# Package 'BrailleR'

June 26, 2015

**Type** Package

**Title** Improved Access for Blind Users

**Version** 0.22.0

**Date** 2015-06-26

**Author** A. Jonathan R. Godfrey [aut, cre], Greg Snow [ctb], James Curtis [ctb], Paul Murrell [ctb], Timothy Bilton [ctb], Yihui Xie [ctb]

**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. This includes the text descriptions of graphs, convenience functions to replace the functionality offered in many GUI front ends, and experimental functionality for optimising graphical content to prepare it for embossing as tactile images.

**Repository** CRAN

**License** GPL-2

**Depends** R (>= 3.0.0), knitr

**Imports** devtools, extrafont, gridGraphics, gridSVG, moments, nortest, rmarkdown, xtable

**VignetteBuilder** knitr

**NeedsCompilation** no

**Date/Publication** 2015-06-26 14:00:31

## 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. The functions here are created so that blind people can make even better use of R. This includes the text descriptions of graphs, convenience functions to replace the functionality offered in many GUI front ends, and experimental functionality for optimising graphical content to prepare it for embossing as tactile images.

## Details

| | |
|---|---|
| Package: | BrailleR |
| Type: | Package |
| Version: | 0.22.0 |
| Date: | 2015-06-26 |
| License: | GPL-2 |

## Author(s)

A. Jonathan R. Godfrey and Timothy P. Bilton; with other contributions.

Maintainer: A. Jonathan R. 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), pp73-79.

---

`boxplot` *Create a standard boxplot with a few extra elements added to the output object*

---

## Description

This function is a wrapper to the standard `boxplot()` function in the **graphics** package. It adds detail to the stored object so that a better text description can be formulated using the `VI()` method in the **BrailleR** package.

## Usage

```
boxplot(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | a numeric variable. |
| `...` | additional arguments passed on to the plotting function. |

## Details

This function masks the function of the same name in the **graphics** package. The base R implementation does create an object, but does not give it a class attribute, the object does not store all graphical arguments that are passed to the `boxplot()` function. The functionality should be no different at all for anyone who is not using the `VI()` function to gain a more detailed text description of the boxplot. See the help page for the `graphics::boxplot()` function to get a more complete description of boxplot creation.

## Value

An object of class boxplot.

## Note

I would love to see this function become redundant. This will happen if the extra functionality is included in the `boxplot()` function in the **graphics** package. This should be possible as the user experience will not be any different, no matter if the user is blind or sighted.

## Author(s)

A. Jonathan R. Godfrey

## References

The problem of not including class attributes for graphs was identified in: 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.

## See Also

The base R implementation of the `boxplot` function should be consulted; see the entry in the **graphics** package

## Examples

```
x=rnorm(1000)
# the stamdard boxplot function returns
MyBoxplot=graphics::boxplot(x, main="Example boxplot (graphics package)")
#dev.off()
MyBoxplot

# while this version returns
MyBoxplot=boxplot(x, main="Example boxplot (BrailleR package)")
#dev.off()
MyBoxplot

# The VI() method then uses the extra information stored
VI(MyBoxplot)
```

---

# BRLThis *Convert a graph to a pdf ready for embossing*

---

## Description

The first argument to this function must be a call to create a graph, such as a histogram. Instead of opening a new graphics device, the graph will be created in a pdf file, with all text being presented using a braille font. The function is somewhat experimental as the best braille font is not yet confirmed, and a number of examples need to be tested on a variety of embossers before full confidence in the function is given.

## Usage

```
BRLThis(x, file)
```

## Arguments

| | |
|---|---|
| `x` | the call to create a graph |
| `file` | A character string giving the filename where the image is to be saved. |

## Details

The user's chosen braille font must be installed. This might include the default font shipped as part of the package.

## Value

Nothing within the R session, but a pdf file will be created in the user's working directory.

## Author(s)

A. Jonathan R. Godfrey.

## Examples

```
with(airquality,
BRLThis(hist(Ozone), "Ozone-hist.pdf"))
```

---

# DataViewer *Open a data object in your chosen spreadsheet software*

---

## Description

The chosen data object (data.frame, matrix, or vector) is stored in a temporary csv file. The file is opened while the R terminal/console is suspended until the <enter> key is pressed. This should be done once the spreadsheet software has been closed so that the temporary file is removed and the data object gets updated from the edited csv file. The user must save the csv file if changes are made.

## Usage

```
DataViewer(x, Update = FALSE, New = NULL, Filename = NULL)
```

## Arguments

| | |
|---|---|
| x | an object of class data.frame, matrix, or vector. A check is made for this condition. |
| Update | Logical. Will changes made in the spreadsheet editor be returned to the R session. |
| New | If Update=TRUE, the original object will be replaced unless a name for a new object is given. N.B. the default is to overwrite the original object. |
| Filename | Provide a filename if desired. N.B. this does not guarantee that the csv file will be kept. Do not use this function as a shortcut for saving csv files. |

## Details

The following steps should be taken if the intention is to view and possibly edit a data.frame, matrix, or vector:

There are a few quirks with respect to variables that could be interpreted as factors. The process of updating is to write a csv file using code write.csv() and then read it back using code read.csv() after it has been edited. Users must be aware that factors may not register as such, or that character vectors may be interpreted as factors.

## Value

If code Update=TRUE and code New=NULL, the original data object is overwritten. To create a new object the user mustspecify code New.

## Author(s)

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

## Examples

```
data(airquality)
DataViewer(airquality, Update=TRUE, New="NewAirQuality")
# remove the new object using rm()
```

---

# dotplot *create a dotplot using stripchart*

---

## Description

A method for creating dotplots. The functions call the `stripchart` command from the **graphics** package and assign the output to have the class `dotplot`.

## Usage

```
dotplot(x, ...)

## S3 method for class 'formula'
dotplot(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | a vector or formula, where the right hand side of the formula is a factor. |
| `...` | other graphical parameters including those passed to `title`. |

## Details

This function was created as a result of being unable to assign all graphical parameters that are created when a formula is used in `stripchart`. Users not intending to use the `VI` method should use `stripchart` instead.

### Author(s)

A. Jonathan R. Godfrey

### See Also

This function is dependent on the `stripchart` function from the **graphics** package. Consult its help page for more information.

### Examples

```
VI(with(airquality, dotplot(Ozone~Month)))
```

---

## GetGoing *Set options for using brailleR*

---

### Description

An interactive question-and-answer interface suitable for blind users wanting to set the options for using the BrailleR package.

### Usage

```
GetGoing()
```

### Details

Defaults are offered for all questions so that pressing <Enter> means no changes are made. Users answer yes/no questions as TRUE or FALSE respectively; the short form T or F is also allowed.

The user can also choose to perform various setup tasks using this interface.

### Value

NULL. This function is a tool for executing other functions that will set options and setup the package according to the user's wants.

### Author(s)

A. Jonathan R. Godfrey

### See Also

All options being set through this function have specific functions that achieve the same ends. For example, see `GoSighted` for the options that are binary settings, or `SetAuthor` for options requiring a specific character or numeric value to be chosen.

The setup functionality can be reviewed at `MakeBatch`.

`hist` *Create a standard histogram with a few extra elements added to the output object*

## Description

This function is a wrapper to the standard `hist()` function in the **graphics** package. It adds detail to the stored object so that a better text description can be formulated using the `VI()` method in the **BrailleR** package.

## Usage

```
hist(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | a numeric variable. |
| `...` | additional arguments passed on to the plotting function. |

## Details

This function masks the function of the same name in the **graphics** package. Even though the base R implementation does create an object of class histogram, the object does not store all graphical arguments that are passed to the `hist()` function. The functionality should be no different at all for anyone who is not using the `VI()` function to gain a more detailed text description of the histogram. See the help page for the graphics::hist() function to get a more complete description of histogram creation.

## Value

An object of class histogram as per the `hist()` function from the **graphics** package, with the addition of any calls to the main title or axis labels.

## Author(s)

A. Jonathan R. Godfrey

## 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.

## See Also

The base R implementation of the `hist` function should be consulted, using the entry in the **graphics** package

## Examples

```
x=rnorm(1000)
# the stamdard hist function returns
MyHist=graphics::hist(x, xlab="random normal values", main="Example histogram (graphics package)")
#dev.off()
MyHist

# while this version returns
MyHist=hist(x, xlab="random normal values", main="Example histogram (BrailleR package)")
#dev.off()
MyHist

# The VI() method then uses the extra information stored
VI(MyHist)
```

---

Internal *Internal functions for the BrailleR package*

---

## Description

Some functions that probably weren't really necessary, but proved useful at some stage.

## Usage

```
GetAxisTicks(x)

InQuotes(x)
```

## Arguments

| | |
|---|---|
| x | an object that is of the form needed by the internal function concerned. |

## Details

`GetAxisTicks` is intended to create the sequence of values used on an axis.

## Value

These should be fairly obvious from the name of the function

## Author(s)

A. Jonathan R. Godfrey

`MakeBatch` *Create batch files for processing R scripts and markdown files under Windows*

## Description

Convenience function for creating batch files that can be used under Windows to process R scripts and Rmarkdown files. It may also be possible to use them in conjunction with the Open With dialogue in Windows Explorer; this makes use of file associations can be established so that these files are (in effect) executable.

## Usage

```
MakeBatch(file=NULL)
```

## Arguments

| | |
|---|---|
| `file` | A character string for the file to be processed. The file need not yet exist. The extensionn must be either R or Rmd and is case sensitive. |

## Details

These batch files are not for use in an interactive R session. They need to be created in an interactive session, so that users can process R scripts and Rmarkdown files without needing to open an R session later.

If a file is specified, the function will create a single batch file that will process the file appropriately. Processing an R script will generate a Rout file, while an Rmd file is converted into HTML.

If no file is specified, the function creates various files in the current working directory.

Files starting with the word test are for testing the batch files. An R script and an Rmarkdown file were created as well as the batch files that will process them into a Rout file and an HTML document respectively. Pressing <enter> on these test*.bat files will process the test files appropriately.

The other two batch files (ending in .bat) need to be moved to a folder on the user's path so that they can be called from anywhere. They could also be manually edited ot suit the user's needs.

The path.txt file shows the user what folders are already on the path list. The user can review this list and decide to alter the system variable if they so choose. The path.txt file has no value otherwise.

## Further instructions

Once the RBatch.bat file has been moved to the desired folder that is included in the path for your system you can follow the following steps to get this function working fully.

1. Open windows explorer, and browse to the folder containing the test files.

2. Select the test.R script.

3. Under the File menu, look for the item Open with... (This might already be a submenu for some users; if so, the last item is Choose default program.)

4. We are going to choose to use a program on our computer. Do not go looking on the internet to see which program we need.

5. You may be able to write a description of the file type. This is an R script but it may not yet be registered as such. Providing this detail is optional.

6. When given the chance to browse for the program to open the test.R script, browse to the folder where you placed Rbatch.bat and select it.

7. When you select OK, the test.R script will be processed by RBatch.bat and a new file test.Rout will be created.

8. Open test.Rout in any text editor you like. This file has the appearance of an R session window except for some processing time detail at the end. You will be able to read the commands that were originally in test.R as well as the output from these commands.

## Author(s)

A. Jonathan R. Godfrey

---

`MakeRmdFiles` *Work flow convenience funcions*

---

## Description

Time-saving funcions that help create files in more useful formats for later processing.

## Usage

```
History2Rmd(file = "History.Rmd")

R2Rmd(ScriptFile)
```

## Arguments

| | |
|---|---|
| `file` | the name of the fil to be created. |
| `ScriptFile` | the R script to be processed into the Rmarkdown file. |

## Details

The History2Rmd function was intended for turning a short interactive R session into an Rmarkdown file. Lines of code are all separated into distinct code chunks in the Rmd file. the resulting file will need to be edited if commands have spanned more than one line.

The R2Rmd function does try to limit the number of blank lines copied from the R script into the Rmarkdown file. The Rmd file may need som editing.

## Value

NULL. These functions are for creating files in the current working directory.

### Author(s)

A. Jonathan R. Godfrey

### See Also

These functions were inspired by the spin functionality of the **knitr** package. You may wish to move onto using it for more features.

---

## MakeRprofile *Load BrailleR on Startup in Current Working Directory*

---

### Description

Writes the single command "library(BrailleR)" to a .First() function in .Rprofile in the current working directory. This forces the BrailleR package to be automatically loaded when R is opened in this working directory.

### Usage

```
MakeRprofile(Overwrite = FALSE)
```

### Arguments

| | |
|---|---|
| Overwrite | Logical: Should an existing .Rprofile file be overwritten? |

### Value

Nothing. This function is used for its side effect of creation of a file in the current working directory. A warning message is created if the file exists and Overwrite=FALSE.

### Author(s)

A. Jonathan R. Godfrey.

---

## OneFactor *Analysis for a continuous response for one group factor*

---

### Description

A convenience funcion that creates an analysis for a continuous response variable with one grouping factor. The function creates a number of graphs and tables relevant for the analysis.

## Usage

```
OneFactor(Response, Factor, Data = NULL, HSD = TRUE, AlphaE = 0.05,
Filename = NULL, Folder = NULL,
VI = getOption("BrailleR.VI"), Latex = getOption("BrailleR.Latex"),
View = getOption("BrailleR.View"))
```

## Arguments

| | |
|---|---|
| Response | Name of the continuous response variable. |
| Factor | The grouping factor. |
| Data | The data.frame that contains both the response and the factor. |
| HSD | Logical: Should Tukey's HSD be evaluated for the data? |
| AlphaE | The family-wise Type I error rate for Tukey's HSD calculations. |
| Filename | Name of the Rmarkdown and HTML files to be created. A default will be created that uses the names of the variables if this is left set to NULL. |
| Folder | Name of the folder to store graph and LaTeX files. A default will be created based on the name of the data.frame being used. |
| VI | Logical: Should the VI method for blind users be employed? |
| Latex | Logical: Should the tabulated sections be saved in LaTeX format? |
| View | Logical: Should the HTML file be opened for inspection? |

## Details

This function writes an R markdown file that is knitted into HTML and purled into an R script. All graphs are saved in subdirectories in png, eps, pdf and svg formats. Tabulated results are stored in files suitable for importing into LaTeX documents.

## Value

This function is used for creation of the files saved in the working directory and a few of its subdirectories.

## Author(s)

A. Jonathan R. Godfrey and Timothy P. Bilton

## See Also

Other convenience functions can be investigated via their help pages. See UniDesc, OnePredictor, and TwoFactors

## Examples

```
data(airquality)

# the following line  returns an error:
## OneFactor("Ozone", "Month", airquality, View=FALSE)
# so we make a copy of the data.frame, and fix that:

airquality2 = airquality
airquality2$Month = as.factor(airquality$Month)
# and now all is good to try:
OneFactor("Ozone", "Month", airquality2)
# N.B. Various files and a folder were created in the working directory.
# Please investigate them to see how this function worked.
```

---

`OnePredictor` *Exploration of the relationship between a response and a single predictor*

---

## Description

A conveience function for generating exploratory graphs and numeric summaries, regression analysis output, and the residual analysis of the simple linear model.

## Usage

```
OnePredictor(Response, Predictor, Data = NULL,
    Filename = NULL, Folder = NULL,
    VI = getOption("BrailleR.VI"), Latex = getOption("BrailleR.Latex"),
   View = getOption("BrailleR.View"))
```

## Arguments

| | |
|---|---|
| `Response` | Name of the continuous response variable. |
| `Predictor` | Name of the continuous response variable. |
| `Data` | The data.frame that contains both the response and the factor. |
| `Filename` | Name of the Rmarkdown and HTML files to be created. A default will be created that uses the names of the variables if this is left set to NULL. |
| `Folder` | Name of the folder to store graph and LaTeX files. A default will be created based on the name of the data.frame being used. |
| `VI` | Logical: Should the VI method for blind users be employed? |
| `Latex` | Logical: Should the tabulated sections be saved in LaTeX format? |
| `View` | Logical: Should the HTML file be opened for inspection? |

### Details

This function writes an R markdown file that is knitted into HTML and purled into an R script. All graphs are saved in subdirectories in png, eps, pdf and svg formats. Tabulated results are stored in files suitable for importing into LaTeX documents.

### Value

This function is used for creation of the files saved in the working directory and a few of its subdirectories.

### Author(s)

A. Jonathan R. Godfrey and Timothy P. Bilton

### See Also

Other convenience functions can be investigated via their help pages. See UniDesc, OneFactor, and TwoFactors

### Examples

```
data(airquality)
OnePredictor("Ozone", "Wind", airquality, View=FALSE)
# look at the new files and folders created in the current working directory.
```

---

## Options *Set package options*

---

### Description

A set of convenience functions to alter the settings that control how much output is generated and displayed by other **BrailleR** functions.

### Usage

```
GoBlind()
GoSighted()

LatexOn()
LatexOff()

ViewOn()
ViewOff()
```

## Details

The function names should be fairly self explanatory. `GoBlind()` and `GoSighted()` control use of the `VI()` method which provides extra infromation about graphical objects for the assistance of blind users; `ViewOn()` and `ViewOff()` are for the automatic opening of `HTML` pages created by **BrailleR** functions; and `LatexOn()` and `LatexOff()` control the production of tables into LaTeX via the **xtable** package.

## Value

Nothing is returned. These functions are only used for their side effects.

## Author(s)

A. Jonathan R. Godfrey

## See Also

See these settings applied as default arguments to `UniDesc`

---

# Premiere100 *Prepare BrailleR settings for specific braille embossers*

---

## Description

Convenience functions for setting package options based on experimentation using specific embossers.

## Usage

```
Premiere100()
```

## Details

These functions are are only relevant for owners of the specified embossers. Ownership of these models means the user has access to fonts that are licenced to the user.

The Premiere 100 embosser uses standard 11 by 11.5 inch fanfold braille paper. Printing in landscape or portrait is possible.

## Value

Nothing. The functions are only used to set package options.

## Author(s)

A. Jonathan R. Godfrey.

## See Also

ChooseEmbosser

## Examples

```
Premiere100() # Specify use of the Premiere 100 embosser.
ChooseEmbosser() # reset to default: using no embosser.
```

---

PrepareWriteR *Getting started with WriteR*

---

## Description

The WriteR application was written in wxPython so that blind users can process Rmarkdown documents. This functionality is because RStudio is not an accessible application for screen reader users.

## Usage

```
PrepareWriteR(Author = getOption("BrailleR.Author"))
```

## Arguments

| | |
|---|---|
| Author | Your name as you want it to appear in the default text that starts your Rmarkdown documents. |

## Details

This function writes the settings file (called WriteR.init) for WriteR and copies the files that were part of the BrailleR installation into the current working directory. You will be able to run the WriteR application from there, or move to a folder of your choosing.

## Value

NULL. This function is for your convenience and not for doing any work inside an R session.

## Note

You must have Python 2.7 and the associated wxPython installation on your system to use the WriteR application.

## Author(s)

A. Jonathan R. 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 use of markdown documents..

Do not use these functions in combination with the **R2HTML** package 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)
```

---

SetOptions *Functions for setting package options.*

---

## Description

Some package options have arguments which need validation. Setting the default significance level for analyses was the first function of this kind. Setting the name of the user to be inserted into documents was the second. Others are detailed further below.

## Usage

```
ChooseEmbosser(Embosser = "none", Permanent = TRUE)

ChooseStyle(css = "BrailleR.css", Permanent = TRUE)

ResetDefaults()

SetAuthor(name = "BrailleR", Permanent = TRUE)

SetBRLPointSize(pt, Permanent=FALSE)
SetPaperHeight(Inches, Permanent=FALSE)
SetPaperWidth(Inches, Permanent=FALSE)

SetPValDigits(digits, Permanent = TRUE)

SetSigLevel(alpha, Permanent = TRUE)
```

## Arguments

| | |
|---|---|
| alpha | The level of alpha to be used for analyses. Must be between zero and one or a warning is given and the option is not changed. |

| | |
|---|---|
| css | a cascading style sheet file to be inserted in HTML documents created by convenience functions. The file must be placed in the css folder within the BrailleR package folder for this to work. |
| digits | The number of decimal places to display. Must be an integer greater than one or a warning is given and the option is not changed. |
| Embosser | the name of the embosser to be used for tactile images. Not all embossers will be immediately supported by the package. The supported embossers are listed in the relevant section below. Please contact the package maintainer to introduce a new embosser to the list of supported models. |
| Inches | The size of the area to use for embossing. This should be the size of the embossed area not the actual size of the paper itself. |
| name | a character string to be used for author details in various file writing functions. |
| Permanent | Should the change be made permanent? Set to FALSE for a temporary change. |
| pt | The point size of the chosen braille font. |

## Details

More convenience functions for BrailleR users. Most are self explanatory, but the following details should be noted.

The Choose...() functions are used for establishing default parameters for other details. The ChooseStyle() command can be used to alter the appearance of HTML output by way of cascading style sheets. You can create your own css file and add it to the package before calling this function.

The ChooseEmbosser() will look for the default settings recommended for particular types of embosser. Initial testing was done on a Tiger Premiere 100 embosser manufactured by ViewPlus Inc. The default paper size is 11 by 11.5 inches, but the recommended embossing area for graphics is 10 by 10 inches.

The Set..() commands will let the user specify any desired value for the options as long as it is valid: Options assumed to be character strings are checked to be so, integers must be integers and a proportion must be between zero and one.

SetPaperHeight and SetPaperWidth are temporary changes by default because some types of images are not meant to use the maximum area set down by the original default settings for an embosser. Careful experimentation may be required to get optimal results. If permanent changes are desired, then please contact the package maintainer to explain why you have made these changes so that we can help other users get the best from a wide range of embossers.

SetPValDigits() is used for rounding purposes to avoid the use of scientific notation. It is not used for determining significance.

## Author(s)

A. Jonathan R. Godfrey

## Examples

```
# SetSigLevel(5) # not a valid alpha
SetSigLevel(0.05) # valid alpha value
SetAuthor()
SetAuthor("Jonathan Godfrey")
```

SVGThis *Save commonly used graphs as structured SVG files.*

## Description

Converts a graph object (as long as it has a class assigned) or the current graph window to an SVG file that can be viewed using the Tiger Player software availabel from ViewPlus Inc. At present, the SVG needs manual editing using the Tiger Transformer software before viewing.

## Usage

```
SVGThis(x, file = "test.svg")
```

## Arguments

| | |
|---|---|
| x | a graph object for which a method exists, or the current graphics device if set to NULL. |
| file | The SVG file to be created. |

## Details

The Cairo SVG device found in the gr.devices package does not create a structured SVG file that includes the semantics of the graphic being displayed. The SVG created by the gridSVG package does meet this need, but only works on graphs drawn using the grid package. Any graph created using functions from the more common graphics package can be converted to the grid package system using the gridGraphics package.

## Value

NULL. This function is solely for the purpose of creating SVG files in thne current working directory or in a path of the user's choosing.

## Author(s)

A. Jonathan R. Godfrey and Paul Murrell

## References

Need Murrell for gridGraphics and Murrell and Potter for gridSVG.

## Examples

```
x=rnorm(1000)
#SVGThis(hist(x))
```

TwoFactors *A convenience function for a two-way analysis*

## Description

Prepares an analysis of a data set with one response and two predictors that are both factors. An interaction between the two factors is also allowed for. The function creates a number of graphs and tables relevant for the analysis.

## Usage

```
TwoFactors(Response, Factor1, Factor2, HSD = TRUE, AlphaE = 0.05,
   Data = NULL, Inter = FALSE, Filename = NULL, Folder = NULL,
    VI = getOption("BrailleR.VI"), Latex = getOption("BrailleR.Latex"),
    View = getOption("BrailleR.View"))
```

## Arguments

| | |
|---|---|
| Response | Name of the continuous response variable. |
| Factor1, Factor2 | Name the two factors to be included. |
| HSD | Logical: Should Tukey's HSD be evaluated for the data? |
| AlphaE | The family-wise Type I error rate for Tukey's HSD calculations. |
| Data | Name the data.frame that includes the three variables of interest. |
| Inter | Logical: Should the interaction of hte two factors be included? |
| Filename | Name of the Rmarkdown and HTML files to be created. A default will be created that uses the names of the variables if this is left set to NULL. |
| Folder | Name of the folder to store graph and LaTeX files. A default will be created based on the name of the data.frame being used. |
| VI | Logical: Should the VI method for blind users be employed? |
| Latex | Logical: Should the tabulated sections be saved in LaTeX format? |
| View | Logical: Should the HTML file be opened for inspection? |

## Details

to complete

## Value

to complete

## Author(s)

Timothy P. Bilton and A. Jonathan R. Godfrey

## See Also

The OneFactor script was the basis for this function;.

## Examples

```
TG <- ToothGrowth
TG$dose <- as.factor(TG$dose)

# Without interaction
TwoFactors('len','supp','dose',Data=TG, Inter=FALSE)

# With two-way interaction
TwoFactors('len', 'supp', 'dose', Data=TG, Inter=TRUE)

# For unbalanced data
TG_Unb <- TG[-c(53:60),]
TwoFactors('len', 'supp', 'dose', Data=TG_Unb, Inter=TRUE)
rm(TG); rm(TG_Unb)
```

---

# 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

UniDesc *Descriptive statistics and graphs for univariate data*

## Description

This function is a convenience function for analyzing univariate data. It provides histograms, boxplots and tabulated results for normality tests as well as those for skewness and kurtosis. The intended use of this function is principally for a blind user of R who also has the advantage of retrieving textual descriptions of the graphs created along the way, via the `VI()` methods.

## Usage

```
UniDesc(Response = NULL, ResponseName = as.character(match.call()$Response),
    Basic = TRUE, Graphs = TRUE, Normality = TRUE, Tests = TRUE,
    Title = NULL, Filename = NULL, Folder = ResponseName, Process = TRUE,
VI = getOption("BrailleR.VI"), Latex = getOption("BrailleR.Latex"),
View = getOption("BrailleR.View"), PValDigits=getOption("BrailleR.PValDigits"))
```

## Arguments

| | |
|---|---|
| `Response` | The numeric vector to be analyzed. This must be specified as a variable that is directly available in the workspace, not as a data.frame$variable construct. |
| `ResponseName` | This is the same as Response but use quote marks around it. Exactly one of Response or ResponseName must be specified. |
| `Basic` | logical, asking for basic numeric summary measures |
| `Graphs` | logical, indicating if the graphs are to be created. These will be eps files suitable for insertion in LaTeX documents, pdf files for more general use, and SVG for easier use by blind users. |
| `Normality` | logical, asking if the various normality tests offered in the **nortest** package should be used |
| `Tests` | logical, should skewness and kurtosis tests be performed. |
| `Title` | the title of the R markdown document being created. NULL leads to a default string being chosen. |
| `Filename` | Specify the name of the R markdown and html files (without extensions). |
| `Folder` | the folder where results and graph files will be saved. |
| `Process` | logical, should the R markdown file be processed? |
| `VI` | logical, should the VI method be used to give added text descriptions of graphs? This is most easily set via the package options. |
| `Latex` | logical, Should the **xtable** package be used to convert the tabulated results into LaTeX tables? This is most easily set via the package options. |
| `View` | logical, should the resulting HTML file be opened in a browser? This is most easily set via the package options. |
| `PValDigits` | Integer. The number of decimal places to use for p values. This is most easily set via the package options. |

## Value

Saves an R markdown file, (and then if `Process=TRUE`) an R script file, and an html file (which may be opened automatically) in the current working folder. Graphs are saved in png, eps, pdf, and SVG formats (if requested) in (optionally) a subfolder of the current working directory.

## Author(s)

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

## Examples

```
Ozone=airquality$Ozone
UniDesc(Ozone, View=FALSE)
rm(Ozone)
```

---

# 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 or HTML 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)

## S3 method for class 'aov'
VI(x)

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

## Arguments

| | |
|---|---|
| x | any R object |

## Details

This is the general help page for the VI() functionality. Specific help pages will be created if the ability to alter the outcome through user input warrants. See below for more detail on these.

Further methods can be written by users (blind or sighted). 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. The `VI.lm()` method is the first to move away from this idea and use a process that builds on the `UniDesc()` function. In this case, the method creates an R markdown file and compiles it into HTML. The HTML document is opened if the R session is interactive.

In general, 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. The `VI.lm()` method does create objects in the current workspace and then deletes them once the HTML document is compiled.

## Author(s)

A. Jonathan R. Godfrey and Timothy P. Bilton

## 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 separately for the x and y variables.

## Usage

```
WhereXY(x, y = NULL, grid = c(4, 4), xDist = "uniform",
       yDist = xDist, addmargins=TRUE)
```

## 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. |
| xDist,yDist | the distribution the variables might be expected to follow. The default is to consider uniformly distributed but any alternative text will lead to an assumption of both margins being normally distributed. |
| addmargins | logical: should sums be added to both rows and columns. |

## 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)

A. Jonathan R. Godfrey

## Examples

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

# Index