Package: PortinariColors (via r-universe)

October 28, 2026

Type Package

Title Color Palettes Inspired by the Paintings of Candido Portinari

Version 0.3.0

Description Color palettes inspired by paintings of Candido Portinari held by the Sao Paulo Museum of Art (MASP) and by his Guerra (War) and Paz (Peace) panels at the United Nations Headquarters. Provides qualitative, diverging and sequential palettes for discrete and continuous data, ready-made 'ggplot2' color and fill scales, and a colorblind-safety check that simulates protanopia, deuteranopia and tritanopia following Machado, Oliveira and Fernandes (2009) <doi:10.1109/TVCG.2009.113>.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2

URL https://rafalopespx.github.io/PortinariColors/, https://github.com/rafalopespx/PortinariColors

BugReports https://github.com/rafalopespx/PortinariColors/issues

Imports ggplot2 (>= 3.4.0), grDevices, graphics, stats, utils

Suggests spelling, testthat (>= 3.1.5)

Config/testthat/edition 3

Config/Needs/website pkgdown, rmarkdown, knitr

NeedsCompilation no

Author Rafa Lopes [aut, cre]

Maintainer Rafa Lopes <lopes1313@gmail.com>

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

Date/Publication 2026-10-28 11:30:15 UTC

RemoteUrl https://github.com/cran/PortinariColors

RemoteRef HEAD

RemoteSha 5e0526ca1d41b582ef5519c7eb741be2f8fd1cd5

Contents

portinari.cvd.check 2
portinari.pal 3
portinari.palettes 4
portinari.works 4
portinari_scales 5
show_portinari 6

Index **8**

`portinari.cvd.check` *Check colorblind safety of a Portinari palette*

Description

Simulates protanopia, deuteranopia and tritanopia (Machado et al., 2009) and measures the smallest CIELAB color difference (CIE76 dE) within the discrete palette of `n` colors returned by `portinari.pal()`, for every `n`. A discrete palette of `n` colors is colorblind-safe when all three simulations keep that minimum at or above `threshold`.

Usage

```
portinari.cvd.check(palette = "cafe", threshold = 10)
```

Arguments

`palette`	Name of the palette. See `portinari.palettes()`.
`threshold`	Minimum acceptable dE. Default `10`.

Value

A data frame with one row per palette size `n` (from 2 up to the palette length) and columns `normal`, `protanopia`, `deuteranopia` and `tritanopia` (minimum pairwise dE in each condition) and `colorblind_safe`.

Examples

```
portinari.cvd.check("inah")
```

`portinari.pal` *Portinari color palette*

Description

Returns a vector of hex colors from a Portinari palette.

Usage

```
portinari.pal(
  palette = "cafe",
  n,
  type = c("discrete", "continuous"),
  direction = 1,
  colorblind = FALSE
)
```

Arguments

`palette`	Name of the palette. See `portinari.palettes()` for choices.
`n`	Number of colors. Defaults to the length of the palette (or of its colorblind-safe subset when `colorblind = TRUE`).
`type`	`"discrete"` returns distinct colors and errors when `n` is larger than the palette. `"continuous"` interpolates. When omitted, the type is chosen from `n`.
`direction`	`1` (default) for the standard order, `-1` to reverse it.
`colorblind`	If `TRUE`, only colors that remain distinguishable under protanopia, deuteranopia and tritanopia simulation are used. See `portinari.cvd.check()`.

Details

Palettes come in three categories (see `portinari.palettes()`): *qualitative* palettes for unordered groups, *sequential* palettes for ordered values, and *diverging* palettes for values around a meaningful midpoint. Discrete palettes pick `n` colors that suit the category: the most distinct colors for qualitative palettes, evenly spaced steps for sequential palettes, and balanced steps either side of the light center for diverging palettes. Continuous palettes interpolate the full palette in CIELAB space.

Value

A character vector of hex colors with class `"portinari_palette"`.

See Also

`scale_color_portinari()`, `scale_fill_portinari()`, `portinari.palettes()`

Examples

```
portinari.pal("cafe")
portinari.pal("retirantes", n = 3)
portinari.pal("guerra_azul", n = 20, type = "continuous")
portinari.pal("inah", colorblind = TRUE)
```

---

portinari.palettes *List the available Portinari palettes*

Description

List the available Portinari palettes

Usage

```
portinari.palettes(category = NULL)
```

Arguments

`category`	Optional filter: one or more of `"qualitative"`, `"diverging"` and `"sequential"`.

Value

A data frame with one row per palette: its category, the number of colors, how many colors are colorblind-safe (`cvd_n`), whether the whole palette is colorblind-friendly, and a short description. See `portinari.works()` for the paintings behind each palette.

Examples

```
portinari.palettes()
portinari.palettes("diverging")
```

---

portinari.works *Paintings behind the Portinari palettes*

Description

The works each palette was drawn from, with links to the collection pages. The sequential palettes `guerra_azul` and `paz_ouro` are derived from `guerra` and `paz` and share their works.

Usage

```
portinari.works(palette = NULL)
```

Arguments

`palette`	Optional character vector of palette names to filter by.

Value

A data frame with columns `palette`, `title`, `year`, `collection`, `inventory`, `url` (collection or story page) and `image_url` (the museum's image, where one is published; `NA` otherwise).

Examples

```
portinari.works()
portinari.works("retirantes")
```

---

portinari_scales *Portinari color and fill scales for ggplot2*

Description

Discrete or continuous color and fill scales built from a Portinari palette. They are ordinary `ggplot2` scales and accept the usual arguments (`name`, `labels`, `breaks`, `guide`, ...). Match the palette category to the data: qualitative palettes for unordered groups, sequential palettes for ordered values, and diverging palettes for values around a midpoint (see `portinari.palettes()`).

Usage

```
scale_color_portinari(
  palette = "cafe",
  type = c("discrete", "continuous"),
  direction = 1,
  colorblind = FALSE,
  ...
)

scale_colour_portinari(
  palette = "cafe",
  type = c("discrete", "continuous"),
  direction = 1,
  colorblind = FALSE,
  ...
)

scale_fill_portinari(
  palette = "cafe",
  type = c("discrete", "continuous"),
  direction = 1,
  colorblind = FALSE,
  ...
)
```

Arguments

palette	Name of the palette. See portinari.palettes().
type	"discrete" (default) for categorical variables, or "continuous" for numeric variables.
direction	1 (default) or -1 to reverse the palette.
colorblind	If TRUE, only colorblind-safe colors are used.
...	Further arguments passed to ggplot2::discrete_scale() (discrete) or ggplot2::scale_colour_grad / ggplot2::scale_fill_gradientn() (continuous).

Value

A ggplot2 scale object.

See Also

portinari.pal()

Examples

```
library(ggplot2)
# Qualitative: unordered groups
ggplot(mtcars, aes(wt, mpg, colour = factor(cyl))) +
  geom_point(size = 3) +
  scale_color_portinari("cafe", name = "Cylinders")

# Sequential: ordered values
ggplot(faithfuld, aes(waiting, eruptions, fill = density)) +
  geom_raster() +
  scale_fill_portinari("guerra_azul", type = "continuous", direction = -1)

# Diverging: values around a midpoint
ggplot(mtcars, aes(wt, mpg, colour = mpg - mean(mpg))) +
  geom_point(size = 3) +
  scale_color_portinari("guerra", type = "continuous", name = "mpg vs mean")

# Diverging, discrete: an ordered scale with a neutral middle
ggplot(mtcars, aes(factor(gear), fill = factor(carb))) +
  geom_bar() +
  scale_fill_portinari("retirantes")
```

---

show_portinari *Show Portinari palettes*

Description

Draws each palette as a row of color swatches using base graphics, labeled with its category.

Usage

```
show_portinari(palette = NULL, category = NULL)
```

Arguments

palette	Character vector of palette names. Defaults to all palettes (or all palettes in category).
category	Optional filter: one or more of "qualitative", "diverging" and "sequential".

Value

The palette names, invisibly.

Examples

```
show_portinari()
show_portinari(category = "diverging")
show_portinari(c("cafe", "inah"))
```

Index

ggplot2::discrete_scale(), *6*
ggplot2::scale_colour_gradientn(), *6*
ggplot2::scale_fill_gradientn(), *6*

portinari.cvd.check, 2
portinari.cvd.check(), *3*
portinari.pal, 3
portinari.pal(), *2*, *6*
portinari.palettes, 4
portinari.palettes(), *2, 3*, *5, 6*
portinari.works, 4
portinari.works(), *4*
portinari_scales, 5

scale_color_portinari
(portinari_scales), 5
scale_color_portinari(), *3*
scale_colour_portinari
(portinari_scales), 5
scale_fill_portinari
(portinari_scales), 5
scale_fill_portinari(), *3*
show_portinari, 6