrey@massey.ac.nz>

Description Blind useRs do not have access to the graphical output from R without printing the content of graph windows to an embosser of some kind. This is not as immediate as is required for efficient access to statistical output. The functions here are created so that blind people can make even better use of R.

License GPL-2

Depends R (>= 3.0.0)

Suggests knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2014-06-27 14:08:39

R topics documented:

BrailleR-package *Improved access for blind useRs*

Description

Blind useRs do not have access to the graphical output from R without printing the content of graph windows to an embosser of some kind. This is not as immediate as is required for efficient access to statistical output.

Various functions are also being developed to make text output (that is visually appealing) more useful for a blind useR who is reliant on synthesized speech or braille output to interpret the results.

Ultimately, the functions here are created so that blind people can make even better use of R.

Details

Package:	BrailleR
Type:	Package
Version:	0.9
Date:	2014-06-27
License:	GPL-2

Author(s)

Jonathan Godfrey

Maintainer: Jonathan Godfrey <a.j.godfrey@massey.ac.nz>

References

Godfrey, A.J.R. (2013) 'Statistical Software from a Blind Person's Perspective: R is the Best, but we can make it better', The R Journal 5(1), pp73-80.

BRL *test function*

Description

work in progress

Usage

```
BRL(x, ...)
```

Arguments

x	some R object
...	other parameters

Author(s)

Jonathan Godfrey

R2txtJG *Save a transcript of commands and/or output to a text file.*

Description

These functions save a transcript of your commands and their output to a script file.

They work as combinations of `sink` and `history` with a couple of extra bells and whistles.

Usage

```
txtStart(file, commands=TRUE, results=TRUE, append=FALSE, cmdfile,
          visible.only=TRUE)

txtOut(Filename=NULL)

txtStop()

txtComment(txt,cmdtxt)

txtSkip(expr)
```

Arguments

file	Text file to save transcript in
Filename	A filename to be given for the `txtOut` command. If this is not specified, the user will be prompted for a filename. If the user presses the enter key, a filename will be automatically generated that is based on the current date and time.
commands	Logical, should the commands be echoed to the transcript file
results	Logical, should the results be saved in the transcript file
append	Logical, should we append to `file` or replace it
cmdfile	A filename to store commands such that it can be `source`d or copied and pasted from
visible.only	Should non-printed output be included, not currently implemented.
txt	Text of a comment to be inserted into `file`
cmdtxt	Text of a comment to be inserted into `cmdfile`
expr	An expression to be executed without being included in `file` or `cmdfile`

Details

These functions are used to create transcript/command files of your R session. In the original TeachingDemos package from which the functions were obtained, there are 3 sets of functions. Those starting with "txt",those starting with "etxt", and those starting with wdtxt.

The "txt" functions create a plain text transcript while the "etxt" functions create a text file with extra escapes and commands so that it can be post processed with enscript (an external program) to create a postscript file and can include graphics as well. The postscript file can be converted to pdf or other format file. The "wdtxt" functions will insert the commands and results into a Microsoft Word document.

Users wishing to have the additional functionality that the "etxt" and "wdtxt" functions provide are advised to make use of the TeachingDemos package.

If `results` is TRUE and `commands` is FALSE then the result is similar to the results of `sink`. If `commands` is true as well then the results will show both the commands and results similar to the output on the screen. If both `commands` and `results` are FALSE then pretty much the only thing these functions will accomplish is to waste some computing time.

If `cmdfile` is specified then an additional file is created with the commands used (similar to the `history` command), this file can be used with `source` or copied and pasted to the terminal.

The Start function specifies the file/directory to create and starts the transcript, The prompts are changed to remind you that the commands/results are being copied to the transcript. The Stop function stops the recording and resets the prompts.

The txtOut function is a short cut for the txtStart command that uses the current date and time in the filenames for the transcript and command files. This function is not part of the TeachingDemos package.

The R parser strips comments and does some reformating so the transcript file may not match exactly with the terminal output. Use the `txtComment` functions to add a comment. This will show up as a line offset by whitespace in the transcript file. If `cmdtxt` is specified then that line will be inserted into `cmdfile` preceeded by a \# so it will be skipped if sourced or copied.

The `txtSkip` function will run the code in `expr` but will not include the commands or results in the transcript file (this can be used for side computations, or requests for help, etc.).

Value

Most of these commands do not return anything of use. The exception is:

`txtSkip` returns the value of `expr`.

Note

These commands do not do any fancy formatting of output, just what you see in the regular terminal window. If you want more formatted output then you should look into `Sweave` or the R2HTML package.

Do not use these functions in combination with R2HTML or `sink`.

Author(s)

Greg Snow, <greg.snow@imail.org> is the original author, but Jonathan Godfrey <a.j.godfrey@massey.ac.nz> is responsible for the implementation in the BrailleR package (including the txtOut function), and should therefore be your first point of contact with any problems. If you find the functions useful, you may wish to send a vote of thanks in Greg's direction.

See Also

sink, history, Sweave, the odfWeave package, the R2HTML package, the R2wd package

Examples

```
## Not run:
txtStart()
txtComment('This is todays transcript')
date()
x <- rnorm(25)
summary(x)
stem(x)
txtSkip(?hist)
hist(x)
Sys.Date()
Sys.time()

## End(Not run)
```

---

unfinished *Unfinished Methods to help vision impaired useRs*

Description

A set of methods that will (once coded) extract the most relevant information from a graphical object (or implied set of graphical objects) and display the interpreted results in text form.

The method includes representations of summary methods that are more suitable for blind useRs. For example, the method for a data.frame uses a single line for each variable instead of the normal column layout used by the summary method.

Details

This is the help page for the VI() functions that are not fully functional or below par in some way.

Value

This will vary according to the needs of vision impaired useRs and the specific objects that need to be interpretted.

In general, the output is a series of text strings printed in the console/terminal window in addition to the embedded command's normal functionality.

These functions do not create objects as do many R commands. Manipulations on the objects created by regular R expressions will need those regular expressions issued in addition to those of the VI family of functions.

Author(s)

Jonathan Godfrey

`VI` *Methods to help vision impaired useRs*

Description

A set of methods that extract the most relevant information from a graphical object (or implied set of graphical objects) and display the interpreted results in text form.

The method includes representations of summary methods that are more suitable for blind useRs. For example, the method for a data.frame uses a single line for each variable instead of the normal column layout used by the `summary` method.

Usage

```
VI(x)

## S3 method for class 'histogram'
VI(x)
```

Arguments

`x`	any R object

Details

This is the general help page for the VI() functionality. Some specific pages exist where some ability to alter the outcome through user input have warranted their own help pages. See below for more detail on these.

Further methods can be written by useRs. Please submit to the package maintainer for possible inclusion in subsequent releases of the package.

Value

This will vary according to the needs of vision impaired useRs and the specific objects that need to be interpretted.

In general, the output is a series of text strings printed in the console/terminal window in addition to the embedded command's normal functionality.

These functions do not create objects as do many R commands. Manipulations on the objects created by regular R expressions will need those regular expressions issued in addition to those of the VI family of functions.

Author(s)

Jonathan Godfrey

Examples

```
RandomX=rnorm(500)
PlottedFig=hist(RandomX)
rm(RandomX)
VI(PlottedFig)
rm(PlottedFig)
```

WhereXY *Count points in a scatter plot*

Description

count the number of points that fall into various sized subparts of a scatter plot. The graphing region can be split into cells based on a uniform or normal marginal distribution for both x and y variables.

Usage

```
WhereXY(x, y = NULL, grid = c(3, 3), Dist = "uniform")
```

Arguments

x,y	vectors of x coordinates. If y is not specified, the function expects x to be a two-column matrix with x and y values in columns 1 and 2 respectively.
grid	pair of values to specify the way the graph is to be split into parts. Specify x and then y.
Dist	the distribution the variables might be expected to follow. The default is to consider uniformly distributed but any alternative text will lead ot an assumption of both margins being normally distributed.

Value

A text descriptionof the number of points in each subregion of the scatter plot. The table of counts can then be compared to the expected number of points in each subregion.

Author(s)

Jonathan Godfrey

Examples

```
x=rnorm(50)
y=rnorm(50)
WhereXY(x,y)
WhereXY(x,y, c(3,4))
WhereXY(x,y, Dist="other")
```

Index