Package ‘R2wd’

February 19, 2015

Type Package
Title Write MS-Word documents from R
Version 1.5
Date 2012-03-16
Author Christian Ritter
Maintainer Christian Ritter <R2wd@ridaco.be>
Description This package uses either the statconnDCOM server (via the rcom package) or the RDCOMClient to communicate with MS-Word via the COM interface.
Depends R (>= 2.10.0)
Suggests rcom, RDCOMClient
SystemRequirements Windows
License GPL-2
OS_type windows
LazyLoad yes
Repository CRAN
Date/Publication 2012-03-16 17:14:55
NeedsCompilation no

R topics documented:

R2wd-package ................................................. 2
wdApplyTemplate ........................................ 4
wdApplyTheme ............................................. 5
wdBody ..................................................... 6
wdConvert ............................................... 7
wdEnumerate ............................................. 8
wdEqn .................................................. 8
wdGet ................................................... 9
wdGetProperty ......................................... 10
wdGoToBookmark ........................................ 11
R2wd-package

Write MS-Word documents from R.

Description

R2wd uses a COM client to communicate with MS-Word via the COM interface (Windows only). Two COM clients are supported: "rcom" (Statconn DCOM via Rscproxy) and RDCOMClient (from the omegahat distribution). R2wd can be seen as a collection of convenience wrappers to calls of the COM client.

Details

Package: R2wd
Type: Package
Version: 1.5.0
Date: 2012-02-28
License: GPL-2
LazyLoad: yes
If Word is not already running, wdGet() opens a new Word document, otherwise, it establishes a COM handle to the instance which is already running. If the parameter "filename" is specified, wdGet will check whether this file is already open or (if not) try to open it. The functions wdTitle, wdHeader, wdType, wdBody, and wdParagraph can be used to inject text elements into Word. Moreover, bookmarks can be added via wdInsertBookmarks and wdGoToBookmark allows to navigate among the bookmarks which also exist. There is another set of convenience functions, wdSection, wdSubsection, and wdSubsubsection which insert headers of level 1, 2, or 3, start new 'Sections' in Word, and add bookmarks.

Graphs and dataframes can be inserted into Word, by the wdPlot, wdTable commands. The wdTable command takes a dataframe or an array as arguments, creates a Word table of the appropriate dimensions and injects the content of the dataframe or array into it. It then formats the table in Word using elementary formatting elements.

There are a few convenience functions for carrying out various typical functions in Word, such as undo, page setup, verbatim, etc.

The functions wdApplyTheme and wdApplyTemplate allow to work with themes and templates.

### Examples

```r
### Not run:
## rcom needs to be installed
if (!require(rcom)) warning("Install rcom first")
## This initializes the hook to Word
## if Word is already running it connects to it via StatconnDcom
## if not, it uses StatconnDcom to open word to create a new document
## and to connect to it.
wdGet()
## Then there is a set of small functions which implement elementary
## functionality:
wdTitle("R2wd, A Package for writing Word Documents from R",label="R2wd")
##
wdSection("Introduction")
##
wdBody("This is an example on how to use the R2wd package.")
wdWrite("wdWrite continues writing in the same style. If you set paragraph to FALSE, it doesn’t add a paragraph reference."
wdInsertFootnote("Footnotes have two arguments, the footnote text and the footnote reference. By default, word creates a numbered reference."
wdBody(". As it is shown here")
##
## We can also plot data. The following line uses the default plot command
##
wdPlot(1:100,sin((1:100)/10),type="l",main="a test graph")
##
## The plot command can be exchanged for something else
##
wdPlot(mtcars,plotfun=pairs,main="a test graph",height=6,width=6,pointsize=8)
##
```

Author(s)

Christian Ritter

Maintainer: Christian Ritter <R2wd@ridaco.be>
## wdApplyTemplate

### Description

Apply a Word template.

### Usage

```r
wdApplyTemplate(filename, wdapp = .Rwd)
```
wdApplyTheme

Arguments

filename The name of the template.
wdapp The handle to the Word application (see details).

Details

This function applies the given Word template. Indicating the handle to the Word application is usually not necessary (will be tacitely initialized by the wdGet function and assigned to the session variable .R2wd.

Note

This function is just a sketch on what could be done. Someone should write a few interesting templates to mirror typical LaTeX ones, such as article, report, etc.

Author(s)

Christian Ritter

wdApplyTheme Apply a word template.

Description

Apply a word template.

Usage

wdApplyTheme(theme, wdapp = .R2wd)

Arguments

theme The name of the theme.
wdapp The handle to the Word application (see details).

Details

This function applies the given Word template. Indicating the handle to the Word application is usually not necessary (will be tacitely initialized by the wdGet function and assigned to the session variable .R2wd.

Note

This function is just a sketch on what could be done.

Author(s)

Christian Ritter
Examples

```r
## Not run:  
## apply a default theme with a blue background. 
wdApplyTheme("BlueCalm 100")  
## remove the current theme 
wdApplyTheme("None")  
## End(Not run)
```

---

**wdBody**

*Write in body text.*

---

**Description**

Inserts text in 'Body' style at the current cursor point in Word.

**Usage**

```r
wdBody(text = "", paragraph = TRUE, wdapp = .R2wd)
```

**Arguments**

- `text` a test string to be written to Word.
- `paragraph` whether a paragraph (line return) should be added at the end.
- `wdapp` the handle to word (can be ignored).

**Details**

Switches to 'Body' style, adds the text, and then gives a carriage return (paragraph break) if requested.

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run: wdBody("R2wd is a package to write MS-Word files from R")
```
Description

This function converts between inches, cm, and points.

Usage

wdConvert(input, from="in", to="pt")

Arguments

input the number to convert
from either "in", "cm", or "pt".
to either "in", "cm", or "pt".

Details

Converts the input value from from units to to units.

Value

the converted number.

Author(s)

Christian Ritter

Examples

## Not run:
wdConvert(1,"in","pt")
wdConvert(72,"pt","in")
wdConvert(1,"cm","pt")
## the following gives an error:
wdConvert(1,"cm","mm")
## End(Not run)
**wdEnumerate**  
*Start a numbered list*

**Description**
This corresponds to pushing the [123] button in Word.

**Usage**

```
wdEnumerate(Gallery=2, Template=1, wdapp = .R2wd)
```

**Arguments**

- **Gallery**
  the (numbered) item in the list gallery. The pre-set Gallery=2 implies that bullets are used.
- **Template**
  the (numbered) item in the list template. The pre-set Template=1 implies that the first type is used.
- **wdapp**
  the handle to word (can be ignored).

**Details**
Starts a numbered list like pushing on the [123] button in Word.

**Author(s)**
Christian Ritter

**Examples**
```r
## Not run: wdEnumerate("R2wd is a package to write MS-Word files from R")
```

**wdEqn**  
*Write formulas in Word*

**Description**
This function allows writing formulas/equations into Word using the default formula editor in Word 2007 and 2010.

**Usage**

```
wdEqn(eqtext, bookmark = NULL, iknow=FALSE, waitsec=2, wdapp = .R2wd, paragraph = TRUE)
```
wdGet

Arguments

eqtext  a text string containing the formula
bookmark  a text string containing the bookmark to use (otherwise default)
isknow  acknowledgement of danger using sendkeys
waitsec  time R waits while sendkey sends the equation
wdapp  the handle to the Word application
paragraph  whether to insert a paragraph after the equation

Details

This function uses two tools to write a formula to Word. At first it uses the COM client to write a
text into the range property of the selection and converts it to a formula. Then it uses the sendkeys
method of the wsshell to add a carriage return. This causes Word to parse the formula and to
substitute LaTeX like tokens such as \alpha by their corresponding formula equivalents (the greek
letter alpha). In principle, it should be possible to achieve the parsing of the formula by invoking
the BuildUp method of the omaths(1) object, but this does not work in the current versions of Word.
The route via Sendkeys is fragile, so handle with caution.

Value

none

Examples

## Not run:
wdGet()
wdTitle("Try a formula")
## note that double \ has to be used instead of \
## note that the , the following lines
## must be pasted into the console and run there.
## running them from an Emacs/ESS buffer won't work
wdEqn("\alpha \over ( \beta +3 )")
wdEqn("\alpha \over (( \beta +3 ))")
wdEqn("\matrix(\alpha &0&0&1 )")
## note: the handling of parentheses "(" is a bit peculiar in Word
## also experiment with the handling of empty spaces

## End(Not run)

---

wdGet  

*Gets a handle to Word (using RDCOMClient).*

Description

wdGet checks first if Word is already running. If this is not the case, it will start Word, add a
document and then return the handle to the Word application. If Word is already running and has
an active document, it will just return the handle.
wdGetProperty

Usage
wdGet(filename = NULL, path = "", method = "rcom", visible = TRUE)

Arguments
filename the filename of an existing word document (if null, make a new one)
path the path to an existing word document
method the COM client. Either "rcom" (default) or "RDCOMClient"
visible whether the Word application should be visible.

Value
a handle to the COM object referring to the Word application.

Author(s)
Christian Ritter

Examples
## Not run:
wdGet()
wdTitle("R2wd, a package to write Word documents from R")

## End(Not run)

wdGetProperty Get a property from Word.

Description
This function descends down a property tree given in the parameter vector property and returns
the lowest level.

Usage
wdGetProperty(property, object = wdapp["Selection"], wdapp = .R2wd)

Arguments
property The name of a property or a vector indicating a path in a property tree.
object The object on which the property tree should start.
wdapp The handle to Word.
Details
This function descends down the indicated property tree and returns the lowest level. It can also work with calls to the 'Items' method. In this case, the corresponding item numbers should be inserted in the property vector.

Value
The value of the lowest level of the property tree.

Author(s)
Christian Ritter

See Also
wdSetProperty

Examples
## Not run:
wdGet()
wdTitle("Title",paragraph=FALSE)
wdGetProperty(c("Font","Name"))
wdGetProperty(c("Font","Size"))
wdParagraph()
wdWrite("This is in normal font")
wdGetProperty(c("Font","Name"))
wdGetProperty(c("Font","Size"))

## End(Not run)

---

wdGoToBookmark Navigate to a bookmark in the active Word document.

Description
Looks up the bookmark in the bookmarks collection and goes to it.

Usage
wdGoToBookmark(bookmark, wdapp = .R2wd)

Arguments
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bookmark</td>
<td>the name of the bookmark.</td>
</tr>
<tr>
<td>wdapp</td>
<td>the handle to the Word application (usually not needed).</td>
</tr>
</tbody>
</table>
Examples

```r
## Not run:
wdGet()
## the following command starts a new section and adds a bookmark.
wdSection("This a new section", label="sec1")
## now we add some text
wdBody("R2wd is a package to write Word documents from R")
## we can now go back to the section header
wdGoToBookmark("sec1")

## End(Not run)
```

---

**wdHeading**

*Add text in Heading style.*

Description

Add text in Heading style.

Usage

```r
wdHeading(level = 1, text = "", paragraph = TRUE, wdapp = .Rwd)
```

Arguments

- **level**
  The heading level.
- **text**
  The text to write as a heading.
- **paragraph**
  Whether a paragraph should start after the heading.
- **wdapp**
  The handle to the Word Application (usually not needed).

Author(s)

Christian Ritter

Examples

```r
## Not run:
wdGet()
wdHeading(1,"R2wd")
wdHeading(2,"General ",paragraph=FALSE)
wdHeading(2,"Principles")

## End(Not run)
```
**wdInsertBookmark**  
*Insert a Bookmark.*

**Description**

Inserts a Bookmark at the current selection.

**Usage**

```r
wdInsertBookmark(text, wdapp = .R2wd)
```

**Arguments**

- `text`: the bookmark text.
- `wdapp`: the handle to the Word Application (usually not needed).

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run:
wdGet()
wdTitle("R2wd a package for writing Word documents from R")
wdInsertBookmark("here")
wdBody("This package allows ...")
wdSection("General")
wdGoToBookmark("here")
## End(Not run)
```

---

**wdInsertFootnote**  
*Insert a Footnote*

**Description**

Inserts a footnote at the current selection.

**Usage**

```r
wdInsertFootnote(text=", reference=", wdapp=.R2wd)
```

**Arguments**

- `text`: the footnote text.
- `reference`: the footnote reference. By default, this is generated automatically by Word.
- `wdapp`: the handle to the Word Application (usually not needed).
**wdItemize**

**Value**

(invisible): a handle to the footnote object. Can be used to manipulate (usually not used).

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run:
wdGet()
wsetTitle("R2wd a package for writing Word documents from R")
winsertFootnote("see also 5WORD (rcom.unive.ac.at) for integrating R calculations in word documents.")
wdBody("This package allows ...")

## End(Not run)
```

---

### `wdItemize`

*Start a bullet list*

**Description**

This corresponds to pushing the [123] button in Word.

**Usage**

```r
wdItemize(Gallery=1, Template=1, wdapp = .Rwd)
```

**Arguments**

- **Gallery**
  - the (numbered) item in the list gallery. The pre-set Gallery=1 implies that bullets are used.
- **Template**
  - the (numbered) item in the list template. The pre-set Template=1 implies that the first type is used.
- **wdapp**
  - the handle to word (can be ignored).

**Details**

Starts a bullet list like pushing on the [123] button in Word.

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run: wdItemize("R2wd is a package to write MS-Word files from R")
```
**wdNewDoc**

*Start a new document (when Word is running).*

**Description**

This adds a new document to a running Word Application. If a name is given, the new document is saved under the given name.

**Usage**

```c
wdNewDoc(name = NULL, wdapp = .R2wd)
```

**Arguments**

- **name**: the optional name of the file.
- **wdapp**: the handle to word (usually not needed)

**Author(s)**

Christian Ritter

**See Also**

- `wdGet`

**Examples**

```c
## Not run:
wdGet()
wdNewDoc("This.doc")
wdQuit()

## End(Not run)
```

**wdNormal**

*Type text in Normal style.*

**Description**

Type text in `Normal` style and end with a paragraph break if requested.

**Usage**

```c
wdNormal(text = "", paragraph = TRUE, wdapp = .R2wd)
```
**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>text</td>
<td>the text to write.</td>
</tr>
<tr>
<td>paragraph</td>
<td>whether a paragraph break should be added at the end.</td>
</tr>
<tr>
<td>wdapp</td>
<td>the handle to the Word Application (usually not needed).</td>
</tr>
</tbody>
</table>

**Note**

This is almost the same as wdBody.

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run:
wdGet()
wdTitle("R2wd: A package to write ...")
wdNormal("R2wd is a package for ...")

## End(Not run)
```

---

**wdPageBreak**

*Insert a page break.*

**Description**

Insert a page break.

**Usage**

```r
wdPageBreak(wdapp = .R2wd)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wdapp</td>
<td>the handle to Word Application (usually not needed).</td>
</tr>
</tbody>
</table>

**Examples**

```r
## Not run:
wdGet()
wdTitle("R2wd: Writing Word Documents from R")
wdBody("R2wd permits writing ...")
wdPageBreak(continuous=FALSE)
wdBody("It goes on on the next page")

## End(Not run)
```
**wdPageSetup**

Modify page setup in Word.

**Description**
Modify page setup in Word.

**Usage**

```r
wdPageSetup(orientation = "portrait", margins = rep(1, 4), scope = "section", wdapp = .R2wd)
```

**Arguments**

- `orientation`: page orientation, either "portrait" or "landscape"
- `margins`: a vector of page margins in inches (a unit variable will be supplied later)
- `scope`: whether the change applies to the entire document ("all") or only to the present section ("section").
- `wdapp`: handle to the Word Application (usually not needed).

**Details**
this invokes the required methods in word to set page orientation and margins

**Examples**

```r
## Not run:
wdGet()
wdPageSetup(orientation="landscape",margins=c(2,2,2),scope="all")

## End(Not run)
```

**wdParagraph**

Insert a paragraph break.

**Description**
Insert a paragraph break.

**Usage**

```r
wdParagraph(wdapp = .R2wd)
```

**Arguments**

- `wdapp`: the handle to Word Application (usually not needed).
**wdPlot**  
*create an R plot and paste it into word.*

**Description**

By default, this uses the `plot` function to create the plot according to the arguments given as `. . .`. The `plot` function can be replaced by another function which creates a graph.

**Usage**

```r
wdPlot(...) plotfun = plot, caption="", method="metafile", height = 5, width = 5, pointsize = 10, bookmark = NULL, wdapp = .R2wd, paragraph = TRUE)
```

**Arguments**

- `...` the arguments to the plot function
- `plotfun` the plot function (by default `plot`)
- `caption` figure caption
- `method` the graphics device type (`metafile` or `bitmap`)
- `height` the height of the plot in R units (commonly inches)
- `width` the width of the plot in R units (commonly inches)
- `pointsize` the pointsize of the plot
- `bookmark` the bookmark text (if missing, a default will be created)
- `wdapp` the handle to the Word Application (usually not needed)
- `paragraph` whether a paragraph is given after inserting the plot.

**Details**

The plot is transferred as a windows metafile.

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run:
wGet()
## a basic call using the default metafile device
wTitle("R2wd: plotting")
wPlot(1:100, sin(1:100), type="l", bty="l")
## a call using a ggplot function which uses semi-transparency
## this requires the bitmap device (the metafile device is nicer but doesn't render semi-transparency)
require(ggplot2)
funny<-function(){
c <- ggplot(mtcars, aes(qsec, wt))
```
wdQuit

print(c + stat_smooth())
}
wdPlot(plotfun=funny,method="bitmap")

## End(Not run)

---

**wdQuit**

*Close Word and remove the handle.*

**Description**

Close Word and remove the handle.

**Usage**

```
wdQuit(wdapp = .Rwd)
```

**Arguments**

- **wdapp**
  
  the handle to the Word Application.

**Details**

This closes Word. Word may ask back whether you wish to save the graph.

**Examples**

```
## Not run:
wdGet()
wdQuit()

## End(Not run)
```

---

**wdSave**

*Save the active document.*

**Description**

Save the active document. If a name is given, it is used for the new document, otherwise Word will ask.

**Usage**

```
wdSave(Name = NULL, wdapp = .Rwd)
```
Arguments

Name
File name (if missing, Word will ask).

wdapp
The handle to the Word Application (usually not needed).

Details

Saves the active document to the name if given or Word will ask for a name.

Author(s)

Christian Ritter

Examples

## Not run:
wdGet()
wdSave("This File.doc")
wdQuit()

## End(Not run)

---

`wdSection`  
*Start a new section of the document.*

Description

Start a section by: Adding a section break, switching to Heading1 style, adding the section title, adding a bookmark, adding a paragraph break, switching to Normal style.

Usage

```r
wdSection(title, label = gsub("\[.-:?!@#*\]", ",", paste("sec", title, sep = ",")), newpage = FALSE, wdapp = .R2wd)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>section title</td>
</tr>
<tr>
<td>label</td>
<td>bookmark text (will be generated automatically if missing)</td>
</tr>
<tr>
<td>newpage</td>
<td>whether the section should start on a new page (doesn’t work)</td>
</tr>
<tr>
<td>wdapp</td>
<td>handle to Word Application (usually not needed).</td>
</tr>
</tbody>
</table>

Author(s)

Christian Ritter
**Examples**

```r
## Not run:
wdGet()
wdTitle("R2wd a package to ... ")
wdSection("Introduction", newpage=TRUE)
```

## End(Not run)

---

**wdSectionBreak**

*Insert a section break.*

**Description**

Insert a section break. Depending on the setting of continuous start a new page for this.

**Usage**

```
wdSectionBreak( continuous = TRUE, bookmark = NULL, wdapp = .R2wd)
```

**Arguments**

- `continuous`: if FALSE, the Word section will start on a new page
- `bookmark`: bookmark text
- `wdapp`: handle to Word application (usually not needed)

**Author(s)**

Christian Ritter

---

**wdSelection**

*Get handle to current Selection in Word Application.*

**Description**

Get handle to current Selection in Word Application.

**Usage**

```
wdSelection(wdapp = .R2wd)
```

**Arguments**

- `wdapp`: handle to Word Application (usually not needed)

**Author(s)**

Christian Ritter
**wdSetFont**  
*Set font in Word.*

**Description**

This function allows to set the font type and the font size of the active Word document.

**Usage**

```
wdSetFont(fontname = NULL, fontsize = NULL, bold=NULL, italic=NULL, wdapp = .R2wd)
```

**Arguments**

- **fontname**: A font name Word knows about, such as Arial, Times New Roman, etc.
- **fontsize**: The point size of the font.
- **bold**: The point size of the font.
- **italic**: The point size of the font.
- **wdapp**: The handle to Word.

**Details**

This function uses the handle .R2wd to talk to Word and to change font name and size at the current position of the cursor in the active document.

**Author(s)**

Christian Ritter

**Examples**

```
## Not run:
wGet()
wTitle("Title")
wWrite("This is in normal font")
wSetFont(fontname="Garamond",fontsize=14,bold=TRUE,italic=TRUE)
wWrite("This is in 14pt Garamond")
wNormal("This switches back to Normal")

## End(Not run)
```


**wdSetProperty**

Set a property in Word.

**Description**

This function descends down a property tree given in the parameter vector `property` and sets the lowest level to `value`.

**Usage**

```
wdSetProperty(property, value, object = wdapp["Selection"], wdapp = .R2wd)
```

**Arguments**

- `property` The name of a property or a vector indicating a path in a property tree.
- `value` The value, the lowest level in the property tree should be set to.
- `object` The object on which the property tree should start.
- `wdapp` The handle to Word.

**Details**

This function descends down the indicated property tree. It can also work with calls to the 'Items' method. In this case, the corresponding item numbers should be inserted in the `property` vector.

**Author(s)**

Christian Ritter

**See Also**

- `wdGetProperty`

**Examples**

```r
## Not run:
wdGet()
wdTitle("Title")
wdWrite("This is in normal font")
wdSetProperty(c("Font","Name"),"Garamond")
wdSetProperty(c("Font","Size"),20)
wdWrite("This is in 14pt Garamond")
wdNormal("This switches back to Normal")

## End(Not run)
```
wdStylesetup  

Modify document style

Description
Modify document style

Usage
wdStylesetup(style = "Normal", fontsize = 11, align=3, wdapp = .R2wd)

Arguments
- style: default: "Normal"
- fontsize: default 11pt
- align: type of alignment: 3 corresponds to "justified"
- wdapp: handle to the Word Application (usually not needed).

Details
this invokes the required methods in word to set the document style

Author(s)
Christian Ritter

Examples
```r
## Not run:
wdGet()
wdStylesetup()

## End(Not run)
```

wdSubsection  

Start a new Word section, add a heading and a bookmark.

Description
Start a subsection (see wdSection for details.

Usage
wdSubsection(title, label = gsub("[.-:?!@#++ ]", ",", paste("subsec", title, sep = " "),
              newpage = FALSE, wdapp = .R2wd)
wdSubsubsection

Arguments

- **title**  the sub section title.
- **label**  the bookmark.
- **newpage**  whether the section should start on a new page.
- **wdapp**  the handle to the Word Application (usually not needed).

Author(s)

Christian Ritter

Examples

```r
## Not run:
wdGet()
wdTitle("Title")
wdSection("Section 1", newpage=TRUE)
wdSubsection("Sub-section 1.1")
wdSubsubsection("Sub-sub-section 1.1.a")

## End(Not run)
```

wdSubsubsection  

Start a new Word section, add a heading and a bookmark.

Description

Start a subsection (see [wdSection](#) for details.

Usage

```r
wdSubsubsection(title, label = gsub("[.,-:?!@#* ]", ",",
paste("subsec", title, sep = ",")),
newpage = FALSE, wdapp = .R2wd)
```

Arguments

- **title**  the sub-sub-section title.
- **label**  the bookmark.
- **newpage**  whether the section should start on a new page.
- **wdapp**  the handle to the Word Application (usually not needed).

Author(s)

Christian Ritter
Examples

```r
## Not run:
wdGet()
wdTitle("Title")
wdSection("Section 1",newpage=TRUE)
wdSubsection("Sub-section 1.1")
wdSubsubsection("Sub-sub-section 1.1.a")
```

wdTable

Write a dataframe or an array as a Word table.

Description

Make a Word table to the adequate dimensions, fill it, format it, and add a bookmark (and a caption - doesn’t work yet).

Usage

```r
wdTable(data, caption = "", caption.pos="below",bookmark = NULL, pointsize = 9, padding = 5, autoformat = 0, rowNames = TRUE, align = c("l",rep("r",ncol(data))))
```

Arguments

- `data`: a data frame or an array
- `caption`: the caption to use in Word
- `caption.pos`: the caption position "below" or "above"
- `bookmark`: the bookmark to use in Word
- `pointsize`: the pointsize in Word
- `padding`: how much cell-padding (in points)
- `autoformat`: which of the Word autoformats to use (try 1, 2, 3)
- `rowNames`: whether row names should be printed (default=TRUE)
- `align`: alignment instruction. Default: `c("l",rep("r",ncol(data)))`: align first column left and the others right. "c" stands for centering. The character "l" can be inserted where the user wishes a vertical line to be drawn.
- `hlines`: a character vector of length nrow(data)+1 containing "b","t","bt",or "n" to indicate whether a horizontal line should be drawn below, on top, or both. "n" means none.
- `wdapp`: the handle to the Word Application (usually not needed).

Details

Creates a Word table to the adequate dimensions and inserts it between two paragraph breaks. Fills it with dataframe or array (using the clipboard). Creates a bookmark and adds a caption. Uses word autoformatting and aligns columns according to user input.
**wdTitle**

Add text in title style.

**Description**

Add text in title style and add a paragraph break if needed.

**Usage**

```
wdTitle(title, label = substring(gsub("[.]", ",", paste("text", title, sep = ",")), 1, 16), paragraph = TRUE, wdapp = .R2wd)
```

**Arguments**

- **title**: the text to format in title style
- **label**: the bookmark text (if given)
- **paragraph**: whether a paragraph break should be added at the end.
- **wdapp**: the handle to the Word Application (usually not needed).

**Author(s)**

Christian Ritter

**Examples**

```
## Not run:
wdGet()
wdTitle("Title")
wdSection("Section 1", newpage=TRUE)
wdSubsection("Sub-section 1.1")
wdSubsubsection("Sub-sub-section 1.1.a")

## End(Not run)
```
wdType

**Description**

Inserts text at the current cursor point in Word (by default without messing with styles).

**Usage**

\[\text{wdType(text} = \text{"", italic=}\text{FALSE, alignment=}\text{"nothing", paragraph=}\text{TRUE, wdapp} = .R2wd)\]

**Arguments**

- **text** a test string to be written to Word.
- **italic** if TRUE then the text will be written in italics
- **alignment** choice of "nothing", "left", "center", "right".
- **paragraph** whether a paragraph (line return) should be added at the end.
- **wdapp** the handle to word (can be ignored).

**Details**

Switches to 'Type' style, adds the text, and then gives a carriage return (paragraph break) if requested.

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run: wdType("R2wd is a package to write MS-Word files from R")
```

wdUndo

**Description**

Save the active document. If a name is given, it is used for the new document, otherwise Word will ask.

**Usage**

\[\text{wdUndo(times} = \text{1, wdapp} = .R2wd)\]
**wdVerbatim**

**Arguments**

- **times**  
  the times argument of the Undo method of the word document
- **wdapp**  
  The handle to the Word Application (usually not needed).

**Details**

Saves the active document to the name if given or Word will ask for a name.

**Author(s)**

Christian Ritter

**Examples**

```r
## Not run:
wdGet()
wdTitle("By default, writing a title represents 5 calls to word: Style, type text, inserting a bookmark, typing a paragraph, and setting the style back to normal")
# we can undo it by invoking the Undo method on the document five times
wdUndo(5)
wdQuit()
```

```r
## End(Not run)
```

---

**Description**

write single spaced text in (by default) Courier font to word. This is useful to render R output 'as is'.

**Usage**

```r
wdVerbatim(text = "", paragraph = TRUE, fontsize = 9, fontname = "Courier New", wdapp = .R2wd)
```

**Arguments**

- **text**  
  the text to write
- **paragraph**  
  whether the chunk should be closed by a paragraph return.
- **fontsize**  
  size of the font
- **fontname**  
  name of the font
- **wdapp**  
  handle to the Word application (usually not used).```
Details

Saves the current style. Switches to Courier New, single spaced, 9 pt (by default) and renders the text. Switches back to the previous style.

Note

This is almost the similar to wdBody and wdWrite.

Author(s)

Christian Ritter

Examples

```r
## Not run:
wdGet()
tt <- capture.output(summary(lm(mpg~hp, data=mtcars)))
wdVerbatim(tt)

## End(Not run)
```

---

**wdWrite**

*Write text to word.*

### Description

Write text in current style and end with a paragraph break if requested.

### Usage

```r
wdWrite(text = "", paragraph = FALSE, wdapp = .R2wd)
```

### Arguments

- `text` the text to write.
- `paragraph` whether a paragraph break should be added at the end.
- `wdapp` the handle to the Word Application (usually not needed).

### Note

This is almost the same as wdBody.

### Author(s)

Christian Ritter
Examples

```r
## Not run:
wdGet()
wdTitle("R2wd: A package to write ...")
wdSetFont(fontname="Symbol")
wdWrite("R2wd is a package for ...")

## End(Not run)
```
Index

*Topic IO

wdApplyTemplate, 4
wdApplyTheme, 5
wdBody, 6
wdConvert, 7
wdEnumerate, 8
wdEqn, 8
wdGet, 9
wdGetProperty, 10
wdGoToBookmark, 11
wdHeading, 12
wdInsertBookmark, 13
wdInsertFootnote, 13
wdItemize, 14
wdNewDoc, 15
wdNormal, 15
wdPageBreak, 16
wdPageSetup, 17
wdParagraph, 17
wdPlot, 18
wdQuit, 19
wdSave, 19
wdSection, 20
wdSectionBreak, 21
wdSelection, 21
wdSetFont, 22
wdSetProperty, 23
wdStyleSetup, 24
wdSubsection, 24
wdSubsubsection, 25
wdTable, 26
wdTitle, 27
wdType, 28
wdUndo, 28
wdVerbatim, 29
wdWrite, 30

*Topic package

R2wd-package, 2

R2wd (R2wd-package), 2
R2wd-package, 2

wdApplyTemplate, 4
wdApplyTheme, 5
INDEX

wdBody, 6
wdConvert, 7
wdEnumerate, 8
wdEqn, 8
wdGet, 9, 15
wdGetProperty, 10, 23
wdGoToBookmark, 11
wdHeading, 12
wdInsertBookmark, 13
wdInsertFootnote, 13
wdItemize, 14
wdNewDoc, 15
wdNormal, 15
wdPageBreak, 16
wdPageSetup, 17
wdParagraph, 17
wdPlot, 18
wdQuit, 19
wdSave, 19
wdSection, 20, 24, 25
wdSectionBreak, 21
wdSelection, 21
wdSetFont, 22
wdSetProperty, \textit{22, 23}
wdStyleSetup, 24
wdSubsection, 24
wdSubsubsection, 25
wdTable, 26
wdTitle, 27
wdType, 28
wdUndo, 28
wdVerbatim, 29
wdWrite, 30