Package 'vprint'

September 25, 2026

Version 1.5

Date 2026-09-25

Title More Flexible Form of Boolean Verbose

Maintainer Barry Zeeberg <barryz2013@gmail.com>

Author Barry Zeeberg [aut, cre]

Depends R (>= 4.2.0)

Imports crayon, stringr

Description R functions are not supposed to print text without giving the user the option to turn the printing off or on using a Boolean 'verbose' in a construct like 'if(verbose) print(...)'. But this black/white approach is rather rigid, and an approach with shades of gray might be more appropriate in many circumstances. As of Version 1.4, also supports text and background colors, as well as text styles.

License GPL (>= 2)

Encoding UTF-8

VignetteBuilder knitr

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Repository CRAN

Date/Publication 2026-09-25 21:40:02 UTC

Config/roxygen2/version 8.1.0

Contents

ncharz 2
promptz 2
splitNewline 3
vprint 3
vprint0 5

Index **7**

ncharz *ncharz*

Description

nchar() decremented by number of escape \

Usage

```
ncharz(txt)
```

Arguments

txt	character string

Examples

```
ncharz("abc\'")
```

promptz *promptz*

Description

implement readline(prompt = 'Enter xxx: ') with colors from crayon package

Usage

```
promptz(txt, style = "green$bgBlack")
```

Arguments

txt	character string containing the prompt
style	style parameter to crayon package, e.g. yellow$bgMagenta$bold

Details

does not use the conditional mechanism of vprint(,verbose . . .) because a prompt for user input is expected to be required, not optional

Value

returns the user response

Examples

```
n<-promptz("Enter n: ","green$bgBlack")
```

---

splitNewline *splitNewline*

Description

split character string at newline

Usage

```
splitNewline(text)
```

Arguments

text	character string

Value

returns a vector of character strings

Examples

```
splitNewline("Line One\nLine Two\nLine Three")
```

---

vprint *vprint*

Description

more flexible form of if(verbose) print(...)

Usage

```
vprint(
  class,
  verbose,
  txt,
  nNew = 0,
  style = "reset",
  bkFill = FALSE,
  trunc = FALSE,
  original = FALSE
)
```

Arguments

class	integer representing the verbose class
verbose	integer vector representing classes
txt	argument to print
nNew	integer number of extra newlines to precede (nNew<0) or follow (nNew>0) txt
style	style parameter to crayon package, e.g. yellow$bgMagenta$bold
bkFill	Boolean if TRUE extend background color to full width of page
trunc	Boolean if TRUE truncate txt to width of page
original	Boolean if TRUE use original vprint implementing print (for e.g. lists) rather than cat

Details

If you are the developer, and you have some debug statements rather than comment them out, you can turn them off and on likewise, a user can request more or less informational comments

Refer to the documentation of the crayon package for full list of style parameters, and how to concatenate them using '$'

crayon package works in RStudio Console but not in R Console as a workaround, you can open a e.g. UNIX terminal window, type 'R', and use this as an R Console that does support crayon

performs special processing for strings that contain newline and/or single quotes

suggested standardized class codes for vprint():

-1 = developer debugging only

0 = constitutively turned on

1 = help for new user

2 = follow progress of long computation

3 = primary results

4 = meta info (e.g. dims of a mat before/after trimming)

5 = warnings

6 = errors

Note that the class argument is hardwired into the function code by the developer. For example, for a debugging statement, '-1' is hardwired in. The choice of whether or not to display this message is subsequently governed by the user selecting which values to include in the 'verbose' vector parameter.

Value

returns no values but has side effect of printing some text

Examples

```
vprint(1,1:2,"try me",2)
vprint(0,1:2,"try me",1)
vprint(3,1:2,"try me")
vprint(1,1:2,"try me",0,style="yellow$bgMagenta$bold")
vprint(1,1:2,"try me",0,style="yellow$bgMagenta$bold",bkFill=TRUE)
txt<-sprintf("try me%s",strrep(" ", 50))
vprint(1,1:2,txt,0,style="yellow$bgMagenta$bold",bkFill=FALSE,trunc=FALSE)
vprint(1,1:2,"try\nme",0,style="yellow$bgMagenta$bold",bkFill=TRUE)
vprint(1,1:2,"try 'me'",0,style="yellow$bgMagenta$bold",bkFill=TRUE)
vprint(1,1:2,character(0),0,style="yellow$bgMagenta$bold",bkFill=TRUE)
l<-list()
l["a"]]<-"A"
l["b"]]<-"B"
vprint(1,1,l) # fails for list argument
vprint(1,1,l,original=TRUE) # works for list argument
```

---

vprint0 *vprint0*

Description

more flexible form of if(verbose) print(...)

Usage

```
vprint0(class, verbose = NULL, txt, nNew = 0)
```

Arguments

class	integer representing the verbose class
verbose	integer vector representing classes
txt	argument to print
nNew	integer number of extra newlines to precede (nNew<0) or follow (nNew>0) txt

Details

if you are the developer, and you have some debug statements rather than comment them out, you can turn them off and on likewise, a user can request more or less informational comments

suggested standardized class codes for vprint():

-1 = developer debugging only

0 = constitutively turned on

1 = help for new user

2 = follow progress of long computation

3 = primary results

4 = meta info (e.g. dims of a mat before/after trimming)

5 = warnings

6 = errors

Note that the class argument is hardwired into the function code by the developer. For example, for a debugging statement, '-1' is hardwired in. The choice of whether or not to display this message is subsequently governed by the user selecting which values to include in the 'verbose' vector parameter.

Value

returns no values but has side effect of printing some text

Examples

```
vprint0(1,1:2,"try me",2)
vprint0(0,1:2,"try me")
vprint0(3,1:2,"try me")
```

Index

ncharz, 2

promptz, 2

splitNewline, 3

vprint, 3
vprint0, 5