

Package: colorout (via r-universe)

August 9, 2026

Version 1.3-3

Date 2025-06-02

Title Colorize R Output on Terminal Emulators

Author Jakson Aquino and Dominique-Laurent Couturier

Maintainer Jakson Alves de Aquino <jalvesaq@gmail.com>

OS_type unix

Imports utils

Description Colorize R output when it is running on a terminal emulator. The functions of this package only work if R is compiled for Unix systems and it is running interactively in a terminal emulator. The terminal must support Select Graphic Rendition (SGR, also known as ANSI escape codes or sequences). The package contains a C library with a function that replaces the default Rstd_WriteConsoleEx which, when enabled, is responsible for printing messages in the Console when R is running in interactive mode. It works with RStudio, but there is no support for Graphical User Interfaces that does not support ANSI colors, such as Windows RGui.

License GPL (>= 2)

URL <https://github.com/jalvesaq/colorout>

Repository <https://r-multiverse-staging.r-universe.dev>

Date/Publication 2025-06-02 13:04:23 UTC

RemoteUrl <https://github.com/jalvesaq/colorout>

RemoteRef v1.3-3

RemoteSha 15ff0be4a3fcd8504c1d7ff1d05a64b32bab552

Contents

colorout-package	2
addPattern	5
ColorOut	6

deletePattern	6
isColorOut	7
listPatterns	7
noColorOut	8
setOutputColors	8
setOutputColors256	11
setZero	11
show256Colors	12
unsetZero	13
Index	14

colorout-package	<i>Colorize R output on terminal emulators</i>
------------------	--

Description

Colorize R output when it is running on a Unix terminal emulator. The library also works within RStudio, but there is no support for other Graphical User Interfaces, such as Windows RGui, RKnward, JGR, Rcmdr that cannot display ANSI colors. It also did not work with Emacs with ess. The functions of this package only work if R is compiled for Unix systems (e.g., Linux and OS X) and it is running interactively in a terminal emulator. The terminal emulator might have been called by a text editor, such as Vim, Gedit, Kate or Geany.

Homepage: <https://github.com/jalvesaq/colorout>

Screenshot:

```

R
setwidth 1.0-3 loaded
vimcom.plus 1.0-0 loaded
> cat("Different colors for normal text, \"strings\", dates (",
+   as.character(Sys.Date()), ")\n",
+   "numbers (12, -1.3), NULL, NA, NaN, Inf, TRUE and FALSE.\n", sep = "")
Different colors for normal text, "strings", dates (2014-05-03)
numbers (12, -1.3), NULL, NA, NaN, Inf, TRUE and FALSE.
> x <- data.frame(logic=c(T, T, F), factor=factor(c("abc","def", "ghi")),
+   string=c("ABC","DEF", "GHI"), real=c(1.23, -4.56, 7.89),
+   cien.not=c(1.234e-23, -4.56e+45, 7.89e78),
+   date=as.Date(c("2012-02-21","2013-02-12", "2014-03-04")),
+   stringsAsFactors = FALSE)
> rownames(x) <- 1:3
> x
  logic factor string  real  cien.not    date
1 TRUE   abc   ABC  1.23  1.234e-23 2012-02-21
2 TRUE   def   DEF -4.56 -4.560e+45 2013-02-12
3 FALSE  ghi   GHI  7.89  7.890e+78 2014-03-04
> summary(x[, c(1, 2, 4, 6)])
  logic factor string  real  cien.not    date
Mode :logical abc:1  Min.  :-4.560  Min.   :2012-02-21
FALSE:1  def:1  1st Qu.: -1.665  1st Qu.: 2012-08-17
TRUE :2   ghi:1  Median :  1.230  Median : 2013-02-12
NA's :0              Mean  :  1.520  Mean   : 2013-02-21
              3rd Qu.:  4.560  3rd Qu.: 2013-08-23
              Max.   :  7.890  Max.   : 2014-03-04
> # Warnings and errors are highlighted (even if not in English):
> warning("This is an example of warning.")
Mensagens de aviso perdidas:
This is an example of warning.
> example.of.error
Erro: objeto 'example.of.error' não encontrado
> # Messages sent to stderr are highlighted:
> library(KernSmooth)
KernSmooth 2.23 loaded
Copyright M. P. Wand 1997-2009
> # The colors are customizable:
> setOutputColors()
normal, number, negnum, date, string, const, stderr, warn, error.
> setOutputColors256(normal = 39, number = 51, negnum = 183, date = 43,
+   string = 79, const = 75, verbose = FALSE)
> x
  logic factor string  real  cien.not    date
1 TRUE   abc   ABC  1.23  1.234e-23 2012-02-21
2 TRUE   def   DEF -4.56 -4.560e+45 2013-02-12
3 FALSE  ghi   GHI  7.89  7.890e+78 2014-03-04
> setOutputColors256(202, 214, 209, 184, 172, 179, verbose = FALSE)
> x
  logic factor string  real  cien.not    date
1 TRUE   abc   ABC  1.23  1.234e-23 2012-02-21
2 TRUE   def   DEF -4.56 -4.560e+45 2013-02-12
3 FALSE  ghi   GHI  7.89  7.890e+78 2014-03-04
>

```

Details

The terminal must support Select Graphic Rendition (SGR, also known as ANSI escape codes or sequences), otherwise you may see garbage like this:

```

> rnorm(5)
[32m[ [33m1 [32m] [0m [32m [33m0.07574585 [32m [0m [32m
[33m0.88167822 [32m [0m [32m [33m0.60788656 [32m [0m [32m
[33m1.13590951 [32m [0m [32m [33m1.07758879 [32m [0m [32m [0m

```

The package contains a C library with a function that replaces the default `Rstd_WriteConsoleEx` which, when enabled, is responsible for printing messages in the Console when R is running in interactive mode. R should be built against `libreadline` since `libedit` may cause issues.

The function that enables the colorization of R output is [ColorOut](#), and it is called automatically when the package is loaded. However, it will do nothing if `Sys.getenv("TERM")` returns either "" or "dumb". The output will not be colorized also if the result of either `interactive()` or `isatty(stdout())` is FALSE. You can change this behavior by putting in your '~/.Rprofile' one or more of the following options:

```
options(colorout.verbose = 0)
options(colorout.anyterm = TRUE)
options(colorout.dumb = TRUE)
options(colorout.noninteractive = TRUE)
options(colorout.notatty = TRUE)
```

With `colorout.anyterm == TRUE`, the package will not check the 'TERM' environment variable, and it will also not test whether R is running interactively. If your terminal emulator is capable of displaying ANSI colors but is not being recognized you may put the following in your '~/.Rprofile':

```
if(interactive()){
  options(colorout.anyterm = TRUE)
  library(colorout)
}
```

And of course, you can use any arbitrary condition to decide whether to load colorout or not. For instance:

```
if(isatty(stdout())){
  options(colorout.anyterm = TRUE)
  library(colorout)
}
```

If `colorout.verbose > 0`, the package will display a warning if the output is not going to be colorized.

A 256 color scheme is used by default. If your terminal emulator only supports 8 colors, you will have to set your own color scheme. Please, see the example at [setOutputColors](#).

Author(s)

Jakson Alves de Aquino <jalvesaq@gmail.com> and Dominique-Laurent Couturier <d1c48@medsch1.cam.ac.uk>.

See Also

[ColorOut](#), [noColorOut](#), [setOutputColors](#), [show256Colors](#) and [setZero](#). The package `setwidth` updates the value of `options("width")` automatically when the terminal is resized.

addPattern	<i>Add custom pattern to be colored</i>
------------	---

Description

Enable coloring of custom patterns.

Usage

```
addPattern(pattern, color)
```

Arguments

pattern	String representing the pattern to be colored. The patterns are literal strings. The only two types of regular expression accepted are (1) a range of ASCII characters surrounded by square brackets and separated by a hyphen and (2) either an ASCII character or a range of ASCII characters followed by an asterisk (see section Examples below).
color	Formating and color of the pattern (see the Details section in setOutputColors).

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

Examples

```
# Slash separated date (YYYY-MM-DD):
addPattern("[1-2][0-9][0-9][0-9]/[0-1][0-9]/[0-3][0-9]", 179)

# Date in Year-quarter format:
addPattern("[1-2][0-9][0-9][0-9] Q[0-4]", 179)

# Date in "Month Year" format (Not perfect; it will wrongly colorize strings
# such as "Not 2019"):
addPattern("[A-S][a-u][b-z] [1-2][0-9][0-9][0-9]", 179)

# If all columns in a data.frame start with the letter 'q' followed by a
# number end, in some cases, by other letters, colorize them:
addPattern("q[0-9]*", 123)
addPattern("q[0-9]*[a-z]*", 123)
```

ColorOut*Colorize R output in terminal emulator*

Description

Colorize output of R running in a terminal emulator. The function is called automatically when the package is loaded.

Usage

```
ColorOut()
```

Details

The library works within RStudio, but there is no support for Graphical User Interfaces, that does not support ANSI colors. If running in a terminal emulator, it must support ANSI escape codes. Please, read [colorout-package](#) for more details.

Emacs/ESS users should read [colorout-package](#) to know how to enable and use the package.

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

deletePattern*Delete custom pattern that was being colorized*

Description

Do not colorize previously defined string pattern.

Usage

```
deletePattern(pattern)
```

Arguments

pattern String representing the pattern to be colorized. The patterns are literal strings.

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

See Also

[addPattern.](#)

Examples

```
# Slash separated date (YYYY-MM-DD):
deletePattern("[1-2][0-9][0-9][0-9]/[0-1][0-9]/[0-3][0-9]")

# Delete all user defined patterns:
deletePattern(listPatterns())
```

isColorOut

Check if colorizing of R output is enabled

Description

Function to check if colorizing of R output is enabled.

Usage

```
isColorOut()
```

Value

TRUE if the colorizing of R output is enabled and FALSE otherwise.

Author(s)

Neal Fultz and Jakson Aquino <jalvesaq@gmail.com>

listPatterns

List custom defined patterns that are colorized

Description

List patterns defined by users in their colors.

Usage

```
listPatterns()
```

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

noColorOut

Stop colorizing R output

Description

Stop colorizing R output.

Usage

```
noColorOut()
```

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

setOutputColors

Set the Colors to Be Used on R Output

Description

Set the output colors to be used if R is running in a terminal emulator. The terminal must be capable of displaying 256 colors. See a screenshot at [colorout-package](#).

Usage

```
setOutputColors(normal = 40, negnum = 209, zero = 226,
               number = 214, date = 179, string = 85,
               const = 35, false = 203, true = 78,
               infinite = 39, index = 30, stderr = 213,
               warn = c(1, 16, 196), error = c(160, 231),
               verbose = TRUE, zero.limit = NA)
```

Arguments

normal	Formating and color of normal text.
negnum	Formating and color of negative numbers.
zero	Formating and color of values being close to 0.
number	Formating and color of positive numbers.
date	Formating and color of dates and hours. For dates, the output format should be 'YYYYxMMxDD' or 'DDxMMxYYYY', where the separator x may equal '-' or '/'. For hours, the output format should be 'HH:MM:SS'.
string	Formating and color of quoted text.
const	Formating and color of NULL, NA and NaN.
false	Formating and color of FALSE.
true	Formating and color of TRUE.
infinite	Formating and color of Inf and -Inf.
index	Formating and color of vector indexes.
stderr	Formating and color of text sent to stderr.
warn	Formating and color of warnings.
error	Formating and color of errors.
verbose	Logical value indicating whether to print colored words showing the result of the setup.
zero.limit	The limit below which a value is considered close to 0. If set, the colorizing of zero and values near zero is enabled. You can also use the function setZero to set this value at any time and unsetZero to disable the colorizing of values near zero.

Details

The function takes numeric vectors of at most three elements each. The three numbers indicate, respectively, formating, background color and foreground color. If a vector has only one element, the number will be used to set the foreground color; if it has two elements, they will be used to set the background and foreground colors.

If you pass a string as a color, it will be used as is. Thus, you should pass valid strings (please, see the true color example below). The string must have at most 63 characters.

The table below shows valid formating values (some formating codes do not work properly in some terminal emulators):

Value	Formating
0	No formating
1	Bold or bright
2	Faint
3	Italic or inverse
4	Underline
5	Blink slowly

6	Blink quickly
7	Invert

The values of ANSI escape codes for 256 colors are different than the ones used in the function `setOutputColors`. Run `show256Colors` to get a table of the 256 colors and their codes.

Messages are colored as errors and warnings if they start with "Error" or "Warning" (or their translations, if not running in an English locale).

If the 'TERM' environment variable is "fbterm", fbterm escape codes will be used instead of the ANSI ones and the format values will be ignored.

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

Examples

```
# Color scheme for terminal emulator with only 8 colors:
setOutputColors(normal = 2, negnum = 3, zero = 3, number = 3,
               date = 3, string = 6, const = 5, false = 5,
               true = 2, infinite = 5, index = 2, stderr = 4,
               warn = c(1, 0, 1), error = c(1, 7),
               verbose = TRUE, zero.limit = NA)

# Color scheme for white background:
setOutputColors(normal = c(0, 0, 18), number = c(0, 0, 90),
               negnum = c(0, 0, 88), date = c(0, 0, 53),
               string = c(0, 0, 22), const = c(0, 0, 94), stderr = 52)

# Color schemes for black background:
setOutputColors(normal = 39, negnum = 183, zero = 114, number = 51,
               date = 43, string = 79, const = 75, zero.limit = 0.1)
setOutputColors(normal = 202, number = 214, zero = 220, negnum = 209,
               date = 184, string = 172, const = 179, zero.limit = 0.01)
setOutputColors(normal = 202, number = 214, zero = NA, negnum = 209,
               date = 184, string = 172, const = 179, zero.limit = 0.01)

# True color example:
# The color code starts with "\x1b[" and ends with "m"
# 38;2 means "foreground"
# 48;2 means "background"
# valid values for red, green, and blue are integers between 0 and 255
setOutputColors(normal = "\x1b[38;2;0;200;0m",      # red = 0; green = 200; blue = 0
               negnum = "\x1b[38;2;255;200;0m",
               zero = "\x1b[38;2;255;255;0m",
               number = "\x1b[38;2;200;255;75m",
               date = "\x1b[38;2;155;155;255m",
```

```

string  = "\x1b[38;2;0;255;175m",
const   = "\x1b[38;2;0;255;255m",
false   = "\x1b[38;2;255;125;125m",
true    = "\x1b[38;2;125;255;125m",
infinite = "\x1b[38;2;75;75;255m",
index   = "\x1b[38;2;00;150;80m",
stderr  = "\x1b[38;2;255;0;255m",
warn    = "\x1b[38;2;255;0;0m",
error   = "\x1b[38;2;255;255;255;48;2;255;0;0m",
zero.limit = 0.01)

```

setOutputColors256	<i>Deprecated Function</i>
--------------------	----------------------------

Description

This function is deprecated. Use [setOutputColors](#) instead.

Usage

```
setOutputColors256(...)
```

Arguments

... Arguments to be passed to [setOutputColors](#).

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

setZero	<i>Enable colorizing of numbers near zero</i>
---------	---

Description

Enable colorizing of numbers near zero

Usage

```
setZero(z = 1e-12)
```

Arguments

z The limit below which a value is considered close to 0.

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

See Also

[unsetZero](#)

show256Colors

Create and show a table with 256 colors

Description

Create and show a table with 256 colors and their ANSI escape codes. The table should be displayed in your browser. You have to hover the mouse over the colors to see their codes.

Usage

```
show256Colors(outfile = "/tmp/table256.html")
```

Arguments

outfile String. Path to file where the table will be saved.

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>, based on code from Todd Larason ('256colors.pl') and Ben Fritz ('2html.vim').

`unsetZero`*Stop colorizing numbers near zero*

Description

Stop colorizing numbers near zero

Usage

```
unsetZero()
```

Value

NULL.

Author(s)

Jakson A. Aquino <jalvesaq@gmail.com>

See Also

[setZero](#)

Index

- * **package**
 - colorout-package, [2](#)
- addPattern, [5](#), [7](#)
- ColorOut, [4](#), [6](#)
- colorout (colorout-package), [2](#)
- colorout-package, [2](#), [6](#), [8](#)
- deletePattern, [6](#)
- isColorOut, [7](#)
- listPatterns, [7](#)
- noColorOut, [4](#), [8](#)
- setOutputColors, [4](#), [5](#), [8](#), [10](#), [11](#)
- setOutputColors256, [11](#)
- setZero, [4](#), [9](#), [11](#), [13](#)
- show256Colors, [4](#), [10](#), [12](#)
- unsetZero, [9](#), [12](#), [13](#)