Package 'colorspace'

December 14, 2016

Version 1.3-2

Date 2016-12-14

Title Color Space Manipulation

Description Carries out mapping between assorted color spaces including RGB, HSV, HLS, CIEXYZ, CIELUV, HCL (polar CIELUV), CIELAB and polar CIELAB. Qualitative, sequential, and diverging color palettes based on HCL colors are provided along with an interactive palette picker (with either a Tcl/Tk or a shiny GUI).

Depends R (>= 2.13.0), methods

Imports graphics, grDevices

Suggests datasets, stats, utils, KernSmooth, MASS, kernlab, mvtnorm, vcd, dichromat, tcltk, shiny, shinyjs

License BSD_3_clause + file LICENSE

URL https://hclwizard.org/

LazyData yes

NeedsCompilation yes

Author Ross Ihaka [aut],
Paul Murrell [aut],
Kurt Hornik [aut],
Jason C. Fisher [aut],
Reto Stauffer [aut],
Achim Zeileis [aut, cre]

Maintainer Achim Zeileis <Achim.Zeileis@R-project.org>

Repository CRAN

Date/Publication 2016-12-14 23:28:25

R topics documented:

`choose_palette` *Graphical User Interface for Choosing HCL Color Palettes*

Description

A graphical user interface (GUI) for viewing, manipulating, and choosing HCL color palettes.

Usage

```
choose_palette(pal = diverge_hcl, n = 7L, parent = NULL, gui = "tcltk")
hclwizard(n = 7L, gui = "shiny", shiny.trace = FALSE)
```

Arguments

`pal`	function; the initial palette, see 'Value' below. Only used if `gui="tcltk"`.
`n`	integer; the initial number of colors in the palette.
`parent`	tkwin; the GUI parent window. Only used if `gui="tcltk"`.
`gui`	character; GUI to use. Available options are `tcltk` and `shiny`, see 'Details' below.
`shiny.trace`	boolean, default `FALSE`. Used for debugging if `gui="shiny"`.

Details

Computes palettes based on the HCL (hue-chroma-luminance) color model (as implemented by `polarLUV`). The GUIs interface the palette functions `rainbow_hcl` for qualitative palettes, `sequential_hcl` for sequential palettes with a single hue, `heat_hcl` for sequential palettes with multiple hues, and `diverge_hcl` for diverging palettes (composed from two single-hue sequential palettes).

Two different GUIs are implemented and can be selected using the function input argument `gui` (`"tcltk"` or `"shiny"`). Both GUIs allows for interactive modification of the arguments of the respective palette-generating functions, i.e., starting/ending hue (wavelength, type of color), minimal/maximal chroma (colorfulness), minimal maximal luminance (brightness, amount of gray), and a power transformations that control how quickly/slowly chroma and/or luminance are changed through the palette. Subsets of the parameters may not be applicable depending on the type of palette chosen. See `rainbow_hcl` and Zeileis et al. (2009) for a more detailed explanation of the different arguments. Stauffer et al. (2015) provide more examples and guidance.

Optionally, active palette can be illustrated by using a range of examples such as a map, heatmap, scatter plot, perspective 3D surface etc.

To demonstrate different types of deficiencies, the active palette may be desaturated (emulating printing on a grayscale printer) and, if the `dichromat` package is available, collapsed to emulate different types of color-blindness (without red-green or green-blue contrasts).

Value

Returns a palette-generating function with the selected arguments. Thus, the returned function takes an integer argument and returns the corresponding number of HCL colors by traversing HCL space through interpolation of the specified hue/chroma/luminance/power values.

Author(s)

Jason C. Fisher, Reto Stauffer, Achim Zeileis

References

Zeileis A, Hornik K, Murrell P (2009). Escaping RGBland: Selecting Colors for Statistical Graphics. *Computational Statistics & Data Analysis*, **53**, 3259–3270. doi: 10.1016/j.csda.2008.11.033 Preprint available from https://eeecon.uibk.ac.at/~zeileis/papers/Zeileis+Hornik+Murrell-2009.pdf.

Stauffer R, Mayr GJ, Dabernig M, Zeileis A (2015). Somewhere over the Rainbow: How to Make Effective Use of Colors in Meteorological Visualizations. *Bulletin of the American Meteorological Society*, **96**(2), 203–216. doi: 10.1175/BAMS-D-13-00155.1

See Also

`rainbow_hcl`

Examples

```
if(interactive()) {
## Using tcltk GUI
## Analog to: hclwizard(gui="tcltk")
```

```
pal <- choose_palette()
## Using shiny GUI
## Analog to: choose_palette(gui = "shiny")
pal <- hclwizard()

## use resulting palette function
filled.contour(volcano, color.palette = pal, asp = 1)
}
```

---

color-class *Class "color"*

Description

Objects from the class *color* represent colors in a number of color spaces. In particular, there are subclasses of color which correspond to RGB, HSV, HLS, CIEXYZ, CIELUV, CIELAB and polar versions of the last two spaces.

Objects from the Class

Objects can be created by calls to the functions `RGB`, `sRGB`, `HSV`, `HLS`, `XYZ`, `LUV`, `LAB`, `polarLUV`, and `polarLAB`. These are all subclasses of the virtual class *color*.

Slots

`coords`**:** An object of class `"matrix"`.

Methods

[`signature(x = "color")`: This method makes it possible to take subsets of a vector of colors.

coerce `signature(from = "color", to = "RGB")`: convert a color vector to RGB.

coerce `signature(from = "color", to = "sRGB")`: convert a color vector to sRGB.

coerce `signature(from = "color", to = "XYZ")`: convert a color vector to XYZ.

coerce `signature(from = "color", to = "LAB")`: convert a color vector to LAB.

coerce `signature(from = "color", to = "polarLAB")`: convert a color vector to polarLAB.

coerce `signature(from = "color", to = "HSV")`: convert a color vector to HSV.

coerce `signature(from = "color", to = "HLS")`: convert a color vector to HLS.

coerce `signature(from = "color", to = "LUV")`: convert a color vector to LUV.

coerce `signature(from = "color", to = "polarLUV")`: convert a color vector to polarLUV.

coords `signature(color = "color")`: extract the color coordinates from a color vector.

plot `signature(x = "color")`: plot a color vector

show `signature(object = "color")`: show a color vector.

Author(s)

Ross Ihaka

See Also

RGB, XYZ, HSV, HLS, LAB, polarLAB, LUV, polarLUV, mixcolor.

Examples

```
x = RGB(runif(1000),runif(1000),runif(1000))
plot(as(x, "LUV"))
```

---

coords *Extract the numerical coordinates of a color*

Description

This function returns a matrix with three columns which give the coordinates of a color in its natural color space.

Usage

```
coords(color)
```

Arguments

color	A color.

Value

A numeric matrix giving the coordinates of the color.

Author(s)

Ross Ihaka.

See Also

RGB, XYZ, LAB, polarLAB, LUV, polarLUV, mixcolor.

Examples

```
x <- RGB(1, 0, 0)
coords(as(x, "HSV"))
```

desaturate *Desaturate Colors by Chroma Removal in HCL Space*

Description

Transform a vector of given colors to the corresponding colors with chroma removed (collapsed to zero) in HCL space.

Usage

```
desaturate(col)
```

Arguments

col	vector of any of the three kind of R colors, i.e., either a color name (an element of colors), a hexadecimal string of the form "#rrggbb" or "#rrggbbaa" (see rgb), or an integer i meaning palette()[i].

Details

Given colors are first transformed to RGB (either using hex2RGB or col2rgb) and then to HCL (polarLUV). In HCL, chroma is removed (i.e., collapsed to zero) and then the color is transformed back to a hexadecimal string.

Value

A character vector with (s)RGB codings of the colors in the palette.

Author(s)

Achim Zeileis

See Also

polarLUV, hex

Examples

```
## rainbow of colors and their desaturated counterparts
rainbow_hcl(12)
desaturate(rainbow_hcl(12))

## convenience demo function
wheel <- function(col, radius = 1, ...)
  pie(rep(1, length(col)), col = col, radius = radius, ...)

## compare base and colorspace palettes
## (in color and desaturated)
par(mar = rep(0, 4), mfrow = c(2, 2))
```

```
## rainbow color wheel
wheel(rainbow_hcl(12))
wheel(rainbow(12))
wheel(desaturate(rainbow_hcl(12)))
wheel(desaturate(rainbow(12)))
```

---

hex *Convert Colors To Hexadecimal Strings*

Description

This functions converts "color" objects into hexadecimal strings.

Usage

```
hex(from, gamma = NULL, fixup = FALSE)
```

Arguments

from	The color object to be converted.
gamma	Deprecated.
fixup	Should the color be corrected to a valid RGB value before correction. The default is to convert out-of-gamut colors to the string "NA".

Details

The color objects are first converted to sRGB color objects. They are then multiplied by 255 and rounded to obtain an integer value. These values are then converted to hexadecimal strings of the form "#RRGGBB" and suitable for use as color descriptions for R graphics. Out of gamut values are either corrected to valid RGB values by translating the the individual primary values so that they lie between 0 and 255.

Value

A vector of character strings.

Author(s)

Ross Ihaka

See Also

hex2RGB, RGB, sRGB, HSV, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
hsv = HSV(seq(0, 360, length = 7)[-7], 1, 1)
hsv
hex(hsv)
barplot(rep(1,6), col = hex(hsv))
```

hex2RGB *Convert Hexadecimal Color Specifications To RGB Objects*

Description

This function takes a vector of strings of the form "#RRGGBB" (hexadecimal color descriptions) into RGB objects.

Usage

```
hex2RGB(x, gamma = FALSE)
```

Arguments

x	a vector of hexadecimal color descriptions.
gamma	Whether to apply gamma-correction.

Details

This function converts device dependent color descriptions of the form "#RRGGBB" into sRGB color descriptions (linearized if gamma is TRUE). The alpha channel will be ignored if given ("#RRGGBBAA").

Value

An RGB object describing the colors.

Author(s)

Ross Ihaka

See Also

hex, RGB, sRGB, HSV, XYZ, polarLAB, LUV, polarLUV.

Examples

```
hex2RGB(c("#FF0000","#00FF00", "#0000FF50"))
```

HLS *Create HLS Colors*

Description

This function creates colors of class HLS; a subclass of the virtual "color" class.

Usage

```
HLS(H, L, S, names)
```

Arguments

H,L,S	These arguments give the hue, lightness, and saturation of the colors. The values can be provided in separate H, L and S vectors or in a three-column matrix passed as H.
names	A vector of names for the colors (by default the row names of H are used).

Details

This function creates colors in the HLS color space which corresponds to the standard sRGB color space (IEC standard 61966). The hues should lie between between 0 and 360, and the lightness and saturations should lie between 0 and 1.

Value

An object of class "HLS" which inherits from class "color."

Author(s)

Ross Ihaka

References

www.srgb.com

See Also

RGB, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
# A rainbow of full-intensity hues
HLS(seq(0, 360, length=13)[-13], 0.5, 1)
```

HSV *Create HSV Colors*

Description

This function creates colors of class HSV; a subclass of the virtual "color" class.

Usage

```
HSV(H, S, V, names)
```

Arguments

H,S,V	These arguments give the hue, saturation and value of the colors. The values can be provided in separate H, S and V vectors or in a three-column matrix passed as H.
names	A vector of names for the colors (by default the row names of H are used).

Details

This function creates colors in the HSV color space which corresponds to the standard sRGB color space (IEC standard 61966). The hues should lie between between 0 and 360, and the saturations and values should lie between 0 and 1.

Value

An object of class "HSV" which inherits from class "color."

Author(s)

Ross Ihaka

References

www.srgb.com

See Also

RGB, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
# A rainbow of full-intensity hues
HSV(seq(0, 360, length=13)[-13], 1, 1)
```

LAB *Create LAB Colors*

Description

This function creates colors of class "LAB"; a subclass of the virtual "color" class.

Usage

```
LAB(L, A, B, names)
```

Arguments

L,A,B	these arguments give the L, A and B coordinates of the colors. The values can be provided in separate L, A and B vectors or in a three-column matrix passed as L.
names	a vector of names for the colors (by default the row names of L are used).

Details

The L, A and B values give the coordinates of the colors in the CIE $L^*a^*b^*$ space. This is a transformation of the 1931 CIE XYZ space which attempts to produce perceptually based axes. Luminance takes values between 0 and 100, and the other coordinates take values between -100 and 100. The a and b coordinates measure positions on green/red and blue/yellow axes.

Value

An object of class "LAB" which inherits from class "color."

Author(s)

Ross Ihaka

See Also

RGB, HSV, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
## Show the LAB space
set.seed(1)
x <- RGB(runif(1000), runif(1000), runif(1000))
y <- as(x, "LAB")
head(x)
head(y)
plot(y)
```

LUV *Create LUV Colors*

Description

This function creates colors of class "LUV"; a subclass of the virtual "color" class.

Usage

```
LUV(L, U, V, names)
```

Arguments

L,U,V	these arguments give the L, U and V coordinates of the colors. The values can be provided in separate L, U and V vectors or in a three-column matrix passed as L.
names	a vector of names for the colors (by default the row names of L are used).

Details

The L, U and V values give the coordinates of the colors in the CIE (1976) $L^*u^*v^*$ space. This is a transformation of the 1931 CIE XYZ space which attempts to produce perceptually based axes. Luminance takes values between 0 and 100, and the other coordinates take values between -100 and 100. The a and b coordinates measure positions on green/red and blue/yellow axes.

Value

An object of class "LUV" which inherits from class "color."

Author(s)

Ross Ihaka

See Also

RGB, HSV, XYZ, LAB, polarLAB, polarLUV.

Examples

```
## Show the LUV space
set.seed(1)
x <- RGB(runif(1000), runif(1000), runif(1000))
y <- as(x, "LUV")
head(x)
head(y)
plot(y)
```

`mixcolor` *Compute the convex combination of two colors*

Description

This function can be used to compute the result of color mixing (it assumes additive mixing).

Usage

```
mixcolor(alpha, color1, color2, where = class(color1))
```

Arguments

`alpha`	The mixed color is obtained by combining an amount `1-alpha` of `color1` with an amount `alpha` of `color2`.
`color1`	The first color.
`color2`	The second color.
`where`	The color space where the mixing is to take place.

Value

The mixed color. This is in the color space specified by `where`.

Author(s)

Ross Ihaka

See Also

`RGB`, `HSV`, `XYZ`, `LAB`, `polarLAB`, `LUV`, `polarLUV`.

Examples

```
mixcolor(0.5, RGB(1, 0, 0), RGB(0, 1, 0))
```

polarLAB *Create polarLAB Colors*

Description

This function creates colors of class "polarLAB"; a subclass of the virtual "color" class.

Usage

```
polarLAB(L, C, H, names)
```

Arguments

L,C,H	these arguments give the L, C and H coordinates of the colors. The values can be provided in separate L, C and H vectors or in a three-column matrix passed as L.
names	A vector of names for the colors (by default the row names of L are used).

Details

The polarLAB space is a transformation of the CIE $L^*a^*b^*$ space so that the a and b values are converted to polar coordinates. The radial component C measures chroma and the angular coordinate H is measures hue.

Value

An object of class "polarLAB" which inherits from class "color."

Author(s)

Ross Ihaka

See Also

RGB, HSV, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
## Show the polarLAB space
set.seed(1)
x <- RGB(runif(1000), runif(1000), runif(1000))
y <- as(x, "polarLAB")
head(x)
head(y)
plot(y)
```

polarLUV *Create polarLUV Colors*

Description

This function creates colors of class "polarLUV"; a subclass of the virtual "color" class.

Usage

```
polarLUV(L, C, H, names)
```

Arguments

L,C,H	these arguments give the L, C and H coordinates of the colors. The values can be provided in separate L, C and H vectors or in a three-column matrix passed as L.
names	A vector of names for the colors (by default the row names of L are used).

Details

The polarLUV space is a transformation of the CIE $L^*u^*u^*$ space so that the u and u values are converted to polar coordinates. The radial component C measures chroma and the angular coordinate H is measures hue.

Value

An object of class "polarLUV" which inherits from class "color."

Author(s)

Ross Ihaka

See Also

RGB, HSV, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
## Show the polarLUV space
set.seed(1)
x <- RGB(runif(1000), runif(1000), runif(1000))
y <- as(x, "polarLUV")
head(x)
head(y)
plot(y)
```

rainbow_hcl *HCL and HSV Color Palettes*

Description

Color palettes based on the HCL and HSV color spaces.

Usage

```
rainbow_hcl(n, c = 50, l = 70, start = 0, end = 360*(n-1)/n,
  gamma = NULL, fixup = TRUE, alpha = 1, ...)

sequential_hcl(n, h = 260, c. = c(80, 0), l = c(30, 90), power = 1.5,
  gamma = NULL, fixup = TRUE, alpha = 1, ...)
heat_hcl(n, h = c(0, 90), c. = c(100, 30), l = c(50, 90), power = c(1/5, 1),
  gamma = NULL, fixup = TRUE, alpha = 1, ...)
terrain_hcl(n, h = c(130, 0), c. = c(80, 0), l = c(60, 95), power = c(1/10, 1),
  gamma = NULL, fixup = TRUE, alpha = 1, ...)

diverge_hcl(n, h = c(260, 0), c = 80, l = c(30, 90), power = 1.5,
  gamma = NULL, fixup = TRUE, alpha = 1, ...)
diverge_hsv(n, h = c(240, 0), s = 1, v = 1, power = 1,
  gamma = NULL, fixup = TRUE, alpha = 1, ...)
```

Arguments

n	the number of colors (≥ 1) to be in the palette.
c, c.	chroma value in the HCL color description.
l	luminance value in the HCL color description.
start	the hue at which the rainbow begins.
end	the hue at which the rainbow ends.
h	hue value in the HCL or HSV color description, has to be in [0, 360] for HCL and in [0, 1] for HSV colors.
s	saturation value in the HSV color description.
v	value value in the HSV color description.
power	control parameter determining how chroma and luminance should be increased (1 = linear, 2 = quadratic, etc.).
gamma	Deprecated.
fixup	logical. Should the color be corrected to a valid RGB value before correction?
alpha	numeric vector of values in the range [0, 1] for alpha transparency channel (0 means transparent and 1 means opaque).
...	Other arguments passed to hex.

Details

All functions compute palettes based on either the HCL (polarLUV) or the HSV (HSV) color space.

rainbow_hcl computes a rainbow of colors (qualitative palette) defined by different hues given a single value of each chroma and luminance. It corresponds to rainbow which computes a rainbow in HSV space.

sequential_hcl gives a sequential palette starting at the full color HCL(h, c[1], l[1]) through to a light color HCL(h, c[2], l[2]) by interpolation.

diverge_hcl and diverge_hsv, compute a set of colors diverging from a neutral center (gray or white, without color) to two different extreme colors (blue and red by default). This is similar to cm.colors. For the diverging HSV colors, two hues h are needed, a maximal saturation s and a fixed value v. The saturation is then varied to obtain the diverging colors. For the diverging HCL colors, again two hues h are needed, a maximal chroma c and two luminances l. The colors are then created by an interpolation between the full color HCL(h[1], c, l[1]), a neutral color HCL(h, 0, l[2]) and the other full color HCL(h[2], c, l[1]).

The palette heat_hcl gives an implementation of heat.colors in HCL space. By default, it goes from a red to a yellow hue, while simultaneously going to lighter colors (i.e., increasing luminance) and reducing the amount of color (i.e., decreasing chroma). The terrain_hcl palette simply calls heat_hcl with different parameters, providing colors similar in spirit to terrain.colors. The lighter colors are not strictly HCL colors, though.

Value

A character vector with (s)RGB codings of the colors in the palette.

Author(s)

Achim Zeileis

References

Zeileis A, Hornik K, Murrell P (2009). Escaping RGBland: Selecting Colors for Statistical Graphics. *Computational Statistics & Data Analysis*, **53**, 3259–3270. doi: 10.1016/j.csda.2008.11.033 Preprint available from https://eeecon.uibk.ac.at/~zeileis/papers/Zeileis+Hornik+Murrell-2009.pdf.

Stauffer R, Mayr GJ, Dabernig M, Zeileis A (2015). Somewhere over the Rainbow: How to Make Effective Use of Colors in Meteorological Visualizations. *Bulletin of the American Meteorological Society*, **96**(2), 203–216. doi: 10.1175/BAMS-D-13-00155.1

See Also

polarLUV, HSV, hex

Examples

```
## convenience demo functions
wheel <- function(col, radius = 1, ...)
  pie(rep(1, length(col)), col = col, radius = radius, ...)
```

```
pal <- function(col, border = "light gray")
{
  n <- length(col)
  plot(0, 0, type="n", xlim = c(0, 1), ylim = c(0, 1), axes = FALSE, xlab = "", ylab = "")
  rect(0:(n-1)/n, 0, 1:n/n, 1, col = col, border = border)
}

## qualitative palette
wheel(rainbow_hcl(12))

## a few useful diverging HCL palettes
par(mar = rep(0, 4), mfrow = c(4, 1))
pal(diverge_hcl(7))
pal(diverge_hcl(7, h = c(246, 40), c = 96, l = c(65, 90)))
pal(diverge_hcl(7, h = c(130, 43), c = 100, l = c(70, 90)))
pal(diverge_hcl(7, h = c(180, 70), c = 70, l = c(90, 95)))
pal(diverge_hcl(7, h = c(180, 330), c = 59, l = c(75, 95)))
pal(diverge_hcl(7, h = c(128, 330), c = 98, l = c(65, 90)))
pal(diverge_hcl(7, h = c(255, 330), l = c(40, 90)))
pal(diverge_hcl(7, c = 100, l = c(50, 90), power = 1))

## sequential palettes
pal(sequential_hcl(12))
pal(heat_hcl(12, h = c(0, -100), l = c(75, 40), c = c(40, 80), power = 1))
pal(terrain_hcl(12, c = c(65, 0), l = c(45, 95), power = c(1/3, 1.5)))
pal(heat_hcl(12, c = c(80, 30), l = c(30, 90), power = c(1/5, 1.5)))

## compare base and colorspace palettes
## (in color and desaturated)
par(mar = rep(0, 4), mfrow = c(2, 2))
## rainbow color wheel
wheel(rainbow_hcl(12))
wheel(rainbow(12))
wheel(desaturate(rainbow_hcl(12)))
wheel(desaturate(rainbow(12)))

## diverging red-blue colors
pal(diverge_hsv(7))
pal(diverge_hcl(7, c = 100, l = c(50, 90)))
pal(desaturate(diverge_hsv(7)))
pal(desaturate(diverge_hcl(7, c = 100, l = c(50, 90))))

## diverging cyan-magenta colors
pal(cm.colors(7))
pal(diverge_hcl(7, h = c(180, 330), c = 59, l = c(75, 95)))
pal(desaturate(cm.colors(7)))
pal(desaturate(diverge_hcl(7, h = c(180, 330), c = 59, l = c(75, 95))))

## heat colors
pal(heat.colors(12))
pal(heat_hcl(12))
pal(desaturate(heat.colors(12)))
pal(desaturate(heat_hcl(12)))
```

```
## terrain colors
pal(terrain.colors(12))
pal(terrain_hcl(12))
pal(desaturate(terrain.colors(12)))
pal(desaturate(terrain_hcl(12)))
```

---

readhex *Read Hexadecimal Color Descriptions*

Description

This function reads a set of hexadecimal color descriptions from a file and creates a color object containing the corresponding colors.

Usage

```
readhex(file = "", class = "RGB")
```

Arguments

file	The file containing the color descriptions.
class	The kind of color object to be returned.

Details

The file is assumed to contain hexadecimal color descriptions of the form #RRGGBB.

Value

An color object of the specified class containing the color descriptions.

Author(s)

Ross Ihaka.

See Also

writehex, readRGB, hex2RGB, RGB, HSV, XYZ, LAB, polarLAB, LUV, polarLUV,

Examples

```
## Not run:
rgb <- readhex("pastel.txt")
hsv <- readhex("pastel.txt", "HSV")

## End(Not run)
```

readRGB *Read RGB Color Descriptions*

Description

This function reads a set of RGB color descriptions (of the form written by `gcolorsel`) from a file and creates a color object containing the corresponding colors.

Usage

```
readRGB(file = "", class = "RGB")
```

Arguments

`file`	The file containing the color descriptions.
`class`	The kind of color object to be returned.

Details

The file is assumed to contain RGB color descriptions consisting of three integer values in the range from 0 to 255 followed by a color name.

Value

An color object of the specified class containing the color descriptions.

Author(s)

Ross Ihaka.

See Also

`writehex`, `readhex`, `hex2RGB`, `RGB`, `HSV`, `XYZ`, `LAB`, `polarLAB`, `LUV`, `polarLUV`.

Examples

```
## Not run:
rgb <- readRGB("pastel.rgb")
hsv <- readRGB("pastel.rgb", "HSV")

## End(Not run)
```

RGB *Create RGB Colors*

Description

This function creates colors of class RGB; a subclass of the virtual "color" class.

Usage

```
RGB(R, G, B, names)
```

Arguments

R,G,B	these arguments give the red, green and blue intensities of the colors (the values should lie between 0 and 1). The values can be provided in separate R, G and B vectors or in a three-column matrix passed as R.
names	A vector of names for the colors (by default the row names of R are used).

Details

This function creates colors in the linearized sRGB color space (IEC standard 61966).

Value

An object of class "RGB" which inherits from class "color."

Author(s)

Ross Ihaka

References

www.srgb.com

See Also

sRGB, HSV, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
# Create a random set of colors
set.seed(1)
RGB(R = runif(20), G = runif(20), B = runif(20))
```

specplot *Color Spectrum Plot*

Description

Visualization of color palettes (given as hex codes) in RGB and/or HCL coordinates.

Usage

```
specplot(x, rgb = TRUE, hcl = TRUE, fix = TRUE, cex = 1,
  type = "l", lwd = 2 * cex, lty = 1, pch = NULL,
  legend = TRUE, palette = TRUE, plot = TRUE)
```

Arguments

x	character vector containing color hex codes.
rgb	logical or color specification. Should the RGB spectrum be visualized? Can also be a vector of three colors for the R/G/B coordinates.
hcl	logical or color specification. Should the HCL spectrum be visualized? Can also be a vector of three colors for the H/C/L coordinates.
fix	logical. Should the hues be fixed to be on a smooth(er) curve? For details see below.
cex	numeric. Character extension for figure axes and labels.
type, lwd, lty, pch	plotting parameters passed to lines for drawing the RGB and HCL coordinates, respectively. Can be vectors of length 3.
legend	logical. Should legends for the coordinates be plotted?
palette	logical. Should the given palette x be plotted?
plot	logical. Should the RGB and/or HCL coordinates be plotted?

Details

The function specplot transforms a given color palette in hex codes into their RGB (sRGB) or HCL (polarLUV) coordinates. As the hues for low-chroma colors are not (or poorly) identified, by default a smoothing is applied to the hues (fix = TRUE). Also, to avoid jumps from 0 to 360 or vice versa, the hue coordinates are shifted suitably.

By default (plot = TRUE) the resulting RGB and HCL coordinates are visualized by simple line plots along with the color palette x itself.

Value

specplot invisibly returns a list with components

RGB	a matrix of sRGB coordinates,
HCL	a matrix of HCL coordinates,
hex	original color palette x.

Author(s)

Reto Stauffer, Achim Zeileis

References

Zeileis A, Hornik K, Murrell P (2009). Escaping RGBland: Selecting Colors for Statistical Graphics. *Computational Statistics & Data Analysis*, **53**, 3259–3270. doi: 10.1016/j.csda.2008.11.033 Preprint available from https://eeecon.uibk.ac.at/~zeileis/papers/Zeileis+Hornik+Murrell-2009.pdf.

Stauffer R, Mayr GJ, Dabernig M, Zeileis A (2015). Somewhere over the Rainbow: How to Make Effective Use of Colors in Meteorological Visualizations. *Bulletin of the American Meteorological Society*, **96**(2), 203–216. doi: 10.1175/BAMS-D-13-00155.1

See Also

rainbow_hcl

Examples

```
## spectrum of the (in)famous RGB rainbow palette
specplot(rainbow(100))

## spectrum of HCL-based palettes: qualitative/sequential/diverging
specplot(rainbow_hcl(100))
specplot(sequential_hcl(100))
specplot(diverge_hcl(100))

## return computed RGB and HCL coordinates
res <- specplot(rainbow(10), plot = FALSE)
print(res)
```

---

sRGB *Create sRGB Colors*

Description

This function creates colors of class sRGB; a subclass of the virtual "color" class.

Usage

```
sRGB(R, G, B, names)
```

Arguments

R,G,B	these arguments give the red, green and blue intensities of the colors (the values should lie between 0 and 1). The values can be provided in separate R, G and B vectors or in a three-column matrix passed as R.
names	A vector of names for the colors (by default the row names of R are used).

Details

This function creates colors in the standard sRGB color space (IEC standard 61966).

Value

An object of class "sRGB" which inherits from class "color."

Author(s)

Ross Ihaka

References

www.srgb.com

See Also

RGB, HSV, XYZ, LAB, polarLAB, LUV, polarLUV.

Examples

```
# Create a random set of colors
set.seed(1)
sRGB(R = runif(20), G = runif(20), B = runif(20))
```

USSouthPolygon *Polygon for County Map of US South States: Alabama, Georgia, and South Carolina*

Description

County polygons for Alabama, Georgia, and South Carolina plus an artifical variable used for coloring.

Usage

```
data("USSouthPolygon")
```

Format

A data frame with coordinates of the vertices of the county polygons (x, y) and an artificial variable z constructed for illustrating colored maps.

Source

Polygon data taken from **maps** package of Becker, Wilks, Brownrigg, and Minka (2012). Version 2.2-6. https://CRAN.R-project.org/package=maps

Examples

```
## generate color palette
pal <- diverge_hcl(9)
n <- length(pal)

## draw shaded polygons
plot(0, 0, type = "n", xlab = "", ylab = "", xaxt = "n", yaxt = "n", bty = "n",
  xlim = c(-88.5, -78.6), ylim = c(30.2, 35.2), asp = 1)
polygon(USSouthPolygon, col = pal[cut(na.omit(USSouthPolygon$z), breaks = 0:n/n)])
```

`writehex` *Write Hexadecimal Color Descriptions*

Description

Given a color object, this function writes a file containing the hexadecimal representation of the colors in the object.

Usage

```
writehex(x, file = "")
```

Arguments

`x`	a color object.
`file`	the name of the file to be written.

Details

This function converts the given color object to RGB and then writes hexadecimal strings (of the form `#RRGGBB`) representing the colors to the specified file.

Value

The name of the file is returned as the value of the function.

Author(s)

Ross Ihaka

See Also

`readhex`, `readRGB`, `hex2RGB`, `RGB`, `HSV`, `XYZ`, `LAB`, `polarLAB`, `LUV`, `polarLUV`.

Examples

```
set.seed(1)
x <- RGB(runif(10), runif(10), runif(10))
writehex(x, "random.txt")
```

XYZ *Create XYZ Colors*

Description

This function creates colors of class XYZ; a subclass of the virtual "color" class.

Usage

```
XYZ(X, Y, Z, names)
```

Arguments

X,Y,Z	these arguments give the X, Y and Z coordinates of the colors. The values can be provided in separate X, Y and Z vectors or in a three-column matrix passed as X.
names	A vector of names for the colors (by default the row names of X are used).

Details

The X, Y and Z values are the levels of the CIE primaries. These are scaled so that the luminance of the display white-point is 100. The white-point is taken to be D65, which means that its coordinates are 95.047, 100.000, 108.883.

Value

An object of class "XYZ" which inherits from class "color."

Author(s)

Ross Ihaka

See Also

RGB, HSV, LAB, polarLAB, LUV, polarLUV.

Examples

```
## Generate white in XYZ space
XYZ(95.047, 100.000, 108.883)
```

Index