Package 'prismatic'

December 1, 2019

Title Color Manipulation Tools

Version 0.2.0

Description Manipulate and visualize colors in a intuitive, low-dependency and functional way.

License MIT + file LICENSE

URL https://github.com/EmilHvitfeldt/prismatic

BugReports https://github.com/EmilHvitfeldt/prismatic/issues

Depends R (>= 3.2)

Imports farver (>= 2.0.1)

Suggests covr, crayon, testthat (>= 2.1.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.0.1.9000

NeedsCompilation no

Author Emil Hvitfeldt [aut, cre] (<https://orcid.org/0000-0002-0679-1945>)

Maintainer Emil Hvitfeldt <emilhhvitfeldt@gmail.com>

Repository CRAN

Date/Publication 2019-12-01 21:40:02 UTC

R topics documented:

`check_color_blindness` *Visualize color vision deficiency*

Description

Visualize color vision deficiency

Usage

```
check_color_blindness(col)
```

Arguments

`col`	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i]. This function will showcase the effect of all 3 kinds of color vision deficiency at the same time side by side.

Value

Nothing

Examples

```
check_color_blindness(rainbow(10))

check_color_blindness(terrain.colors(10))
```

`clr_alpha` *Sets alpha in color*

Description

Sets alpha in color

Usage

```
clr_alpha(col, alpha = 0.5)
```

Arguments

col	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
alpha	Numeric between 0 and 1. 0 will result in full transparency and 1 results in no transparency.

Value

a colors object

Examples

```
plot(clr_alpha(rainbow(10), 0.5))

plot(clr_alpha(rainbow(10), 0.2))

plot(clr_alpha(rainbow(10), seq(0, 1, length.out = 10)))
```

---

clr_darken *Make a color more dark*

Description

Make a color more dark

Usage

```
clr_darken(col, shift = 0.5, space = c("HSL"))
```

Arguments

col	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
shift	Numeric between 0 and 1, 0 will do zero darkening, 1 will do complete darkening turning the color to black. Defaults to 0.5.
space	character string specifying the color space in which adjustment happens. Can be either "HLS", "HCL" or "combined". Defaults to "HSL".

Details

The colors will be trainsformed to HSL color space (hue, saturation, lightness) where the lightness of the color will be modified. The lightness of a color takes a value between 0 and 1, with 0 being black and 1 being white. The `shift` argument takes a value between 0 and 1, where 0 means that the lightness stays unchanged and 1 means completely black. As an example, if the lightness of the color is 0.6 and shift is 0.5, then the lightness be set to the halfway point between 0.6 and 0, which is 0.3.

If space = "HSL" then the colors are transformed to HSL space where the lightness value L is adjusted. If space = "HCL" then the colors are transformed to Cylindrical HCL space where the luminance value L is adjusted. If space = "combined" then the colors are tranformed into HSL and Cylindrical HCL space. Where the color adjusting is happening HLS is copied tto the values in the HCL tranformation. Thus the "combined" transformation adjusts the luminance in HCL space and chroma in HSL space. For more information regarding use of color spaces, please refer to eh colorspace paper https://arxiv.org/abs/1903.06490.

Value

a color object of same length as col.

Source

https://en.wikipedia.org/wiki/HSL_and_HSV

https://en.wikipedia.org/wiki/CIELUV

https://arxiv.org/abs/1903.06490

See Also

clr_lighten

Examples

```
# Using linear shift
plot(clr_darken(rep("red", 11), shift = seq(0, 1, 0.1)))

# Using exponential shifts
plot(clr_darken(rep("red", 11), shift = log(seq(1, exp(1), length.out = 11))))
```

---

clr_desaturate *Make a color more desaturated*

Description

Make a color more desaturated

Usage

```
clr_desaturate(col, shift = 0.5)
```

Arguments

col	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
shift	Numeric between 0 and 1, 0 will do zero desaturation, 1 will do complete desaturation. Defaults to 0.5.

Details

The colors will be trainsformed to HSL color space (hue, saturation, lightness) where the saturation of the color will be modified. The saturation of a color takes a value between 0 and 1, with 0 being black and 1 being white. The `shift` argument takes a value between 0 and 1, where 0 means that the saturation stays unchanged and 1 means completely desaturated. As an example, if the saturation of the color is 0.6 and shift is 0.5, then the saturation be set to the halfway point between 0.6 and 0 which is 0.3.

Value

a colors object of same length as col.

Source

https://en.wikipedia.org/wiki/HSL_and_HSV

See Also

clr_saturate

Examples

```
plot(clr_desaturate(terrain.colors(10), shift = 0.5))

plot(clr_desaturate(terrain.colors(10), shift = 0.9))

plot(clr_desaturate(rep("firebrick", 11), shift = seq(0, 1, 0.1)))
```

---

clr_grayscale *Transform colors to greyscale*

Description

This function has a selection of different methods to turn colors into grayscale.

Usage

```
clr_grayscale(
  col,
  method = c("luma", "averaging", "min_decomp", "max_decomp", "red_channel",
    "green_channel", "blue_channel")
)

clr_greyscale(
  col,
  method = c("luma", "averaging", "min_decomp", "max_decomp", "red_channel",
    "green_channel", "blue_channel")
)
```

Arguments

col	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
method	character string specifying the grayscaling method. Can be one of "luma", "averaging", "min_decomp", "max_decomp", "red_channel", "green_channel" and "blue_channel". Defaults to "luma".

Details

if method = "averaging" then the red, green and blue have been averaged together to create the grey value. This method does a poor job of representing the way the human eye sees color. If method = "luma" (the default) then then a weighted average is used to calculate the grayscale values. The BT. 709 method from the ITU Radiocommunication Sector have determined the weights. It method = "min_decomp" or method = "max_decomp", then a decomposition method is used where the minimum or maximum color value have been selected for the color value. So the color rgb(60, 120, 40) would have the min_decomp value of 40 and max_decomp value of 120. If method is "red_channel", "green_channel" or "blue_channel", then the corresponding color channel been selected for the values of grayscale.

Value

a colors object of same length as col.

Source

https://www.tannerhelland.com/3643/grayscale-image-algorithm-vb6/

https://en.wikipedia.org/wiki/Luma

Examples

```
plot(clr_grayscale(rainbow(10)))

plot(clr_grayscale(terrain.colors(10)))
```

```
plot(clr_grayscale(hcl.colors(10), method = "luma"))
plot(clr_grayscale(hcl.colors(10), method = "averaging"))
plot(clr_grayscale(hcl.colors(10), method = "min_decomp"))
plot(clr_grayscale(hcl.colors(10), method = "max_decomp"))
plot(clr_grayscale(hcl.colors(10), method = "red_channel"))
plot(clr_grayscale(hcl.colors(10), method = "green_channel"))
plot(clr_grayscale(hcl.colors(10), method = "blue_channel"))
```

`clr_lighten` *Make a color more light*

Description

Make a color more light

Usage

```
clr_lighten(col, shift = 0.5, space = c("HSL", "HCL", "combined"))
```

Arguments

`col`	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
`shift`	Numeric between 0 and 1, 0 will do zero lightening, 1 will do complete lightening turning the color to white. Defaults to 0.5.
`space`	character string specifying the color space in which adjustment happens. Can be either "HLS", "HCL" or "combined". Defaults to "HSL".

Details

The colors will be trainsformed to HSL color space (hue, saturation, lightness) where the lightness of the color will be modified. The lightness of a color takes a value between 0 and 1, with 0 being black and 1 being white. The `shift` argument takes a value between 0 and 1, where 0 means that the lightness stays unchanged and 1 means completely white. As an example, if the lightness of the color is 0.6 and shift is 0.5, then the lightness be set to the halfway point between 0.6 and 1 which is 0.8.

If space = "HSL" then the colors are transformed to HSL space where the lightness value L is adjusted. If space = "HCL" then the colors are transformed to Cylindrical HCL space where the luminance value L is adjusted. If space = "combined" then the colors are tranformed into HSL and Cylindrical HCL space. Where the color adjusting is happening HLS is copied tto the values in the HCL tranformation. Thus the "combined" transformation adjusts the luminance in HCL space and chroma in HSL space. For more information regarding use of color spaces, please refer to eh colorspace paper https://arxiv.org/abs/1903.06490.

Value

a colors object of same length as col.

Source

https://en.wikipedia.org/wiki/HSL_and_HSV

https://en.wikipedia.org/wiki/CIELUV

https://arxiv.org/abs/1903.06490

See Also

clr_darken

Examples

```
# Using linear shift
plot(clr_lighten(rep("red", 11), shift = seq(0, 1, 0.1)))
plot(clr_lighten(rep("red", 11), shift = seq(0, 1, 0.1), space = "HCL"))

# Using exponential shifts
plot(clr_lighten(rep("red", 11), shift = log(seq(1, exp(1), length.out = 11))))
```

---

clr_mix *Mixes a color into*

Description

Mixes a color into

Usage

```
clr_mix(col, mix_in, ratio = 0.5)
```

Arguments

col	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
mix_in	A single color any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
ratio	Numeric between 0 and 1. 0 will result on no mixing. 1 results in all the colors turning to mix_in. Must be of length 1 or same length as col.

Value

a colors object

Examples

```
plot(clr_mix(rainbow(10), "blue"))

plot(clr_mix(rainbow(10), "red"))

plot(clr_mix(rainbow(10), "#5500EE"))

plot(clr_mix(rainbow(10), "black", seq(1, 0, length.out = 10)))
```

`clr_negate` *Negates colors in RGB space*

Description

Negates colors in RGB space

Usage

```
clr_negate(col)
```

Arguments

`col`	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].

Details

The negation of color is happening in the red-green-blue colorspace RGB. Meaning that if we take the specification for Orange which is rgb(255, 165, 0), then we negate by taking the oppesite number on the scale from 0 to 255, leaving us wih rgb(0, 90, 255) which is a shade of blue.

Value

a colors object of same length as col.

Examples

```
terr <- color(terrain.colors(10))

terr
clr_negate(terr)

plot(terr)
plot(clr_negate(terr))
```

`clr_protan` *Simulate color vision deficiency*

Description

Simulate color vision deficiency

Usage

```
clr_protan(col, severity = 1)

clr_deutan(col, severity = 1)

clr_tritan(col, severity = 1)
```

Arguments

`col`	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
`severity`	A numeric, Severity of the color vision defect, a number between 0 and 1. 0 means no deficiency, 1 means complete deficiency. Defaults to 1.

Details

The matrices uses to perform transformations have been taken as the 1.0 value in table 1 in http://www.inf.ufrgs.br/~oliveira/pubs_files/CVD_Simulation/CVD_Simulation.html.

Value

a colors object of same length as col.

Source

http://www.inf.ufrgs.br/~oliveira/pubs_files/CVD_Simulation/CVD_Simulation.html

References

Gustavo M. Machado, Manuel M. Oliveira, and Leandro A. F. Fernandes "A Physiologically-based Model for Simulation of Color Vision Deficiency". IEEE Transactions on Visualization and Computer Graphics. Volume 15 (2009), Number 6, November/December 2009. pp. 1291-1298.

Examples

```
rainbow_colors <- color(rainbow(10))

plot(clr_protan(rainbow_colors))
plot(clr_deutan(rainbow_colors))
plot(clr_tritan(rainbow_colors))

viridis_colors <- color(hcl.colors(10, palette = "viridis"))

plot(clr_protan(viridis_colors))
plot(clr_deutan(viridis_colors))
plot(clr_tritan(viridis_colors))
```

---

clr_rotate *Rotate the colors around the hue wheel*

Description

Rotate the colors around the hue wheel

Usage

```
clr_rotate(col, degrees = 0)
```

Arguments

col	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
degrees	Numeric between 0 and 360, denotion the about of degrees the colors should be rotated. Now defaults to 0.

Details

The colors will be trainsformed to HSL color space (hue, saturation, lightness) where the hue of the color will be rotation.

Value

a colors object of same length as col.

Source

https://en.wikipedia.org/wiki/HSL_and_HSV

Examples

```
plot(clr_rotate(terrain.colors(10)))

plot(clr_rotate(terrain.colors(10), degrees = 90))

plot(clr_rotate(terrain.colors(10), degrees = 180))

plot(clr_rotate(rep("magenta", 11), degrees = seq(0, 360, length.out =  11)))
```

---

clr_saturate *Make a color more saturated*

Description

Make a color more saturated

Usage

```
clr_saturate(col, shift = 0.5)
```

Arguments

col	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].
shift	Numeric between 0 and 1, 0 will do zero saturation, 1 will do complete saturation. Defaults to 0.5.

Details

The colors will be trainsformed to HSL color space (hue, saturation, lightness) where the saturation of the color will be modified. The saturation of a color takes a value between 0 and 1, with 0 being black and 1 being white. The `shift` argument takes a value between 0 and 1, where 0 means that the saturation stays unchanged and 1 means completely saturated. As an example, if the saturation of the color is 0.6 and shift is 0.5, then the saturation be set to the halfway point between 0.6 and 1 which is 0.8.

Value

a color object of same length as col.

Source

https://en.wikipedia.org/wiki/HSL_and_HSV

See Also

clr_desaturate

Examples

```
plot(clr_saturate(terrain.colors(10), shift = 0.5))

plot(clr_saturate(terrain.colors(10), shift = 1))

plot(clr_saturate(rep("firebrick", 11), shift = seq(0, 1, 0.1)))
```

`color` *Turn vector to color vector*

Description

Turn vector to color vector

Usage

```
color(col)

colour(col)
```

Arguments

`col`	a color object or vector of any of the three kinds of R color specifications, i.e., either a color name (as listed by colors()), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or a positive integer i meaning palette()[i].

Details

Alpha values will be automatically added to hexcodes. If none at present it will default to no alpha (FF).

Value

a colors object.

Examples

```
terrain_10 <- color(terrain.colors(10))

terrain_10[1:4]

plot(terrain_10)

plot(terrain_10, labels = TRUE)

plot(color(gray.colors(10)), labels = TRUE)
```

`is_color` *Test if the object is a color*

Description

Test if the object is a color

Usage

```
is_color(x)
```

Arguments

`x`	An object

Value

TRUE if the object inherits from the color class.

Index