Package 'colortheme'

November 27, 2026

Type Package

Title A Unified Framework and Interface for Color Palettes and Ramps

Version 0.1.0

Description
The 'colortheme' package provides a unified, environment-based object framework for managing, retrieving, and customizing color palettes and color ramps in R. It acts as a flexible dispatcher that supports both discrete qualitative palettes and continuous sequential/diverging color ramps. Features a compact URL-like syntax for on-the-fly theme modifications (such as color reversal, type coercion, and query-based parameter tuning) and seamless integration with base R graphics and 'ggplot2'.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports graphics, grDevices, stats, utils

Suggests ggplot2, khroma, RColorBrewer, scales, testthat (>= 3.0.0), viridisLite

URL https://github.com/ryo-asashi/colortheme, https://ryo-asashi.github.io/colortheme/

BugReports https://github.com/ryo-asashi/colortheme/issues

Config/testthat/edition 3

Config/Needs/website rmarkdown, DT, htmltools, patchwork

NeedsCompilation no

Author Ryoichi Asashiba [aut, cre] (ORCID: <https://orcid.org/0009-0001-9532-7000>)

Maintainer Ryoichi Asashiba <ryoichi.asashiba@gmail.com>

Repository CRAN

Date/Publication 2026-11-27 18:50:02 UTC

Contents

as_palette 2
colortheme 3
colorthemes 6
map_to_colors 7
scale_color_theme 8

Index **11**

`as_palette` *Extract Palette or Ramp Function*

Description

`as_palette()` and `as_ramp()` extract the underlying discrete palette function or continuous color ramp function from a color theme or color kernel.

Usage

```
as_palette(x, ...)

## S3 method for class 'colortheme'
as_palette(x, ...)

## Default S3 method:
as_palette(x, ...)

as_ramp(x, ...)

## S3 method for class 'colortheme'
as_ramp(x, ...)

## Default S3 method:
as_ramp(x, ...)
```

Arguments

`x`	a `"colortheme"` object, a theme name (character string), a color vector, or a color-generating function.
`...`	additional arguments passed to `colortheme` to modify the theme before extracting the function.

Value

- `as_palette()` returns a palette function taking an integer `n` and returning `n` colors.
- `as_ramp()` returns a ramp function taking a numeric vector `x` in `[0, 1]` and returning corresponding colors.

See Also

colortheme

Examples

```
pal <- as_palette("Okabe-Ito")
pal(8L)

ramp <- as_ramp("Viridis")
ramp(c(0, 0.5, 1.0))
```

---

colortheme *Color Themes for Graphics*

Description

The colortheme() function is the main interface for working with "colortheme" objects. It acts as a generic dispatcher that can retrieve a pre-defined theme by name, create a new theme from a vector of colors or a color-generating function, and modify an existing "colortheme" object.

Usage

```
colortheme(object, ...)

## S3 method for class 'colortheme'
colortheme(
  object,
  kernel.args = NULL,
  options = NULL,
  name = NULL,
  source = NULL,
  type = NULL,
  reverse = FALSE,
  ...
)

## S3 method for class 'character'
colortheme(
  object,
  kernel.args = list(),
  options = list(),
  name = NULL,
  source = NULL,
  type = NULL,
  reverse = FALSE,
  env = NULL,
  ...
```

```
)

## Default S3 method:
colortheme(
  object,
  kernel.args = list(),
  options = list(),
  name = NULL,
  source = NULL,
  type = NULL,
  reverse = FALSE,
  ...
)

## S3 method for class 'colortheme'
plot(x, n = NULL, text = x$name, ...)

## S3 method for class 'colortheme'
print(x, display = TRUE, ...)
```

Arguments

object	a character string to retrieve a pre-defined theme, a color kernel (vector of colors or color-generating function) to create a new theme, or a "colortheme" object to modify.
...	optional named arguments used to modify the color theme (overriding corresponding entries in options or kernel.args).
kernel.args	a list of arguments passed to the color kernel.
options	a list of option values controlling the color theme's behavior.
name	a character string for the color theme name.
source	a character string for the source name of the color theme.
type	a character string specifying the type of the color theme: "sequential", "diverging", or "qualitative".
reverse	logical. If TRUE, the order of colors is reversed.
env	an environment where color themes are registered.
x	a "colortheme" object for print() and plot().
n	optional integer specifying the number of colors to display in plot().
text	optional character string for the title displayed in plot().
display	logical. If TRUE (default), print() also plots the color swatch.

Details

A "colortheme" object is a specialized environment that provides two primary color-generating functions: $palette() and $ramp().

`$palette(n)` takes an integer `n` and returns a character vector of `n` discrete colors. It is primarily intended for qualitative themes, where distinct colors represent categorical data.

`$ramp(x)` takes a numeric vector `x` with values in the `[0, 1]` interval and returns a character vector of corresponding colors. It maps numeric values onto a continuous color gradient, making it suitable for sequential and diverging themes.

`colortheme()` dispatches based on the class of `object`:

- If `object` is a character string (e.g., `"Viridis"`, `"grDevices/RdBu_r@q?alpha=.5"`), it is parsed according to the theme name syntax and loaded from the registry.
- If `object` is a color kernel (a character vector of colors, a palette function, or a ramp function), a new color theme is constructed.
- If `object` is already a `"colortheme"` object, a modified copy is returned with updated settings.

Value

`colortheme()` returns a `"colortheme"` object (an environment with class `c("colortheme", "environment")`) containing `$palette()` and `$ramp()` functions along with theme metadata.

Theme Name Syntax

When retrieving a registered theme by string, you can use a compact URL-like syntax to specify the source and apply on-the-fly modifications:

`"[(source)/](name)[_r][@(type)][?(query)]"`

- `source`: (optional) the source package or collection (e.g., `"grDevices"`).
- `name`: the name of the theme (e.g., `"RdBu"`).
- `"_r"`: (optional) suffix to reverse the color order.
- `type`: (optional) desired theme type, matched against `"sequential"`, `"diverging"`, or `"qualitative"` (abbreviations like `"s"`, `"d"`, `"q"` or `"seq"`, `"div"`, `"qual"` are supported).
- `query`: (optional) key-value pairs separated by `;` or `&` (e.g., `"?alpha=0.5;na.color='gray50'"`) to override theme metadata, `options`, or `kernel.args`.

See Also

`colorthemes`, `register_colortheme`, `as_palette`, `map_to_colors`, `scale_color_theme`

Examples

```
# Retrieve a pre-defined theme
ct <- colortheme("Mako")
print(ct)
ct$palette(5L)
ct$ramp(seq.int(0, 1, length.out = 5))

# Use special syntax to get a reversed, qualitative theme with alpha
ct <- colortheme("grDevices/Zissou 1_r@qual?alpha=0.75")
ct$palette(5L)
```

```
# Create a new theme from a vector of colors
ct <- colortheme(c("#003f5c", "#7a5195", "#ef5675", "#ffa600"))
ct$palette(5L)

# Modify an existing theme
ct2 <- colortheme(ct, type = "qualitative", reverse = TRUE)
ct2$palette(5L)
```

`colorthemes` *List and Register Color Themes*

Description

`colorthemes()` returns a data frame listing available color themes in the registry. `register_colortheme()` registers a custom color theme into the registry.

Usage

```
colorthemes(name = NULL, source = NULL, type = NULL, env = NULL)

register_colortheme(
  kernel,
  kernel.args = list(),
  options = list(),
  name = "newtheme",
  source = "custom",
  type = NULL,
  env = NULL
)
```

Arguments

`name`	for `colorthemes()`, a character vector of theme names to match (case-insensitive prefix matching). For `register_colortheme()`, a single string for the new theme name.
`source`	for `colorthemes()`, a character vector of sources to match (case-insensitive prefix matching). For `register_colortheme()`, a single string for the source name.
`type`	a character string or vector specifying the theme type(s): `"sequential"`, `"diverging"`, or `"qualitative"`.
`env`	an environment where color themes are registered. Defaults to the package registry (`getOption("colortheme.registry")`).
`kernel`	a color vector, a palette function, a ramp function, a `"colortheme"` object, or a length-2 character vector/list for lazy evaluation. If `NULL`, the theme is removed.
`kernel.args`	a list of arguments passed to the color kernel.
`options`	a list of option values controlling the color theme's behavior.

Details

`colorthemes()` extracts metadata from the registry environment and structures it into a data frame, allowing users to discover themes by `name`, `source`, and `type` (with case-insensitive prefix matching).

`register_colortheme()` takes a color vector, a color-generating function, or an existing `"colortheme"` object and registers it under a specified `name` and `source` (defaulting to `"custom/newtheme"`). To keep the registry lightweight, the `kernel` argument supports lazy loading: pass a character vector or list of length 2 where the first element is an R expression returning a kernel (e.g., `"rainbow"`) and the second is the namespace (e.g., `"grDevices"`).

Value

- `colorthemes()`: A data frame with columns `"name"`, `"source"`, and `"type"`.
- `register_colortheme()`: Invisibly returns the metadata of the previous theme that was overwritten (or `NULL` if none existed).

See Also

`colortheme`

Examples

```
# List all available themes
head(colorthemes())

# Case-insensitive prefix matching
colorthemes(name = "ma")

# Register a custom theme
register_colortheme(
  kernel = c("#264653", "#2a9d8f", "#e9c46a", "#f4a261", "#e76f51"),
  name = "sand", source = "custom", type = "sequential"
)
colorthemes(name = "sand")

# Remove the custom theme by passing NULL
register_colortheme(NULL, name = "sand", source = "custom")
```

---

map_to_colors *Map Values to Colors using Color Themes*

Description

`map_to_colors()` maps a vector of data values to a vector of hexadecimal color codes based on a specified color theme. It automatically handles both discrete and continuous variables, appropriately scaling values and handling missing data.

Usage

```
map_to_colors(x, theme = NULL, discrete = NULL, middle = 0, na.value = NULL)
```

Arguments

x	a vector of data values (discrete or continuous) to be mapped to colors.
theme	a color theme name (e.g., "Viridis"), a "colortheme" object, a character vector of color names, or a palette/ramp function. See colortheme for details.
discrete	logical. If TRUE, a discrete scale is used regardless of the data type and the theme type.
middle	a numeric value specifying the midpoint for diverging color themes. Default is 0.
na.value	a character string specifying the color for NA values. If NULL (default), falls back to the theme's na.color option.

Value

A character vector of hexadecimal color codes.

See Also

colortheme

Examples

```
# Continuous mapping (Sequential)
plot(cars, col = map_to_colors(cars$speed, "Viridis"), pch = 19)

# Discrete mapping (Qualitative)
plot(iris$Sepal.Length, iris$Sepal.Width,
     col = map_to_colors(iris$Species, "Set 2"), pch = 19)
```

---

scale_color_theme *Color Theme Scales for ggplot2 Graphics*

Description

scale_color_theme() and its family of functions provide a unified interface to apply color themes to the colour and fill aesthetics of ggplot2 objects.

Usage

```
scale_color_theme(
  theme = NULL,
  ...,
  discrete = NULL,
  middle = 0,
  aesthetics = "colour"
)

scale_colour_theme(
  theme = NULL,
  ...,
  discrete = NULL,
  middle = 0,
  aesthetics = "colour"
)

scale_fill_theme(
  theme = NULL,
  ...,
  discrete = NULL,
  middle = 0,
  aesthetics = "fill"
)
```

Arguments

theme	a color theme name (e.g., "Viridis"), a "colortheme" object, a character vector of color names, or a palette/ramp function. See colortheme for details.
...	optional arguments passed to ggplot2::continuous_scale() or ggplot2::discrete_scale().
discrete	logical. If TRUE, a discrete scale is used regardless of the theme type.
middle	a numeric value specifying the midpoint for diverging color themes. Default is 0.
aesthetics	the aesthetic(s) to be scaled ("colour", "color", or "fill").

Details

This function automatically selects the appropriate **ggplot2** scale based on the theme's type:

- If the theme is "qualitative", a discrete scale is used by default (discrete = TRUE).
- If the theme is "sequential" or "diverging", a continuous scale is used by default. Diverging themes use scales::rescale_mid() to center the gradient around middle.

Value

A ggplot2 scale object ("ScaleContinuous" or "ScaleDiscrete").

See Also

colortheme

Examples

```
d <- ggplot2::ggplot(iris, ggplot2::aes(Sepal.Length, Sepal.Width, colour = Species)) +
  ggplot2::geom_point()
d + scale_color_theme("Set 1")
d + scale_color_theme("SunsetDark", discrete = TRUE)
```

Index

as_palette, 2, *5*
as_ramp *(*as_palette*)*, 2

colortheme, *2, 3*, 3, *7–10*
colorthemes, *5*, 6

map_to_colors, *5*, 7

plot.colortheme *(*colortheme*)*, 3
print.colortheme *(*colortheme*)*, 3

register_colortheme, *5*
register_colortheme *(*colorthemes*)*, 6

scale_color_theme, *5*, 8
scale_colour_theme *(*scale_color_theme*)*, 8
scale_fill_theme *(*scale_color_theme*)*, 8