Package: colorize (via r-universe)

May 23, 2026

Title Render Text in Color for Markdown/Quarto Documents

Description Provides some simple functions for printing text in color in 'markdown' or 'Quarto' documents, to be rendered as HTML or LaTeX. This is useful when writing about the use of colors in graphs or tables, where you want to print their names in their actual color to give a direct impression of the color, like "red" shown in red, or "blue" shown in blue.

Version 0.2.1

Date 2025-11-24

Depends R (>= 4.1.0)

Imports colorspace, knitr

Suggests quarto, glue, crayon

License MIT + file LICENSE

Encoding UTF-8

VignetteBuilder quarto

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Language en-US

URL https://github.com/friendly/colorize

BugReports https://github.com/friendly/colorize/issues

Repository https://friendly.r-universe.dev

Date/Publication 2025-12-16 14:57:11 UTC

RemoteUrl https://github.com/friendly/colorize

RemoteRef HEAD

RemoteSha b0e4a9d12ae7fb935a48c164d829483df504218b

Contents

`colorbox` *Create a Color Box in a Markdown/Quarto Document*

Description

The `colorbox()` function renders text in a colored background for Markdown / Quarto documents. For documents to be rendered in HTML, ...

Usage

```
colorbox(
  text,
  color = text,
  maincolor = "black",
  bgcolor = "white",
  format = NULL,
  padding = 2
)
```

Arguments

`text`	character string with the text to display.
`color`	R color specification. The default is to use `text`, expecting it to be a named color. Beyond this `color` can be any of the three kinds of R colors, i.e., either a color name (an element of `colors`), a hexadecimal (hex) string of the form `"#rrggbb"` or `"#rrggbbaa"` (see `rgb`), or an integer `i` meaning `palette()[i]`. See also `adjust_transparency` which is used internally to convert all colors to hex strings of the form `"#rrggbb"`.
`maincolor`	R color specification (see above) for the main text color in the Quarto document (default: black).
`bgcolor`	R color specification (see above) for the background color in the Quarto document (default: white).
`format`	character specification of the output format. Currently, `"latex"` (default if `is_latex_output`), `"html"` (default if `is_html_output`, and `"text"` (otherwise) are supported where the latter just contains the input `text` without any color box formatting.
`padding`	integer. Amount of padding (in pixels) for the color box if `format = "html"`.

Details

The function `colorbox()` is a small convenience function that sets up a color box in LaTeX (via `\colorbox{...}{...}`) or HTML (via `<span style=...>...</span>`) for a given text. The main convenience is that the function assesses first whether the text in the color box has a better contrast when written in the main font color (default: black) or the background color (default: white). This is

decided based on the `contrast_ratio()` function from the `colorspace` package using the APCA algorithm.

Additionally, the package canonicalizes all R color specifications to hex color codes (of the form `#rrggbb`) which can subsequently be employed easily in both LaTeX and HTML output. The default output format inside Quarto documents processed via `knitr` is determined using the functions `is_latex_output` and `is_html_output`, respectively.

This illustrates the use of `colorbox()` in a Quarto document:

```
I'll be using the penguins data quite a lot, so it is useful to set up custom colors like
those used in @fig-penguin-species. My versions are shown in @fig-peng-colors with their
color codes. These are shades of:

* `r colorbox("Adélie", "orange")`: `r colorbox("orange", "orange")`,
* `r colorbox("Chinstrap", "purple")`: `r colorbox("purple", "purple")`, and
* `r colorbox("Gentoo", "green")`: `r colorbox("green", "green")`.
```

Value

A character string with the input text along with markup for a background color box.

Author(s)

Achim Zeileis

Examples

```
## basic usage: just supply a color name and let both the background color
## and the output format be decided automatically (the latter does not work
## correctly outside of a quarto/knitr document and hence just returns the text)
colorbox("red")

## emulate behavior in a quarto/knitr document with HTML or LaTeX output
colorbox("red", format = "latex")
colorbox("red", format = "html")

## instead of fully saturated red in sRGB, employ other flavors of red:
## - color 2 in R's default palette
## - color #D55E00 from the Okabe-Ito palette
colorbox("red", color = 2, format = "html")
colorbox("red", color = "#D55E00", format = "html")

## by default, either black or white is used for the text color
## (which ever has the better contrast) but alternatively a different
## main font color (say gray10) and a different background color (say cornsilk)
## can be used for the text
colorbox("red", color = 2, maincolor = "gray10", bgcolor = "cornsilk", format = "html")
colorbox("yellow", color = 7, maincolor = "gray10", bgcolor = "cornsilk", format = "html")
```

`colorize` *Render text in color for Markdown / Quarto documents*

Description

The `colorize()` function renders text in color for Markdown / Quarto documents. For documents to be rendered in HTML, it uses a CSS `<span>`; for documents to be converted to LaTeX and rendered as PDF, it uses `\textcolor{}{}` from the xcolor package.

Usage

```
colorize(text, color, format = NULL)

colorize_bg(text, color, format = NULL)
```

Arguments

`text`	Text to display, a character string
`color`	Color to use, a valid color designation; if missing, use `text` as the color
`format`	character specification of the output format. Currently supported formats are: `"latex"` (default if `is_latex_output`), `"html"` (default if `is_html_output`, and `"text"` (otherwise) are supported where the latter just contains the input `text` without any color formatting.

Details

A companion function, `colorize_bg()` does the same, but uses the specified color for the background.

The names of colors Note that a color name not defined in the xcolor package will trigger a LaTeX error. e.g., `darkgreen` is not defined but you can use: `\definecolor{darkgreen}{RGB}{1,50,32}` in a document to be rendered to PDF.

For inline text, in running text, you can use:

```
`r colorize("Gentoo", "orange")` and `r colorize("Adelie", "purple")` are Penguins.
The `r colorize("red")` points and the `r colorize("blue")` points are nice
```

In a chunk, you can also define variables with the names of colors, for ease of use:

```
red <- colorize('red')
pink <- colorize("pink")
blue <- colorize('blue')
green <- colorize("green")
yellow <- colorize("yellow")
```

Then, these can be used directly in strings, interpolated by `glue::glue()`. This is particularly useful in figure captions.

Value

A character string with the input text along with markup for color encoding.

Author(s)

Michael Friendly

Examples

```
# show what the generated text looks like when encoded for either HTML or LaTeX
colorize("red", format = "html") |> cat()
colorize("red", format = "latex") |> cat()

colorize_bg("blue", format = "html") |> cat()
colorize_bg("blue", format = "latex") |> cat()


 red <- colorize('red')
 blue <- colorize('blue')
 green <- colorize("green")
 glue::glue("There are {red} points, {blue} points and {green} points")
```

Index