

# Package ‘multicolor’

March 18, 2019

**Type** Package

**Title** Add Multiple Colors to your Console Output

**Version** 0.1.2

**Description** Add multiple colors to text that is printed to the console.

**Depends** R (>= 2.10)

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** TRUE

**URL** <http://github.com/aedobbyn/multicolor/>

**BugReports** <https://github.com/aedobbyn/multicolor/issues/>

**Imports** cowsay, crayon, dplyr, glue, magrittr, purrr, stringi,  
stringr, tibble (>= 1.2), tidyr

**Suggests** covr, fansi, knitr, rmarkdown, testthat, viridisLite,  
wesanderson

**RoxygenNote** 6.1.1

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Amanda Dobbyn [aut, cre],  
Hernando Cortina [aut] (<<https://orcid.org/0000-0001-6790-4870>>)

**Maintainer** Amanda Dobbyn <[amanda.e.dobbyn@gmail.com](mailto:amanda.e.dobbyn@gmail.com)>

**Repository** CRAN

**Date/Publication** 2019-03-18 16:50:33 UTC

## R topics documented:

center_string	2
crawl	3
insert_rainbow	4
multicolor_logo	4

multi_color . . . . .	5
multi_colour . . . . .	6
nix_first_newline . . . . .	8
starwars_intro . . . . .	8
things . . . . .	9
triangle_string . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

center_string	<i>Center all lines of a string relative to console width.</i>
---------------	--

---

### Description

Center all lines of a string relative to console width.

### Usage

```
center_string(string, remove_last_break = TRUE, display = FALSE)
```

### Arguments

string	(character) Some text to center within console.
remove_last_break	(logical) Set to TRUE to remove last line break. Defaults to FALSE.
display	(logical) Returns string invisibly if FALSE (default), set to TRUE to display returned string

### Details

To removes last line break set `removelastbreak` to TRUE.

### Value

A string

### Examples

```
triangle_string(starwars_intro, display = TRUE) %>%
  center_string() %>%
  multi_color(direction = "horizontal", recycle_chars = TRUE)
```

---

crawl	<i>Multi-color crawling text</i>
-------	----------------------------------

---

### Description

This function crawls over `txt` producing an animated gif-like representation of the text unfolding from left to right or top to bottom, depending on `direction`, colored according to `colors`.

### Usage

```
crawl(txt = "hello world!", colors = NULL, recycle_chars = FALSE,  
      direction = "vertical", pause = 0.05, ...)
```

### Arguments

<code>txt</code>	(character) Some text to color, stripped of line breaks
<code>colors</code>	(character) A vector of colors to color each individual character, if <code>recycle_chars</code> is TRUE, or the whole string if FALSE, defaulting to the Viridis Plasma palette. Must all be <code>crayon</code> -supported colors. Any colors in <code>colors()</code> or hex values (see <code>?rgb</code> ) are fair game.
<code>recycle_chars</code>	(logical) Should the vector of colors supplied apply to the entire string or should it apply to each individual character (if <code>direction</code> is vertical) or line (if <code>direction</code> is horizontal), and be recycled?
<code>direction</code>	(character) How should the colors be spread? One of "horizontal" or "vertical".
<code>pause</code>	(numeric) Seconds to pause between characters in seconds.
<code>...</code>	Further args passed to <code>multi_color</code> .

### Details

This function requires as many colors as there are characters in your string and prints them one at a time. `colors` will be recycled in single-color equal-sized chunks if `recycle_char` is FALSE and character-by-character if `recycle_char` is TRUE.

Colors cannot be applied in RGUI (R.app on some systems) or other environments that do not support colored text. In these cases, the `txt` will simply be crawled over without applying colors.

### Value

A string, printed in colors with pause seconds between printing each character.

### Examples

```
## Not run:  
crawl()  
  
crawl("It was a dark and stormy night")
```

```

crawl("Taste the rainbow", colors = "rainbow")

crawl(things[["hypnotoad"]], colors = c("purple", "blue", "cyan"),
      direction = "horizontal", recycle_chars = TRUE, pause = 0.01)

options("keep.source = FALSE")
crawl('\014A long time ago in a galaxy far, far away...
It is a period of civil war. Rebel spaceships, striking from a hidden base,
have won their first victory against the evil Galactic Empire.')

## End(Not run)

```

---

insert_rainbow	<i>Insert Rainbow</i>
----------------	-----------------------

---

### Description

Take the string "rainbow" and replace it with c("red", "orange", "yellow", "green", "blue", "purple")

### Usage

```
insert_rainbow(clr)
```

### Arguments

clr (character) A vector of one or more colors.

### Value

A character vector of color names.

### Examples

```

insert_rainbow("rainbow")
insert_rainbow(c("lightsteelblue", "rainbow", "lightsalmon"))

```

---

multicolor_logo	<i>The multicolor package logo</i>
-----------------	------------------------------------

---

### Description

The multicolor package logo

### Usage

```
multicolor_logo(...)
```

**Arguments**

... Arguments passed to multi\_color.

**Examples**

```
multicolor_logo()
```

---

multi_color	<i>Multi-color text</i>
-------------	-------------------------

---

**Description**

Multi-color text

**Usage**

```
multi_color(txt = "hello world!", colors = "rainbow",
  type = "message", direction = "vertical", recycle_chars = FALSE,
  ...)
```

**Arguments**

txt	(character) Some text to color.
colors	(character) A vector of colors, defaulting to "rainbow", i.e. c("red", "orange", "yellow", "green", "blue", "purple"). Must all be <b>crayon</b> -supported colors. Any colors in colors() or hex values (see ?rgb) are fair game.
type	(character) "message" (the default), "warning", "string", or "rmd". If "rmd" is used, the type of the RMarkdown document should be html_document the chunk option results = "asis" should be used.
direction	(character) How should the colors be spread? One of "horizontal" or "vertical".
recycle_chars	(logical) Should the vector of colors supplied apply to the entire string or should it apply to each individual character (if direction is vertical) or line (if direction is horizontal), and be recycled?
...	Further args.

**Details**

This function evenly (ish) divides up your string into these colors in the order they appear in colors.

It cannot be used with RGUI (R.app on some systems).

**Value**

A string if type is "string", or colored text if type is "message" or "warning"

**Examples**

```
## Not run:
multi_colour()

multi_colour("ahoy")

multi_colour("taste the rainbow",
             c("mediumpurple",
               "rainbow",
               "cyan3"))

multi_colour(colors = c(rgb(0.1, 0.2, 0.5),
                         "yellow",
                         rgb(0.2, 0.9, 0.1)))

multi_colour(
  cowsay::animals[["buffalo"]],
  c("mediumorchid4", "dodgerblue1", "lemonchiffon1"))

multi_colour(cowsay::rms, sample(colors(), 10))

# Mystery Bulgarian animal
multi_colour(things[[sample(length(things), 1)]],
             c("white", "darkgreen", "darkred"),
             direction = "horizontal")

# Mystery Italian animal
multi_colour(things[[sample(length(things), 1)]],
             c("darkgreen", "white", "darkred"),
             direction = "vertical")

## End(Not run)
```

---

multi\_colour

*Multi-colour text*


---

**Description**

Multi-colour text

**Usage**

```
multi_colour(txt = "hello world!", colors = "rainbow",
             type = "message", direction = "vertical", recycle_chars = FALSE,
             ...)
```

**Arguments**

txt	(character) Some text to colour.
colors	(character) A vector of colours, defaulting to "rainbow", i.e. c("red", "orange", "yellow", "green", "blue", "purple"). Must all be <b>crayon</b> -supported colours. Any colours in colors() or hex values (see ?rgb) are fair game.
type	(character) Message (default), warning, or string.
direction	(character) How should the colours be spread? One of "horizontal" or "vertical".
recycle_chars	(logical) Should the vector of colours supplied apply to the entire string or should it apply to each individual character (if direction is vertical) or line (if direction is horizontal), and be recycled?
...	Further args.

**Details**

This function evenly (ish) divides up your string into these colours in the order they appear in colours.

It cannot be used with RGUI (R.app on some systems).

**Value**

A string if type is "string", or coloured text if type is "message" or "warning"

**Examples**

```
## Not run:
multi_colour()

multi_colour("ahoy")

multi_colour("taste the rainbow",
             c("mediumpurple",
               "rainbow",
               "cyan3"))

multi_colour(colours = c(rgb(0.1, 0.2, 0.5),
                          "yellow",
                          rgb(0.2, 0.9, 0.1)))

multi_colour(
  cowsay::animals[["buffalo"]],
  c("mediumorchid4", "dodgerblue1", "lemonchiffon1"))

multi_colour(cowsay::rms, sample(colours(), 10))

# Mystery Bulgarian animal
multi_colour(things[[sample(length(things), 1)]],
             c("white", "darkgreen", "darkred"),
             direction = "horizontal")
```

```
# Mystery Italian animal
multi_colour(things[[sample(length(things), 1)]],
             c("darkgreen", "white", "darkred"),
             direction = "vertical")

## End(Not run)
```

---

nix\_first\_newline      *Remove the first instance of a newline from a string*

---

### Description

Remove the first instance of a newline from a string

### Usage

```
nix_first_newline(s)
```

### Arguments

s                      (character) A string

### Value

A string with the first instance of a newline removed.

### Examples

```
nix_first_newline("onetwo\nthree\nfour")

# Nothing to remove
nix_first_newline("fivesixseven")
```

---

starwars\_intro      *Star Wars a New Hope Intro*

---

### Description

The intro to Episode IV, for use in multicoloring experiments

### Usage

```
starwars_intro
```

### Format

An object of class character of length 1.



---

things	<i>Things</i>
--------	---------------

---

**Description**

Named vector of animals and other characters e.g. Yoda, from the cowsay package

**Usage**

```
things
```

**Format**

An object of class list of length 43.

**Details**

things is a named character list of ASCII animals and characters.

**Examples**

```
things[["turkey"]]
things[["rms"]] %>% cat()
names(things)
multi_color(things[["stretchycat"]]) # To say something, use the cowsay package
```

---

triangle_string	<i>Turn strings into triangle-shaped strings</i>
-----------------	--

---

**Description**

Turn strings into triangle-shaped strings

**Usage**

```
triangle_string(string, maxlen = 1, step = 1, display = FALSE)
```

**Arguments**

string	(character) Some text to reshape into a triangle.
maxlen	(integer) Width of top of triangle. Defaults to 1. Set larger than 1 for downward-pointing triangle.
step	(integer) Number of characters to expand or contract triangle width per line (set to negative for downward-pointing triangle)
display	(logical) Returns string invisibly if FALSE (default), set to TRUE to display returned string



# Index

## \*Topic **datasets**

- starwars\_intro, 8
- things, 9

center\_string, 2

crawl, 3

insert\_rainbow, 4

multi\_color, 3, 5

multi\_colour, 6

multicolor\_logo, 4

nix\_first\_newline, 8

starwars\_intro, 8

things, 9

triangle\_string, 9