Package: INBOtheme (via r-universe)

August 18, 2024

Type Package

Title Themes for ggplot2

Version 0.6.0

Description Several themes for the ggplot2 package. Among others themes complying with the style guide for the Research Institute for Nature and Forest (INBO) and Elsevier journals.

License GPL-3

URL https://github.com/inbo/INBOtheme

BugReports https://github.com/inbo/INBOtheme/issues

Depends R (>= 3.5.0)

Imports assertthat, colorspace, conflicted, ggplot2 (>= 3.4.0), grid, scales

Suggests cluster, colordistance, knitr, rmarkdown, testthat

VignetteBuilder knitr

Config/checklist/communities inbo

Config/checklist/keywords ggplot2; theme; style guide

Encoding UTF-8

Language en-GB

LazyLoad yes

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

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

RemoteUrl https://github.com/inbo/inbotheme

RemoteRef HEAD

RemoteSha efd415e626b2f873b1c1fe872aa55fe7257d250e

Contents

colour_blind_distance 2
colour_distance 3
demo_palette 3
inbo.2015.colours 4
inbo_hoofd 4
inbo_palette 5
nara_palette 6
optimal_order 6
ordinal_palette 7
page_height 7
scale_colour_discrete 8
scale_colour_gradient 8
scale_colour_gradient2 11
scale_colour_viridis_d 14
scale_fill_discrete 17
scale_fill_gradient 18
scale_fill_gradient2 20
show_palette 23
switch_colour 23
theme_elsevier 24
theme_inbo 24
theme_map 25
theme_nara 26
theme_vlaanderen2015 26
traffic_palette 27
vlaanderen_palette 27
vl_yellow 28

Index **30**

colour_blind_distance *Calculate a distance matrix between colours for colour blind people*

Description

The function calculates the distance matrices for every option. Then aggregates the matrices into a single distance matrix.

Usage

```
colour_blind_distance(
  colours,
  deutan = TRUE,
  protan = FALSE,
  tritan = FALSE,
  gray = FALSE,
  fun = min,
  method = "emd"
)
```

Arguments

`colours` a vector of colours

`deutan` correct for deuteranomaly (red - green colour blindness). Defaults to `TRUE`.

`protan` correct for protanopia (red - green colour blindness). Defaults to `FALSE`.

`tritan` correct for tritanopia (blue - yellow colour blindness). Defaults to `FALSE`.

`gray` correct for monochromacy (unable to distinguish colours). Defaults to `FALSE`.

`fun` Function to aggregate different colour distance matrices.

`method` passed to `colordistance::getColorDistanceMatrix()`

`colour_distance` *Calculate a distance matrix between colours*

Description

Calculate a distance matrix between colours

Usage

```
colour_distance(colours, method = "emd")
```

Arguments

`colours` a vector of colours

`method` passed to `colordistance::getColorDistanceMatrix()`

`demo_palette` *Demonstrate colours in a palette*

Description

Create an image with a number of rectangles coloured along the input vector.

Usage

```
demo_palette(colours)
```

Arguments

`colours` a vector of colours

See Also

Other colours: inbo_hoofd, inbo_palette(), nara_palette(), show_palette(), traffic_palette(), vl_yellow, vlaanderen_palette()

---

inbo.2015.colours *Deprecated functions*

Description

These functions will be removed from INBOtheme in the future.

Usage

```
inbo.2015.colours(n)

switchColour(new_colour = inbo_steun_blauw)

vlaanderen.2015.colours(n)
```

Arguments

n	The number of colours
new_colour	The new default colour.

---

inbo_hoofd *Colour according to the INBO style guide version >= 2015*

Description

Colour according to the INBO style guide as hexadecimal values.

- inbo_hoofd
- inbo_steun_donkerroos
- inbo_steun_blauw
- inbo_steun_geelgroen
- inbo_oranje
- inbo_rood
- inbo_bruinrood
- inbo_grijs
- inbo_lichtgrijs
- inbo_grijsblauw
- inbo_lichtblauw

- `inbo_achtergrond`
- `inbo_munt`
- `inbo_lichtgroen`
- `inbo_donkergroen`
- `inbo_groen`
- `inbo_geelgr`
- `inbo_oranje`
- `inbo_geel`
- `inbo_felrood`
- `inbo_groen`
- `inbo_kaki`
- `inbo_oranjerood`
- `inbo_donkerblauw`

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

`theme_inbo2015()`

Other colours: `demo_palette()`, `inbo_palette()`, `nara_palette()`, `show_palette()`, `traffic_palette()`, `vl_yellow`, `vlaanderen_palette()`

`inbo_palette` *A Colour Palette According the INBO Style Guide Version >= 2015.*

Description

A Colour Palette According the INBO Style Guide Version >= 2015.

Usage

```
inbo_palette(n)
```

Arguments

`n`	The number of colours

Value

a vector of n hexadecimal values defining the colours.

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other colours: `demo_palette()`, `inbo_hoofd`, `nara_palette()`, `show_palette()`, `traffic_palette()`, `vl_yellow`, `vlaanderen_palette()`

`nara_palette` *A Colour Palette for NARA reports.*

Description

Deprecated. Use `inbo_palette()` instead.

Usage

```
nara_palette(n)
```

Arguments

`n`	The number of colours

Value

a vector of n hexadecimal values defining the colours.

See Also

Other colours: `demo_palette()`, `inbo_hoofd`, `inbo_palette()`, `show_palette()`, `traffic_palette()`, `vl_yellow`, `vlaanderen_palette()`

`optimal_order` *Order variables to maximise contrasts*

Description

Start with variable which has the largest distance between all other variables. Then add the variable with the largest distance to the already selected variables.

Usage

```
optimal_order(distance, selected = character(0))
```

Arguments

`distance`	a square distance matrix
`selected`	optional vector of preselected variables

Value

a vector variable names

`ordinal_palette` *Palette for ordinal variables*

Description

The colour ramps depends on the active `ggplot2` theme.

Usage

```
ordinal_palette(n)
```

Arguments

`n`	The number of colours

`page_height` *Standard dimensions for the Elsevier style guide*

Description

Several standard dimensions for the Elsevier style guide -`page_height` Maximal height of a figure (in inch). -`page_width` Width of a figure covering two columns (in inch). -`column_width` Width of a figure covering one column (in inch). -`medium_width` Width of a figure covering 1.5 columns (in inch).

See Also

`theme_elsevier()`

scale_colour_discrete *redefine* ggplot2::scale_colour_discrete()

Description

Selects the colours depending on the current set theme.

Usage

```
scale_colour_discrete(..., type)

scale_color_discrete(..., type)
```

Arguments

...	Additional parameters passed on to the scale type,
type	Ignored argument. Only present to match the arguments of ggplot2::scale_fill_discrete()

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other scales: scale_colour_gradient2(), scale_colour_gradient(), scale_colour_viridis_d(), scale_fill_discrete(), scale_fill_gradient2(), scale_fill_gradient()

scale_colour_gradient *redefine* ggplot2::scale_colour_gradient()

Description

Selects the colours depending on the current set theme.

Usage

```
scale_colour_gradient(
  ...,
  low,
  high,
  space = "Lab",
  na.value = "grey50",
  guide = "colourbar",
  aesthetics = "colour"
)
```

```
scale_color_gradient(
  ...,
  low,
  high,
  space = "Lab",
  na.value = "grey50",
  guide = "colourbar",
  aesthetics = "colour"
)
```

Arguments

`...` Arguments passed on to `continuous_scale`

`scale_name` The name of the scale that should be used for error messages associated with this scale.

`palette` A palette function that when called with a numeric vector with values between 0 and 1 returns the corresponding output values (e.g., `scales::area_pal()`).

`name` The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted.

`breaks` One of:

- `NULL` for no breaks
- `waiver()` for the default breaks computed by the transformation object
- A numeric vector of positions
- A function that takes the limits as input and returns breaks as output (e.g., a function returned by `scales::extended_breaks()`). Also accepts rlang lambda function notation.

`minor_breaks` One of:

- `NULL` for no minor breaks
- `waiver()` for the default breaks (one minor break between each major break)
- A numeric vector of positions
- A function that given the limits returns a vector of minor breaks. Also accepts rlang lambda function notation.

`n.breaks` An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if `breaks = waiver()`. Use `NULL` to use the default number of breaks given by the transformation.

`labels` One of:

- `NULL` for no labels
- `waiver()` for the default labels computed by the transformation object
- A character vector giving labels (must be same length as `breaks`)
- An expression vector (must be the same length as breaks). See ?plotmath for details.

- A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation.

`limits` One of:

- `NULL` to use the default scale range
- A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum
- A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).

`rescaler` A function used to scale the input values to the range [0, 1]. This is always `scales::rescale()`, except for diverging and n colour gradients (i.e., `scale_colour_gradient2()`, `scale_colour_gradientn()`). The `rescaler` is ignored by position scales, which always use `scales::rescale()`. Also accepts rlang lambda function notation.

`oob` One of:

- Function that handles limits outside of the scale limits (out of bounds). Also accepts rlang lambda function notation.
- The default (`scales::censor()`) replaces out of bounds values with `NA`.
- `scales::squish()` for squishing out of bounds values into range.
- `scales::squish_infinite()` for squishing infinite values into range.

`trans` For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time".

A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called `<name>_trans` (e.g., `scales::boxcox_trans()`). You can create your own transformation with `scales::trans_new()`.

`expand` For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function `expansion()` to generate the values for the `expand` argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.

`position` For position scales, The position of the axis. `left` or `right` for y axes, `top` or `bottom` for x axes.

`super` The super class to use for the constructed scale

`low` Colour for the low end of the gradient

`high` Colour for the high end of the gradient

`space` colour space in which to calculate gradient. Must be "Lab" - other values are deprecated.

`na.value` Colour to use for missing values

`guide` Type of legend. Use `"colourbar"` for continuous colour bar, or `"legend"` for discrete colour legend.

`aesthetics` Character string or vector of character strings listing the name(s) of the aesthetic(s) that this scale works with. This can be useful, for example, to apply colour settings to the `colour` and `fill` aesthetics at the same time, via `aesthetics = c("colour", "fill")`.

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other scales: `scale_colour_discrete()`, `scale_colour_gradient2()`, `scale_colour_viridis_d()`, `scale_fill_discrete()`, `scale_fill_gradient2()`, `scale_fill_gradient()`

scale_colour_gradient2

redefine `ggplot2::scale_colour_gradient2()`

Description

Selects the colours depending on the current set theme.

Usage

```
scale_colour_gradient2(
  ...,
  low,
  high,
  mid,
  midpoint = 0,
  space = "Lab",
  guide = "colourbar",
  na.value = "grey50",
  aesthetics = "colour"
)

scale_color_gradient2(
  ...,
  low,
  high,
  mid,
  midpoint = 0,
  space = "Lab",
  guide = "colourbar",
  na.value = "grey50",
  aesthetics = "colour"
)
```

Arguments

`...` Arguments passed on to `continuous_scale`

`scale_name` The name of the scale that should be used for error messages associated with this scale.

`palette` A palette function that when called with a numeric vector with values between 0 and 1 returns the corresponding output values (e.g., `scales::area_pal()`).

`name` The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted.

`breaks` One of:

- `NULL` for no breaks
- `waiver()` for the default breaks computed by the transformation object
- A numeric vector of positions
- A function that takes the limits as input and returns breaks as output (e.g., a function returned by `scales::extended_breaks()`). Also accepts rlang lambda function notation.

`minor_breaks` One of:

- `NULL` for no minor breaks
- `waiver()` for the default breaks (one minor break between each major break)
- A numeric vector of positions
- A function that given the limits returns a vector of minor breaks. Also accepts rlang lambda function notation.

`n.breaks` An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if `breaks = waiver()`. Use `NULL` to use the default number of breaks given by the transformation.

`labels` One of:

- `NULL` for no labels
- `waiver()` for the default labels computed by the transformation object
- A character vector giving labels (must be same length as `breaks`)
- An expression vector (must be the same length as breaks). See ?plotmath for details.
- A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation.

`limits` One of:

- `NULL` to use the default scale range
- A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum

- A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).

`rescaler` A function used to scale the input values to the range [0, 1]. This is always `scales::rescale()`, except for diverging and n colour gradients (i.e., `scale_colour_gradient2()`, `scale_colour_gradientn()`). The `rescaler` is ignored by position scales, which always use `scales::rescale()`. Also accepts rlang lambda function notation.

`oob` One of:

- Function that handles limits outside of the scale limits (out of bounds). Also accepts rlang lambda function notation.
- The default (`scales::censor()`) replaces out of bounds values with `NA`.
- `scales::squish()` for squishing out of bounds values into range.
- `scales::squish_infinite()` for squishing infinite values into range.

`trans` For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time".
A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called `<name>_trans` (e.g., `scales::boxcox_trans()`). You can create your own transformation with `scales::trans_new()`.

`expand` For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function `expansion()` to generate the values for the `expand` argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.

`position` For position scales, The position of the axis. `left` or `right` for y axes, `top` or `bottom` for x axes.

`super` The super class to use for the constructed scale

`low`	Colour for the low end of the gradient
`high`	Colour for the high end of the gradient
`mid`	colour for mid point
`midpoint`	The midpoint (in data value) of the diverging scale. Defaults to 0.
`space`	colour space in which to calculate gradient. Must be "Lab" - other values are deprecated.
`guide`	Type of legend. Use `"colourbar"` for continuous colour bar, or `"legend"` for discrete colour legend.
`na.value`	Colour to use for missing values

`aesthetics` Character string or vector of character strings listing the name(s) of the aesthetic(s) that this scale works with. This can be useful, for example, to apply colour settings to the `colour` and `fill` aesthetics at the same time, via `aesthetics = c("colour", "fill")`.

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other scales: `scale_colour_discrete()`, `scale_colour_gradient()`, `scale_colour_viridis_d()`, `scale_fill_discrete()`, `scale_fill_gradient2()`, `scale_fill_gradient()`

---

scale_colour_viridis_d *redefine* ggplot2::scale_colour_viridis_d()

Description

Selects the colours depending on the current set theme.

Usage

```
scale_colour_viridis_d(
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  aesthetics = "colour"
)

scale_fill_viridis_d(
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  aesthetics = "fill"
)
```

Arguments

`...` Arguments passed on to `continuous_scale`

`scale_name` The name of the scale that should be used for error messages associated with this scale.

`palette` A palette function that when called with a numeric vector with values between 0 and 1 returns the corresponding output values (e.g., `scales::area_pal()`).

`name` The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted.

`breaks` One of:

- `NULL` for no breaks
- `waiver()` for the default breaks computed by the transformation object
- A numeric vector of positions
- A function that takes the limits as input and returns breaks as output (e.g., a function returned by `scales::extended_breaks()`). Also accepts rlang lambda function notation.

`minor_breaks` One of:

- `NULL` for no minor breaks
- `waiver()` for the default breaks (one minor break between each major break)
- A numeric vector of positions
- A function that given the limits returns a vector of minor breaks. Also accepts rlang lambda function notation.

`n.breaks` An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if `breaks = waiver()`. Use `NULL` to use the default number of breaks given by the transformation.

`labels` One of:

- `NULL` for no labels
- `waiver()` for the default labels computed by the transformation object
- A character vector giving labels (must be same length as `breaks`)
- An expression vector (must be the same length as breaks). See ?plotmath for details.
- A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation.

`limits` One of:

- `NULL` to use the default scale range
- A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum
- A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).

`rescaler` A function used to scale the input values to the range [0, 1]. This is always `scales::rescale()`, except for diverging and n colour gradients (i.e., `scale_colour_gradient2()`, `scale_colour_gradientn()`). The `rescaler` is ignored by position scales, which always use `scales::rescale()`. Also accepts rlang lambda function notation.

`oob` One of:

- Function that handles limits outside of the scale limits (out of bounds). Also accepts rlang lambda function notation.
- The default (`scales::censor()`) replaces out of bounds values with `NA`.
- `scales::squish()` for squishing out of bounds values into range.
- `scales::squish_infinite()` for squishing infinite values into range.

`trans` For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time".

A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called `<name>_trans` (e.g., `scales::boxcox_trans()`). You can create your own transformation with `scales::trans_new()`.

`expand` For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function `expansion()` to generate the values for the `expand` argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.

`position` For position scales, The position of the axis. `left` or `right` for y axes, `top` or `bottom` for x axes.

`super` The super class to use for the constructed scale

`alpha` The alpha transparency, a number in [0,1], see argument alpha in `hsv`.

`begin, end` The (corrected) hue in `[0,1]` at which the color map begins and ends.

`direction` Sets the order of colors in the scale. If 1, the default, colors are ordered from darkest to lightest. If -1, the order of colors is reversed.

`option` A character string indicating the color map option to use. Eight options are available:

- "magma" (or "A")
- "inferno" (or "B")
- "plasma" (or "C")
- "viridis" (or "D")
- "cividis" (or "E")
- "rocket" (or "F")
- "mako" (or "G")
- "turbo" (or "H")

`aesthetics` Character string or vector of character strings listing the name(s) of the aesthetic(s) that this scale works with. This can be useful, for example, to apply colour settings to the `colour` and `fill` aesthetics at the same time, via `aesthetics = c("colour", "fill")`.

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other scales: `scale_colour_discrete()`, `scale_colour_gradient2()`, `scale_colour_gradient()`, `scale_fill_discrete()`, `scale_fill_gradient2()`, `scale_fill_gradient()`

Other scales: `scale_colour_discrete()`, `scale_colour_gradient2()`, `scale_colour_gradient()`, `scale_fill_discrete()`, `scale_fill_gradient2()`, `scale_fill_gradient()`

`scale_fill_discrete` *redefine* `ggplot2::scale_fill_discrete()`

Description

Selects the colours depending on the current set theme.

Usage

```
scale_fill_discrete(..., type)
```

Arguments

`...` Additional parameters passed on to the scale type,

`type` One of the following:

- A character vector of color codes. The codes are used for a 'manual' color scale as long as the number of codes exceeds the number of data levels (if there are more levels than codes, `scale_colour_hue()`/`scale_fill_hue()` are used to construct the default scale). If this is a named vector, then the color values will be matched to levels based on the names of the vectors. Data values that don't match will be set as `na.value`.
- A list of character vectors of color codes. The minimum length vector that exceeds the number of data levels is chosen for the color scaling. This is useful if you want to change the color palette based on the number of levels.
- A function that returns a discrete colour/fill scale (e.g., `scale_fill_hue()`, `scale_fill_brewer()`, etc).

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other scales: `scale_colour_discrete()`, `scale_colour_gradient2()`, `scale_colour_gradient()`, `scale_colour_viridis_d()`, `scale_fill_gradient2()`, `scale_fill_gradient()`

`scale_fill_gradient` *redefine* `ggplot2::scale_fill_gradient()`

Description

Selects the colours depending on the current set theme.

Usage

```
scale_fill_gradient(
  ...,
  low,
  high,
  space = "Lab",
  na.value = "grey50",
  guide = "colourbar",
  aesthetics = "fill"
)
```

Arguments

`...` Arguments passed on to `continuous_scale`

`scale_name` The name of the scale that should be used for error messages associated with this scale.

`palette` A palette function that when called with a numeric vector with values between 0 and 1 returns the corresponding output values (e.g., `scales::area_pal()`).

`name` The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted.

`breaks` One of:

- `NULL` for no breaks
- `waiver()` for the default breaks computed by the transformation object
- A numeric vector of positions
- A function that takes the limits as input and returns breaks as output (e.g., a function returned by `scales::extended_breaks()`). Also accepts rlang lambda function notation.

`minor_breaks` One of:

- `NULL` for no minor breaks
- `waiver()` for the default breaks (one minor break between each major break)
- A numeric vector of positions

- A function that given the limits returns a vector of minor breaks. Also accepts rlang lambda function notation.

`n.breaks` An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if `breaks = waiver()`. Use `NULL` to use the default number of breaks given by the transformation.

`labels` One of:

- `NULL` for no labels
- `waiver()` for the default labels computed by the transformation object
- A character vector giving labels (must be same length as `breaks`)
- An expression vector (must be the same length as breaks). See ?plotmath for details.
- A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation.

`limits` One of:

- `NULL` to use the default scale range
- A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum
- A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).

`rescaler` A function used to scale the input values to the range [0, 1]. This is always `scales::rescale()`, except for diverging and n colour gradients (i.e., `scale_colour_gradient2()`, `scale_colour_gradientn()`). The `rescaler` is ignored by position scales, which always use `scales::rescale()`. Also accepts rlang lambda function notation.

`oob` One of:

- Function that handles limits outside of the scale limits (out of bounds). Also accepts rlang lambda function notation.
- The default (`scales::censor()`) replaces out of bounds values with `NA`.
- `scales::squish()` for squishing out of bounds values into range.
- `scales::squish_infinite()` for squishing infinite values into range.

`trans` For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time".

A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called `<name>_trans` (e.g., `scales::boxcox_trans()`). You can create your own transformation with `scales::trans_new()`.

`expand` For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function `expansion()` to generate the values for the `expand` argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.

`position` For position scales, The position of the axis. `left` or `right` for y axes, `top` or `bottom` for x axes.

`super` The super class to use for the constructed scale

`low` Colour for the low end of the gradient

`high` Colour for the high end of the gradient

`space` colour space in which to calculate gradient. Must be "Lab" - other values are deprecated.

`na.value` Colour to use for missing values

`guide` Type of legend. Use `"colourbar"` for continuous colour bar, or `"legend"` for discrete colour legend.

`aesthetics` Character string or vector of character strings listing the name(s) of the aesthetic(s) that this scale works with. This can be useful, for example, to apply colour settings to the `colour` and `fill` aesthetics at the same time, via `aesthetics = c("colour", "fill")`.

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other scales: `scale_colour_discrete()`, `scale_colour_gradient2()`, `scale_colour_gradient()`, `scale_colour_viridis_d()`, `scale_fill_discrete()`, `scale_fill_gradient2()`

`scale_fill_gradient2` *redefine* `ggplot2::scale_fill_gradient2()`

Description

Selects the colours depending on the current set theme.

Usage

```
scale_fill_gradient2(
  ...,
  low,
  high,
  mid,
  midpoint = 0,
  space = "Lab",
  guide = "colourbar",
  na.value = "grey50",
  aesthetics = "fill"
)
```

Arguments

`...` Arguments passed on to `continuous_scale`

`scale_name` The name of the scale that should be used for error messages associated with this scale.

`palette` A palette function that when called with a numeric vector with values between 0 and 1 returns the corresponding output values (e.g., `scales::area_pal()`).

`name` The name of the scale. Used as the axis or legend title. If `waiver()`, the default, the name of the scale is taken from the first mapping used for that aesthetic. If `NULL`, the legend title will be omitted.

`breaks` One of:

- `NULL` for no breaks
- `waiver()` for the default breaks computed by the transformation object
- A numeric vector of positions
- A function that takes the limits as input and returns breaks as output (e.g., a function returned by `scales::extended_breaks()`). Also accepts rlang lambda function notation.

`minor_breaks` One of:

- `NULL` for no minor breaks
- `waiver()` for the default breaks (one minor break between each major break)
- A numeric vector of positions
- A function that given the limits returns a vector of minor breaks. Also accepts rlang lambda function notation.

`n.breaks` An integer guiding the number of major breaks. The algorithm may choose a slightly different number to ensure nice break labels. Will only have an effect if `breaks = waiver()`. Use `NULL` to use the default number of breaks given by the transformation.

`labels` One of:

- `NULL` for no labels
- `waiver()` for the default labels computed by the transformation object
- A character vector giving labels (must be same length as `breaks`)
- An expression vector (must be the same length as breaks). See ?plotmath for details.
- A function that takes the breaks as input and returns labels as output. Also accepts rlang lambda function notation.

`limits` One of:

- `NULL` to use the default scale range
- A numeric vector of length two providing limits of the scale. Use `NA` to refer to the existing minimum or maximum
- A function that accepts the existing (automatic) limits and returns new limits. Also accepts rlang lambda function notation. Note that setting limits on positional scales will **remove** data outside of the limits. If the purpose is to zoom, use the limit argument in the coordinate system (see `coord_cartesian()`).

rescaler A function used to scale the input values to the range [0, 1]. This is always scales::rescale(), except for diverging and n colour gradients (i.e., scale_colour_gradient2(), scale_colour_gradientn()). The rescaler is ignored by position scales, which always use scales::rescale(). Also accepts rlang lambda function notation.

oob One of:

- Function that handles limits outside of the scale limits (out of bounds). Also accepts rlang lambda function notation.
- The default (scales::censor()) replaces out of bounds values with NA.
- scales::squish() for squishing out of bounds values into range.
- scales::squish_infinite() for squishing infinite values into range.

trans For continuous scales, the name of a transformation object or the object itself. Built-in transformations include "asn", "atanh", "boxcox", "date", "exp", "hms", "identity", "log", "log10", "log1p", "log2", "logit", "modulus", "probability", "probit", "pseudo_log", "reciprocal", "reverse", "sqrt" and "time".

A transformation object bundles together a transform, its inverse, and methods for generating breaks and labels. Transformation objects are defined in the scales package, and are called <name>_trans (e.g., scales::boxcox_trans()). You can create your own transformation with scales::trans_new().

expand For position scales, a vector of range expansion constants used to add some padding around the data to ensure that they are placed some distance away from the axes. Use the convenience function expansion() to generate the values for the expand argument. The defaults are to expand the scale by 5% on each side for continuous variables, and by 0.6 units on each side for discrete variables.

position For position scales, The position of the axis. left or right for y axes, top or bottom for x axes.

super The super class to use for the constructed scale

low Colour for the low end of the gradient

high Colour for the high end of the gradient

mid colour for mid point

midpoint The midpoint (in data value) of the diverging scale. Defaults to 0.

space colour space in which to calculate gradient. Must be "Lab" - other values are deprecated.

guide Type of legend. Use "colourbar" for continuous colour bar, or "legend" for discrete colour legend.

na.value Colour to use for missing values

aesthetics Character string or vector of character strings listing the name(s) of the aesthetic(s) that this scale works with. This can be useful, for example, to apply colour settings to the colour and fill aesthetics at the same time, via aesthetics = c("colour", "fill").

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other scales: `scale_colour_discrete()`, `scale_colour_gradient2()`, `scale_colour_gradient()`, `scale_colour_viridis_d()`, `scale_fill_discrete()`, `scale_fill_gradient()`

`show_palette` *Show a palette on a single row*

Description

Show a palette on a single row

Usage

```
show_palette(colours)
```

Arguments

`colours`	a vector of colours

See Also

Other colours: `demo_palette()`, `inbo_hoofd`, `inbo_palette()`, `nara_palette()`, `traffic_palette()`, `vl_yellow`, `vlaanderen_palette()`

`switch_colour` *Switch the default colour of all* `ggplot2` `geoms`.

Description

Switch the default colour of all `ggplot2` `geoms`.

Usage

```
switch_colour(new_colour = inbo_steun_blauw)
```

Arguments

`new_colour`	The new default colour.

See Also

Other theme: `theme_elsevier()`, `theme_inbo()`, `theme_map()`, `theme_nara()`, `theme_vlaanderen2015()`

Examples

```
switch_colour("black")
```

theme_elsevier *Theme in compliance with the style guide of Elsevier journals*

Description

This theme is based on http://cdn.elsevier.com/assets/pdf_file/0010/109963/Artwork.pdf

Usage

```
theme_elsevier(base_size = 7, base_family = "")
```

Arguments

base_size	base font size
base_family	base font type

See Also

Other theme: switch_colour(), theme_inbo(), theme_map(), theme_nara(), theme_vlaanderen2015()

Examples

```
library(ggplot2)
p <- ggplot(mtcars, aes(x = mpg, y = drat)) + geom_point()
p.elsevier <- p + theme_elsevier()
```

theme_inbo *The theme in compliance with the INBO style guide version >= 2015.*

Description

The theme in compliance with the INBO style guide version >= 2015.

Usage

```
theme_inbo(base_size = 12, base_family = "", transparent = FALSE)

theme_inbo2015(base_size = 12, base_family, transparent = FALSE)
```

Arguments

`base_size`	base font size, given in pts.
`base_family`	base font family
`transparent`	Make backgrounds transparent. `FALSE`: all backgrounds are white, `TRUE`: all backgrounds are transparent. You can pass a vector to transparent. In that case, it will check whether the values `"plot"`, `"panel"` and/or `"legend"` are present. The according items will be transparent. Transparent panel will use grey instead of white grid lines.

Author(s)

Thierry Onkelinx, Oona Op de Weerdt, Nicole De Groof

See Also

Other theme: `switch_colour()`, `theme_elsevier()`, `theme_map()`, `theme_nara()`, `theme_vlaanderen2015()`

Examples

```
library(ggplot2)
p <- ggplot(mtcars, aes(x = mpg, y = drat)) + geom_point()
p.inbo <- p + theme_inbo()
```

`theme_map` *a `ggplot2` theme which removes labels, ticks and titles from both axes.*

Description

a `ggplot2` theme which removes labels, ticks and titles from both axes.

Usage

```
theme_map()
```

See Also

Other theme: `switch_colour()`, `theme_elsevier()`, `theme_inbo()`, `theme_nara()`, `theme_vlaanderen2015()`

Examples

```
library(ggplot2)
p <- ggplot(mtcars, aes(x = mpg, y = drat)) + geom_point()
p.map <- p + theme_map()
```

theme_nara *The theme for NARA reports.*

Description

Deprecated. Use `theme_inbo()` instead.

Usage

```
theme_nara(base_size = 12, base_family = "")
```

Arguments

`base_size`	base font size, given in pts.
`base_family`	base font family

See Also

Other theme: `switch_colour()`, `theme_elsevier()`, `theme_inbo()`, `theme_map()`, `theme_vlaanderen2015()`

theme_vlaanderen2015 *The theme in compliance with the Flanders style guide version >= 2015.*

Description

The theme in compliance with the Flanders style guide version >= 2015.

Usage

```
theme_vlaanderen2015(base_size = 12, base_family = "", transparent = FALSE)
```

Arguments

`base_size`	base font size, given in pts.
`base_family`	base font family
`transparent`	Make backgrounds transparent. `FALSE`: all backgrounds are white, `TRUE`: all backgrounds are transparent. You can pass a vector to transparent. In that case, it will check whether the values `"plot"`, `"panel"` and/or `"legend"` are present. The according items will be transparent. Transparent panel will use grey instead of white grid lines.

Author(s)

Thierry Onkelinx, Nicole De Groof

See Also

Other theme: switch_colour(), theme_elsevier(), theme_inbo(), theme_map(), theme_nara()

Examples

```
library(ggplot2)
p <- ggplot(mtcars, aes(x = mpg, y = drat)) + geom_point()
p.vl <- p + theme_vlaanderen2015()
```

traffic_palette *A Colour Palette Ranging From a Dark Red over Medium Orange to Light Green.*

Description

A Colour Palette Ranging From a Dark Red over Medium Orange to Light Green.

Usage

```
traffic_palette(n)
```

Arguments

n	The number of colours

Value

a vector of n hexadecimal values defining the colours.

See Also

Other colours: demo_palette(), inbo_hoofd, inbo_palette(), nara_palette(), show_palette(), vl_yellow, vlaanderen_palette()

vlaanderen_palette *A colour pallet according the Flanders style guide version >= 2015.*

Description

A colour pallet according the Flanders style guide version >= 2015.

Usage

```
vlaanderen_palette(n)
```

Arguments

n	The number of colours

Value

a vector of n hexadecimal values defining the colours.

Author(s)

Thierry Onkelinx

References

http://webstijlgids.vlaanderen.be/element/kleurgebruik

See Also

Other colours: `demo_palette()`, `inbo_hoofd`, `inbo_palette()`, `nara_palette()`, `show_palette()`, `traffic_palette()`, `vl_yellow`

`vl_yellow` *Some colours for the corporate style of vlaanderen.be. Taken from level 1, and level 2 palette 7*

Description

Colour according to the Flanders style guide as hexadecimal values.

- `vl_yellow`
- `vl_darkyellow`
- `vl_black`
- `vl_grey1`
- `vl_grey2`
- `vl_grey3`
- `vl_grey4`
- `vl_grey5`
- `vl_grey6`
- `vl_grey7`
- `vl_lightgreen`
- `vl_darkgreen`
- `vl_lightblue`
- `vl_darkblue`
- `vl_lightred`
- `vl_darkred`
- `vl_lightbrown`
- `vl_darkbrown`

Author(s)

Thierry Onkelinx

References

http://webstijlgids.vlaanderen.be/element/kleurgebruik

See Also

Other colours: `demo_palette()`, `inbo_hoofd`, `inbo_palette()`, `nara_palette()`, `show_palette()`, `traffic_palette()`, `vlaanderen_palette()`

Index

∗ **colours**
- demo_palette, 3
- inbo_hoofd, 4
- inbo_palette, 5
- nara_palette, 6
- show_palette, 23
- traffic_palette, 27
- vl_yellow, 28
- vlaanderen_palette, 27

∗ **datasets**
- inbo_hoofd, 4
- page_height, 7
- vl_yellow, 28

∗ **scales**
- scale_colour_discrete, 8
- scale_colour_gradient, 8
- scale_colour_gradient2, 11
- scale_colour_viridis_d, 14
- scale_fill_discrete, 17
- scale_fill_gradient, 18
- scale_fill_gradient2, 20

∗ **theme**
- switch_colour, 23
- theme_elsevier, 24
- theme_inbo, 24
- theme_map, 25
- theme_nara, 26
- theme_vlaanderen2015, 26

∗ **utils**
- page_height, 7

colour_blind_distance, 2
colour_distance, 3
column_width *(page_height)*, 7
continuous_scale, *9*, *12*, *15*, *18*, *21*
coord_cartesian(), *10*, *13*, *15*, *19*, *21*

demo_palette, 3, *5*, *6*, *23*, *27–29*

expansion(), *10*, *13*, *16*, *20*, *22*

hsv, *16*

inbo.2015.colours, 4
inbo_achtergrond *(inbo_hoofd)*, 4
inbo_bruinrood *(inbo_hoofd)*, 4
inbo_donkerblauw *(inbo_hoofd)*, 4
inbo_donkergroen *(inbo_hoofd)*, 4
inbo_felrood *(inbo_hoofd)*, 4
inbo_geel *(inbo_hoofd)*, 4
inbo_geelgr *(inbo_hoofd)*, 4
inbo_grijs *(inbo_hoofd)*, 4
inbo_grijsblauw *(inbo_hoofd)*, 4
inbo_groen *(inbo_hoofd)*, 4
inbo_hoofd, *4*, 4, *6*, *23*, *27–29*
inbo_kaki *(inbo_hoofd)*, 4
inbo_lichtblauw *(inbo_hoofd)*, 4
inbo_lichtgrijs *(inbo_hoofd)*, 4
inbo_lichtgroen *(inbo_hoofd)*, 4
inbo_munt *(inbo_hoofd)*, 4
inbo_oranje *(inbo_hoofd)*, 4
inbo_oranjerood *(inbo_hoofd)*, 4
inbo_palette, *4*, *5*, 5, *6*, *23*, *27–29*
inbo_rood *(inbo_hoofd)*, 4
inbo_steun_blauw *(inbo_hoofd)*, 4
inbo_steun_donkerroos *(inbo_hoofd)*, 4
inbo_steun_geelgroen *(inbo_hoofd)*, 4

lambda, *9*, *10*, *12*, *13*, *15*, *16*, *18*, *19*, *21*, *22*

medium_width *(page_height)*, 7

nara_palette, *4–6*, 6, *23*, *27–29*

optimal_order, 6
ordinal_palette, 7

page_height, 7
page_width *(page_height)*, 7

scale_color_discrete *(scale_colour_discrete)*, 8

scale_color_gradient (scale_colour_gradient), 8
scale_color_gradient2 (scale_colour_gradient2), 11
scale_color_ordinal (scale_colour_viridis_d), 14
scale_color_viridis_d (scale_colour_viridis_d), 14
scale_colour_discrete, 8, *11*, *14*, *17*, *18*, *20*, *23*
scale_colour_gradient, *8*, 8, *14*, *17*, *18*, *20*, *23*
scale_colour_gradient2, *8*, *11*, 11, *17*, *18*, *20*, *23*
scale_colour_gradient2(), *10*, *13*, *16*, *19*, *22*
scale_colour_gradientn(), *10*, *13*, *16*, *19*, *22*
scale_colour_hue(), *17*
scale_colour_ordinal (scale_colour_viridis_d), 14
scale_colour_viridis_d, *8*, *11*, *14*, 14, *18*, *20*, *23*
scale_fill_brewer(), *17*
scale_fill_discrete, *8*, *11*, *14*, *17*, 17, *20*, *23*
scale_fill_gradient, *8*, *11*, *14*, *17*, *18*, 18, *23*
scale_fill_gradient2, *8*, *11*, *14*, *17*, *18*, *20*, 20
scale_fill_hue(), *17*
scale_fill_ordinal (scale_colour_viridis_d), 14
scale_fill_viridis_d (scale_colour_viridis_d), 14
scales::area_pal(), *9*, *12*, *15*, *18*, *21*
scales::boxcox_trans(), *10*, *13*, *16*, *19*, *22*
scales::censor(), *10*, *13*, *16*, *19*, *22*
scales::extended_breaks(), *9*, *12*, *15*, *18*, *21*
scales::rescale(), *10*, *13*, *16*, *19*, *22*
scales::squish(), *10*, *13*, *16*, *19*, *22*
scales::squish_infinite(), *10*, *13*, *16*, *19*, *22*
scales::trans_new(), *10*, *13*, *16*, *19*, *22*
show_palette, *4–6*, 23, *27–29*
switch_colour, 23, *24–27*
switchColour (inbo.2015.colours), 4

theme_elsevier, *23*, 24, *25–27*
theme_inbo, *23*, *24*, 24, *25–27*
theme_inbo2015 (theme_inbo), 24
theme_map, *23–25*, 25, *26*, *27*
theme_nara, *23–25*, 26, *27*
theme_vlaanderen2015, *23–26*, 26
traffic_palette, *4–6*, *23*, 27, *28*, *29*
transformation object, *9*, *12*, *15*, *18*, *21*

vl_black (vl_yellow), 28
vl_darkblue (vl_yellow), 28
vl_darkbrown (vl_yellow), 28
vl_darkgreen (vl_yellow), 28
vl_darkred (vl_yellow), 28
vl_darkyellow (vl_yellow), 28
vl_grey1 (vl_yellow), 28
vl_grey2 (vl_yellow), 28
vl_grey3 (vl_yellow), 28
vl_grey4 (vl_yellow), 28
vl_grey5 (vl_yellow), 28
vl_grey6 (vl_yellow), 28
vl_grey7 (vl_yellow), 28
vl_lightblue (vl_yellow), 28
vl_lightbrown (vl_yellow), 28
vl_lightgreen (vl_yellow), 28
vl_lightred (vl_yellow), 28
vl_yellow, *4–6*, *23*, *27*, *28*, 28
vlaanderen.2015.colours (inbo.2015.colours), 4
vlaanderen_palette, *4–6*, *23*, *27*, 27, *29*