

Package ‘ggthemes’

January 27, 2015

Version 2.1.0

Title Extra themes, scales and geoms for ggplot

Depends R (>= 3.0.0), ggplot2 (>= 0.9.2)

Imports colorspace, grid, proto, scales

Suggests knitr, plyr, reshape2, extrafont, pander

VignetteBuilder knitr

Description Some extra themes, geoms, and scales for ggplot2.

License GPL-2

URL <http://github.com/jrnold/ggthemes>

BugReports <http://github.com/jrnold/ggthemes>

Collate 'banking.R' 'calc.R' 'colorblind.R' 'economist.R' 'excel.R'
'few.R' 'ggthemes-package.R' 'ggthemes-data.R'
'fivethirtyeight.R' 'gdocs.R' 'geom-rangeframe.R'
'geom-tufteboxplot.R' 'hc.R' 'igray.R' 'pander.R'
'scale-tufte.R' 'shapes.R' 'solarized.R' 'stat-fivenumber.R'
'stata.R' 'tableau.R' 'theme-foundation.R' 'theme-solid.R'
'tufte.R' 'utils.R' 'wsj.R'

Author Jeffrey B. Arnold [aut, cre],
Gergely Daroczi [ctb],
Bo Werth [ctb],
Brian Weitzner [ctb],
Joshua Kunst [ctb]

Maintainer Jeffrey B. Arnold <jeffrey.arnold@gmail.com>

NeedsCompilation no

Repository CRAN

Date/Publication 2014-12-18 06:54:10

R topics documented:

bank_slopes	3
calc_pal	6
calc_shape_pal	6
circlefill_shape_pal	7
cleveland_shape_pal	7
colorblind_pal	8
economist_pal	9
excel_pal	10
extended_range_breaks	10
few_pal	12
fivethirtyeight_pal	12
gdocs_pal	13
geom_rangeframe	13
geom_tufteboxplot	15
ggthemes	16
ggthemes_data	17
hc_pal	20
palette_pander	21
scale_color_pander	21
scale_colour_economist	22
scale_colour_few	23
scale_colour_fivethirtyeight	23
scale_colour_gradient2_tableau	24
scale_colour_gradient_tableau	25
scale_colour_hc	26
scale_colour_stata	26
scale_colour_tableau	27
scale_colour_wsj	28
scale_fill_calc	29
scale_fill_excel	29
scale_fill_gdocs	30
scale_fill_solarized	31
scale_linetype_stata	31
scale_shape_calc	32
scale_shape_circlefill	32
scale_shape_cleveland	33
scale_shape_stata	34
scale_shape_tableau	34
scale_shape_tremmel	35
scale_x_tufte	36
show_linetypes	37
show_shapes	37
smart_digits	38
solarized_pal	39
stata_linetype_pal	39
stata_pal	40

stata_shape_pal	40
stat_fivenumber	41
tableau_color_pal	42
tableau_div_gradient_pal	43
tableau_seq_gradient_pal	44
tableau_shape_pal	44
theme_calc	45
theme_economist	46
theme_excel	47
theme_few	48
theme_fivethirtyeight	49
theme_foundation	50
theme_gdocs	50
theme_hc	51
theme_igray	52
theme_pander	52
theme_solarized	54
theme_solid	55
theme_stata	55
theme_tufte	56
theme_wsj	57
tremmel_shape_pal	58
wsj_pal	59

Index 61

bank_slopes	<i>Bank Slopes to 45 degrees</i>
-------------	----------------------------------

Description

Calculate the optimal aspect ratio of a line graph by banking the slopes to 45 degrees as suggested by W.S. Cleveland. This maximizes the ability to visually differentiate differences in slope. This function will calculate the optimal aspect ratio for a line plot using any of the methods described in Herr and Argwala (2006). In their review of the methods they suggest using median absolute slope banking ("ms"), which produces aspect ratios which are generally the median of the various methods provided here.

Usage

```
bank_slopes(x, y, cull = FALSE, method = "ms", weight = TRUE, ...)
```

Arguments

x	x values
y	y values
cull	logical. Remove all slopes of 0 or Inf.

method	One of "ms" (Median Absolute Slope), "as" (Average Absolute Slope), "ao" (Average Orientation), "lor" (Local Orientation Resolution), "gor" (Global Orientation Resolution).
weight	logical. Weight line segments by their length. Only used when method="ao".
...	Passed to <code>nlm</code> in methods "ao", "lor" and "gor".

Value

numeric The aspect ratio (x, y).

Methods

As written, all of these methods calculate the aspect ratio (x/y), but `bank_slopes` will return (y/x) to be compatible with `link[ggplot2]{coord_fixed}`.

Median Absolute Slopes Banking

Let the aspect ratio be $\alpha = \frac{w}{h}$ then the median absolute slop banking is the α such that,

$$\text{median} \left| \frac{s_i}{\alpha} \right| = 1$$

Let $R_z = z_{max} - z_{min}$ for $z = x, y$, and $M = \text{median} \|s_i\|$. Then,

$$\alpha = M \frac{R_x}{R_y}$$

Average-Absolute-Orientation Banking

This method finds the aspect ratio by setting the average orientation to 45 degrees. For an aspect ratio α , let the orientation of a line segment be $\theta_i(\alpha) = \arctan(s_i/\alpha)$.

$$\frac{\sum_i \theta_i(\alpha) l_i}{\sum_i l_i} = \frac{\pi}{4} \text{rad}$$

where $l_i = 1$ if unweighted, and $l_i = \sqrt{x_i^2 + y_i^2}$ (length of the line segment), if weighted. The value of α is found with `nlm`.

Average Absolute Slope Banking

Let the aspect ratio be $\alpha = \frac{w}{h}$. then the mean absolute slope banking is the α such that,

$$\text{mean} \left| \frac{s_i}{\alpha} \right| = 1$$

Let $R_z = z_{max} - z_{min}$ for $z = x, y$, and $M = \text{mean} \|s_i\|$. Then,

$$\alpha = MR_x/R_y$$

Banking by Optimizing Orientation Resolution

The angle between line segments i and j is $r_{i,j} = \|\theta_i(\alpha) - \theta_j(\alpha)\|$, where $\theta_i(\alpha) = \arctan(s_i/\alpha)$ and s_i is the slope of line segment i . This function finds the α that maximizes the sum of the angles between all pairs of line segments.

$$\max_{\alpha} \sum_i \sum_{j:j<1} r_{i,j}$$

The local optimization only includes line-segments that are next to each other. Suppose there are n line segments, then the local orientation orientation has the following objective function.

$$\max_{\alpha} \sum_{i=2}^n r_{i,i-1}$$

References

- Cleveland, W. S., M. E. McGill, and R. McGill. The Shape Parameter of a Two-Variable Graph. *Journal of the American Statistical Association*, 83:289-300, 1988
- Heer, Jeffrey and Maneesh Agrawala, 2006. "Multi-Scale Banking to 45" *IEEE Transactions On Visualization And Computer Graphics*.
- Cleveland, W. S. 1993. "A Model for Studying Display Methods of Statistical Graphs." *Journal of Computational and Statistical Graphics*.
- Cleveland, W. S. 1994. *The Elements of Graphing Data*, Revised Edition.

See Also

[banking](#)

Examples

```
# Use the classic sunspot data from Cleveland's orig paper
x <- seq_along(sunspot.year)
y <- as.numeric(sunspot.year)
# Without banking
m <- qplot(x, y, geom="line")
m

## Using the default method, Median Absolute Slope
ratio <- bank_slopes(x, y)
m + coord_fixed(ratio = ratio)

## Alternative methods to calculate the banking
bank_slopes(x, y, method="ms")
## Using culling
bank_slopes(x, y, method="ms", cull=TRUE)
## Average Absolute Slope
bank_slopes(x, y, method="as")
bank_slopes(x, y, method="as", cull=TRUE)
## Average Orientation
bank_slopes(x, y, method="ao")
bank_slopes(x, y, method="ao", cull=TRUE)
## Average Orientation (Weighted)
bank_slopes(x, y, method="ao", weight=TRUE)
```

```
bank_slopes(x, y, method="ao", cull=TRUE, weight=TRUE)
## Global Orientation Resolution
bank_slopes(x, y, method="gor")
bank_slopes(x, y, method="gor", cull=TRUE)
## Local Orientation Resolution
bank_slopes(x, y, method="lor")
bank_slopes(x, y, method="lor", cull=TRUE)
```

calc_pal

Calc color palette (discrete)

Description

Color palettes from LibreOffice Calc.

Usage

```
calc_pal()
```

See Also

Other colour calc: [scale_color_calc](#), [scale_colour_calc](#), [scale_fill_calc](#)

Examples

```
library(scales)
show_col(calc_pal()(12))
```

calc_shape_pal

Calc shape palette (discrete)

Description

Shape palette based on the shapes used in LibreOffice Calc.

Usage

```
calc_shape_pal()
```

See Also

Other shapes calc: [scale_shape_calc](#)

Examples

```
show_shapes(calc_shape_pal()(15))
```

circlefill_shape_pal *Filled Circle Shape palette (discrete)*

Description

Shape palette with circles varying by amount of fill. This uses the set of 3 circle fill values in Lewandowsky and Spence (1989): solid, hollow, half-filled, with two additional fill amounts: three-quarters, and one-quarter.

Usage

```
circlefill_shape_pal()
```

References

Lewandowsky, Stephan and Ian Spence (1989) "Discriminating Strata in Scatterplots", Journal of the American Statistical Association, <http://www.jstor.org/stable/2289649>

See Also

Other shapes: [cleveland_shape_pal](#); [scale_shape_circlefill](#); [scale_shape_cleveland](#); [scale_shape_tremmel](#); [tremmel_shape_pal](#)

Examples

```
(ggplot(mtcars, aes(x=mpg, y=hp, shape=factor(cyl)))  
+ geom_point() + scale_shape_tremmel())
```

cleveland_shape_pal *Shape palette from Cleveland "Elements of Graphing Data" (discrete).*

Description

Shape palettes for overlapping and non-overlapping points.

Usage

```
cleveland_shape_pal(overlap = TRUE)
```

Arguments

overlap logical Use the scale for overlapping points?

Note

In the *Elements of Graphing Data*, W.S. Cleveland suggests two shape palettes for scatter plots: one for overlapping data and another for non-overlapping data. The symbols for overlapping data relies on pattern discrimination, while the symbols for non-overlapping data vary the amount of fill. This palette attempts to create these palettes. However, I found that these were hard to replicate. Using the R shapes and unicode fonts: the symbols can vary in size, they are dependent of the fonts used, and there does not exist a unicode symbol for a circle with a vertical line. If someone can improve this palette, please let me know.

Following Tremmel (1995), I replace the circle with a vertical line with an encircled plus sign.

References

Cleveland WS. *The Elements of Graphing Data*. Revised Edition. Hobart Press, Summit, NJ, 1994, pp. 154-164, 234-239.

Tremmel, Lothar, (1995) "The Visual Separability of Plotting Symbols in Scatterplots", *Journal of Computational and Graphical Statistics*, <http://www.jstor.org/stable/1390760>

See Also

Other shapes: [circlefill_shape_pal](#); [scale_shape_circlefill](#); [scale_shape_cleveland](#); [scale_shape_tremmel](#); [tremmel_shape_pal](#)

Examples

```
# overlapping symbol palette
dsamp <- diamonds[sample(nrow(diamonds), 100), ]
(qplot(carat, price, data=dsamp, shape=cut)
+ theme_bw() + scale_shape_cleveland())
# non-overlapping symbol palette
(qplot(carat, price, data=dsamp, shape=cut)
+ theme_bw() + scale_shape_cleveland(overlap=FALSE))
```

colorblind_pal

Colorblind Color Palette (Discrete) and Scales

Description

An 8-color colorblind safe qualitative discrete palette from <http://jfly.iam.u-tokyo.ac.jp/color> and the .

Usage

```
colorblind_pal()

scale_colour_colorblind(...)

scale_color_colorblind(...)
```

```
scale_fill_colorblind(...)
```

Arguments

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

The **dichromat** package, [dichromat_pal](#), and [scale_color_tableau](#) for other colorblind palettes.

Examples

```
library(scales)
show_col(colorblind_pal()(8))
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
p <- qplot(carat, price, data=dsamp, colour=clarity) + theme_igray()
p + scale_colour_colorblind()
```

economist_pal

Economist color palette (discrete)

Description

The hues in the palette are blues, grays, and greens. Red is not included in these palettes and should be used to indicate important data.

Usage

```
economist_pal(stata = FALSE, fill = TRUE)
```

Arguments

stata Use the palette in the Stata economist scheme.
fill Use the fill palette.

See Also

Other colour economist: [scale_color_economist](#), [scale_colour_economist](#), [scale_fill_economist](#)

Examples

```
library(scales)
show_col(economist_pal()(6))
## fill palette
show_col(economist_pal(fill=TRUE)(6))
## RGB values from Stata's economist scheme
show_col(economist_pal(stata=TRUE)(16))
```

excel_pal	<i>Excel color palette (discrete)</i>
-----------	---------------------------------------

Description

Color palettes from Excel, both current and the pre-2007 ugly palettes.

Usage

```
excel_pal(palette = "line")
```

Arguments

palette One of "old", "fill", or "new".

Details

The color palettes are

line Excel 2003 default color palette.

fill Excel 2003 bar chart color palette.

new Color palette from newer Excel versions.

See Also

Other colour excel: [scale_color_excel](#), [scale_colour_excel](#), [scale_fill_excel](#)

Examples

```
library(scales)
show_col(excel_pal()(8))
show_col(excel_pal("fill")(8))
show_col(excel_pal("new")(10))
```

extended_range_breaks	<i>Pretty axis breaks inclusive of extreme values</i>
-----------------------	---

Description

This function returns pretty axis breaks that always include the extreme values of the data. This works by calling the extended Wilkinson algorithm (Talbot et. al, 2010), constrained to solutions interior to the data range. Then, the minimum and maximum labels are moved to the minimum and maximum of the data range.

Usage

```
extended_range_breaks(dmin, dmax, n = 5, Q = c(1, 5, 2, 2.5, 4, 3),  
  w = c(0.25, 0.2, 0.5, 0.05))  
  
scales_extended_range_breaks(expand = c(0, 0), ...)
```

Arguments

dmin	minimum of the data range
dmax	maximum of the data range
n	desired number of breaks
Q	set of nice numbers
w	weights applied to the four optimization components (simplicity, coverage, density, and legibility)
expand	see scale_x_continuous .
...	other arguments passed to <code>extended_range_breaks</code>

Details

`extended_range_breaks` implements the algorithm and returns the break values. `scales_extended_range_breaks` uses the conventions of the **scales** package, and returns a function.

Value

For `extended_range_breaks`, the vector of axis label locations. For `scales_extended_range_breaks`, a function which takes a single argument, a vector of data, and returns the vector of axis label locations.

Author(s)

Justin Talbot <jtalbot@stanford.edu>, Jeffrey B. Arnold, Baptiste Auguie

References

Talbot, J., Lin, S., Hanrahan, P. (2010) An Extension of Wilkinson's Algorithm for Positioning Tick Labels on Axes, InfoVis 2010.

See Also

[scale_y_tufte](#), [scale_x_tufte](#)

`few_pal`*Color Palletes from Few "Practical Rules for Using Color in Charts"*

Description

Qualitative color palettes from Stephen Few, "[Practical Rules for Using Color in Charts](#)".

Usage

```
few_pal(palette = "medium")
```

Arguments

`palette` One of "medium", "dark", or "light"

Details

He suggests the following

- For bars, use medium.
- For lines and points use dark if small or thin, and medium otherwise.

See Also

Other colour few: [scale_color_few](#), [scale_colour_few](#), [scale_fill_few](#)

Examples

```
library(scales)
show_col(few_pal()(7))
show_col(few_pal("dark")(7))
show_col(few_pal("light")(7))
```

`fivethirtyeight_pal`*fivethirtyeight.com color palette*

Description

The standard fivethirtyeight.com palette for line plots is blue, red, green.

Usage

```
fivethirtyeight_pal()
```

See Also

Other colour fivethirtyeight: [scale_color_fivethirtyeight](#), [scale_colour_fivethirtyeight](#), [scale_fill_fivethirtyeight](#)

Examples

```
library("scales")
show_col(fivethirtyeight_pal()(3))
```

gdocs_pal

Google Docs color palette (discrete)

Description

Color palettes from Google Docs.

Usage

```
gdocs_pal()
```

See Also

Other colour gdocs: [scale_color_gdocs](#), [scale_colour_gdocs](#), [scale_fill_gdocs](#)

Examples

```
library(scales)
show_col(gdocs_pal()(20))
```

geom_rangeframe

Range Frames

Description

Axis lines which extend to the maximum and minimum of the plotted data.

Usage

```
geom_rangeframe(mapping = NULL, data = NULL, stat = "identity",
  position = "identity", sides = "bl", fun_min = min, fun_max = max,
  ...)
```

Arguments

mapping	The aesthetic mapping, usually constructed with aes or aes_string . Only needs to be set at the layer level if you are overriding the plot defaults.
data	A layer specific dataset - only needed if you want to override the plot defaults.
stat	The statistical transformation to use on the data for this layer.
position	The position adjustment to use for overlapping points on this layer
sides	A string that controls which sides of the plot the frames appear on. It can be set to a string containing any of "trbl", for top, right, bottom, and left.
fun_min	Function used to calculate the maximum of the range frame line.
fun_max	Function used to calculate the minimum of the range frame line.
...	other arguments passed on to layer . This can include aesthetics whose values you want to set, not map. See layer for more details.

Aesthetics

`geom_tufteboxplot` understands the following aesthetics (required aesthetics are in bold):

- alpha
- colour
- linetype
- size

References

Tufte, Edward R. (2001) *The Visual Display of Quantitative Information*, Chapter 6.

See Also

Other geom tufte: [geom_tufteboxplot](#)

Examples

```
(ggplot(mtcars, aes(wt, mpg))
+ geom_point() + geom_rangeframe()
+ theme_tufte())
```

geom_tufteboxplot *Tufte's Box Blot*

Description

Edward Tufte's revision of the box plot erases the box and replaces it with a single point and the whiskers.

Usage

```
geom_tufteboxplot(mapping = NULL, data = NULL, stat = "boxplot",
  position = "dodge", outlier.colour = "black", outlier.shape = 16,
  outlier.size = 2, fatten = 4, median.type = "point", boxwidth = 0.25,
  ...)
```

Arguments

mapping	The aesthetic mapping, usually constructed with aes or aes_string . Only needs to be set at the layer level if you are overriding the plot defaults.
data	A layer specific dataset - only needed if you want to override the plot defaults.
stat	The statistical transformation to use on the data for this layer.
position	The position adjustment to use for overlapping points on this layer
outlier.colour	colour for outlying points
outlier.shape	shape of outlying points
outlier.size	size of outlying points
fatten	a multiplicative factor to fatten the middle point (or line) by
median.type	One of "box", "line", or "point". If median.type="point", then use whitespace to represent the central quartiles and a point at the median. If median.type="box", then use a box to represent the standard error of the median. This is similar to what the notch option does in a standard boxplot. the same thing as the notch does in a standard boxplot. If median.type="line", the use offset lines to represent the central quartile and whitespace at the median
boxwidth	a number between 0 and 1 which represents the relative width of the box to the middle line.
...	other arguments passed on to layer . This can include aesthetics whose values you want to set, not map. See layer for more details.

Aesthetics

geom_tufteboxplot understands the following aesthetics (required aesthetics are in bold):

- **lower**
- **middle**
- **upper**

- **x**
- **ymin**
- **ymax**
- alpha
- colour
- fill
- linetype
- shape
- size

References

Tufte, Edward R. (2001) The Visual Display of Quantitative Information, Chapter 6.

McGill, R., Tukey, J. W. and Larsen, W. A. (1978) Variations of box plots. The American Statistician 32, 12-16.

See Also

[geom_boxplot](#)

Other geom tufte: [geom_rangeframe](#)

Examples

```
p <- ggplot(mtcars, aes(factor(cyl), mpg))
## with only a point
p + geom_tufteboxplot()
## with a middle box
p + geom_tufteboxplot(median.type="box", fatten=1)
## using lines
p + geom_tufteboxplot(median.type="line")
```

ggthemes

ggthemes

Description

This package contains extra themes, scales, and geoms, and functions for and related to **ggplot2**.

Details

In addition to the help pages, see the README page on [github](#) for examples.

ggthemes_data

*Palette data for ggthemes package***Description**

List with the data used by the palettes in the ggthemes package.

Usage

```
ggthemes_data
```

Format

```
List of 13
$ economist      :List of 3
..$ bg           : Named chr [1:5] "#d5e4eb" "#c3d6df" "#ed111a" "#ebebeb" ...
.. ..- attr(*, "names")= chr [1:5] "ebg" "edkbg" "red" "ltgray" ...
..$ fg           : Named chr [1:12] "#6794a7" "#014d64" "#76c0c1" "#01a2d9" ...
.. ..- attr(*, "names")= chr [1:12] "blue_gray" "blue_dark" "green_light" "blue_mid" ...
..$ stata:List of 2
.. ..$ bg: Named chr [1:2] "#C6D3DF" "#B2BFCB"
.. .. ..- attr(*, "names")= chr [1:2] "ebg" "edkbg"
.. ..$ fg: Named chr [1:15] "#3E647D" "#7B92A8" "#82C0E9" "#2D6D66" ...
.. .. ..- attr(*, "names")= chr [1:15] "edkblue" "emidblue" "eltblue" "emerald" ...
$ excel          :List of 3
..$ line: chr [1:7] "#FF00FF" "#FFFFFF00" "#00FFFF" "#800080" ...
..$ fill: chr [1:7] "#993366" "#FFFCC" "#CCFFF" "#660066" ...
..$ new : chr [1:10] "#365e96" "#983334" "#77973d" "#5d437c" ...
$ solarized      :List of 2
..$ base        : Named chr [1:8] "#002b36" "#073642" "#586e75" "#657b83" ...
.. ..- attr(*, "names")= chr [1:8] "base03" "base02" "base01" "base00" ...
..$ accents: Named chr [1:8] "#b58900" "#cb4b16" "#dc322f" "#d33682" ...
.. ..- attr(*, "names")= chr [1:8] "yellow" "orange" "red" "magenta" ...
$ stata          :List of 3
..$ colors      : Named chr [1:73] "#97b6b0" "#55752f" "#ffe474" "#ffd200" ...
.. ..- attr(*, "names")= chr [1:73] "eltgreen" "forest_green" "sandb" "gold" ...
..$ shapes      : Named num [1:41] 16 16 16 16 18 18 18 18 17 17 ...
.. ..- attr(*, "names")= chr [1:41] "0" "o" "circle" "smcircle" ...
..$ linetypes: chr [1:15] "solid" "84" "23" "F414" ...
$ few            :List of 3
..$ medium: Named chr [1:8] "#737373" "#F15A60" "#7AC36A" "#5A9BD4" ...
.. ..- attr(*, "names")= chr [1:8] "gray" "red" "green" "blue" ...
..$ dark : Named chr [1:8] "#010202" "#EE2E2F" "#008C48" "#185AA9" ...
.. ..- attr(*, "names")= chr [1:8] "black" "red" "green" "blue" ...
..$ light : Named chr [1:8] "#CCCCCC" "#F2AFAD" "#D9E4AA" "#B8D2EC" ...
.. ..- attr(*, "names")= chr [1:8] "gray" "red" "green" "blue" ...
$ tableau       :List of 4
```

```

..$ colors      :List of 9
.. ..$ tableau20      : Named chr [1:20] "#1F77B4" "#AEC7E8" "#FF7F0E" "#FFBB78" ...
.. .. ..- attr(*, "names")= chr [1:20] "blue" "blue_light" "orange" "orange_light" ...
.. ..$ tableau10medium: Named chr [1:10] "#729ECE" "#FF9E4A" "#67BF5C" "#ED665D" ...
.. .. ..- attr(*, "names")= chr [1:10] "blue" "orange" "green" "red" ...
.. ..$ gray5          : chr [1:5] "#60636A" "#A5ACAF" "#414451" "#8F8782" ...
.. ..$ colorblind10   : chr [1:10] "#006BA4" "#FF800E" "#ABABAB" "#595959" ...
.. ..$ trafficlight   : chr [1:9] "#B10318" "#DBA13A" "#309343" "#D82526" ...
.. ..$ purplegray12   : chr [1:12] "#7B66D2" "#A699E8" "#DC5FBD" "#FFC0DA" ...
.. ..$ bluered12      : chr [1:12] "#2C69B0" "#B5C8E2" "#F02720" "#FFB6B0" ...
.. ..$ greenorange12 : chr [1:12] "#32A251" "#ACD98D" "#FF7F0F" "#FFB977" ...
.. ..$ cyclic         : chr [1:20] "#1F83B4" "#1696AC" "#18A188" "#29A03C" ...
..$ sequential:List of 16
.. ..$ Red           : Named chr [1:2] "#BCCFB4" "#9C0824"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Green         : Named chr [1:2] "#BCCFB4" "#09622A"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Blue          : Named chr [1:2] "#B4D4DA" "#26456E"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Orange        : Named chr [1:2] "#F0C294" "#7B3014"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Gray          : Named chr [1:2] "#C3C3C3" "#1E1E1E"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Red Light     : Named chr [1:2] "#E5E5E5" "#FFB2B6"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Green Light   : Named chr [1:2] "#E5E5E5" "#B7E6A7"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Blue Light    : Named chr [1:2] "#E5E5E5" "#C4D8F3"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Orange Light  : Named chr [1:2] "#E5E5E5" "#FFCC9E"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Area Red      : Named chr [1:2] "#F5CAC7" "#BD1100"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Area Green    : Named chr [1:2] "#DBE8B4" "#3C8200"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Area Brown    : Named chr [1:2] "#F3E0C2" "#BB5137"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Blue-Green Sequential: Named chr [1:2] "#FEFFD9" "#41B7C4"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Brown Sequential : Named chr [1:2] "#F7E4C6" "#BB5137"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Purple Sequential : Named chr [1:2] "#EFEDF5" "#807DBA"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
.. ..$ Grey Sequential : Named chr [1:2] "#F0F0F0" "#737373"
.. .. ..- attr(*, "names")= chr [1:2] "low" "high"
..$ diverging :List of 16
.. ..$ Red-Blue      : Named chr [1:3] "#9C0824" "#CACACA" "#26456E"
.. .. ..- attr(*, "names")= chr [1:3] "low" "mid" "high"

```

```

.. ..$ Red-Green          : Named chr [1:3] "#9C0824" "#CACACA" "#09622A"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Red-White-Green   : Named chr [1:3] "#9C0824" "#FFFFFF" "#09622A"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Red-Black         : Named chr [1:3] "#9C0824" "#CACACA" "#1E1E1E"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Red-White-Black   : Named chr [1:3] "#9C0824" "#FFFFFF" "#1E1E1E"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Green-Blue        : Named chr [1:3] "#09622A" "#CACACA" "#26456E"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Orange-Blue       : Named chr [1:3] "#7B3014" "#CACACA" "#26456E"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Orange-White-Blue : Named chr [1:3] "#7B3014" "#FFFFFF" "#26456E"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Red-Green Light   : Named chr [1:3] "#FFB2B6" "#E5E5E5" "#B7E6A7"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Red-White-Green Light : Named chr [1:3] "#FFB2B6" "#FFFFFF" "#B7E6A7"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Red-White-Black Light : Named chr [1:3] "#FFB2B6" "#FFFFFF" "#C6C6C6"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Orange-Blue Light : Named chr [1:3] "#FFCC9E" "#E5E5E5" "#C4D8F3"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Orange-White-Blue Light: Named chr [1:3] "#FFCC9E" "#FFFFFF" "#C4D8F3"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Orange-Blue       : Named chr [1:3] "#E0AD30" "#E4E4E2" "#7492AA"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Light Red-Green   : Named chr [1:3] "#EDA389" "#CDE1D3" "#5C8B70"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
.. ..$ Temperature      : Named chr [1:3] "#529985" "#DBCF47" "#C26B51"
.. ..  .- attr(*, "names")= chr [1:3] "low" "mid" "high"
..$ shapes              :List of 7
.. ..$ proportions: int [1:5] -9675 -605 -9681 -9685 -9679
.. ..$ default         : int [1:10] 1 0 3 4 -8727 5 -9651 -9661 -9655 -9665
.. ..$ filled          : int [1:10] 16 15 -10133 -10006 -9733 18 -9650 -9660 -9654 -9664
.. ..$ gender           : int [1:2] -9794 -9792
.. ..$ kpi              : int [1:6] -10003 21 4 16 17 18
.. ..$ thin_arrows: int [1:8] -8595 -8600 -8594 -8599 -8593 -8598 -8592 -8601
.. ..$ weather          : int [1:4] -9728 -9729 -9730 -9731
$ manyeyes              : chr [1:19] "#9c9ede" "#7375b5" "#4a5584" "#cedb9c" ...
$ wsj                   :List of 2
..$ bg                  : Named chr [1:4] "#efefef" "#e9f3ea" "#d4dee7" "#f8f2e4"
..  .- attr(*, "names")= chr [1:4] "gray" "green" "blue" "brown"
..$ palettes:List of 5
.. ..$ rgby             : Named chr [1:4] "#d3ba68" "#d5695d" "#5d8ca8" "#65a479"
.. ..  .- attr(*, "names")= chr [1:4] "yellow" "red" "blue" "green"
.. ..$ red_green        : Named chr [1:2] "#088158" "#ba2f2a"
.. ..  .- attr(*, "names")= chr [1:2] "green" "red"
.. ..$ black_green: Named chr [1:4] "#000000" "#595959" "#59a77f" "#008856"

```

```

.. .. .- attr(*, "names")= chr [1:4] "black" "gray" "ltgreen" "green"
.. ..$ dem_rep      : Named chr [1:3] "#006a8e" "#b1283a" "#a8a6a7"
.. .. .- attr(*, "names")= chr [1:3] "blue" "red" "gray"
.. ..$ colors6      : Named chr [1:6] "#c72e29" "#016392" "#be9c2e" "#098154" ...
.. .. .- attr(*, "names")= chr [1:6] "red" "blue" "gold" "green" ...
$ colorblind       : Named chr [1:8] "#000000" "#E69F00" "#56B4E9" "#009E73" ...
.- attr(*, "names")= chr [1:8] "black" "orange" "sky_blue" "bluish_green" ...
$ gdocs            : chr [1:20] "#3366CC" "#DC3912" "#FF9900" "#109618" ...
$ calc             :List of 2
..$ colors: Named chr [1:12] "#004586" "#FF420E" "#FFD320" "#579D1C" ...
.. ..- attr(*, "names")= chr [1:12] "Chart 1" "Chart 2" "Chart 3" "Chart 4" ...
..$ shapes: num [1:15] 15 18 -9660 -9650 -9654 ...
$ fivethirtyeight: Named chr [1:6] "#3C3C3C" "#D2D2D2" "#F0F0F0" "#FF2700" ...
.- attr(*, "names")= chr [1:6] "dkgray" "medgray" "ltgray" "red" ...
$ hc               :List of 2
..$ palettes:List of 2
.. ..$ default      : chr [1:10] "#7cb5ec" "#434348" "#90ed7d" "#f7a35c" ...
.. ..$ darkunica: chr [1:11] "#2b908f" "#90ee7e" "#f45b5b" "#7798BF" ...
..$ bg              : Named chr [1:2] "#FFFFFF" "#2a2a2b"
.. ..- attr(*, "names")= chr [1:2] "default" "darkunica"

```

hc_pal

Highcharts JS color palette (discrete)

Description

The Highcharts JS uses many different color palettes in its plots. This collects a few of them.

Usage

```
hc_pal(palette = "default")
```

Arguments

palette character The color palette to use. This must be a name in [ggthemes_data\\$hc\\$palettes](#).

Palettes

The following palettes are defined,

default #7cb5ec, #434348, #90ed7d, #f7a35c, #8085e9, #f15c80", #e4d354, #8085e8, #8d4653, #91e8e1 theme. Examples: <http://www.highcharts.com/demo>.

darkunica #2b908f, #90ee7e, #f45b5b, #7798BF, #aaeccc, #ff0066, #eeaaee, #55BF3B, #DF5353, #7798BF, #aaeccc". Examples: <http://www.highcharts.com/demo/line-basic/dark-unica>.

palette_pander *Color palette from the pander package*

Description

The **pander** ships with a default colorblind and printer-friendly color palette borrowed from <http://jfly.iam.u-tokyo.ac.jp/color/>.

Usage

```
palette_pander(n, random_order = FALSE)
```

Arguments

n	number of colors
random_order	if the palette should be reordered randomly before rendering each plot to get colorful images

See Also

Other colour pander: [scale_color_pander](#), [scale_colour_pander](#), [scale_fill_pander](#)

Examples

```
## Not run:  
palette_pander(TRUE)  
  
## End(Not run)
```

scale_color_pander *Color scale from the pander package*

Description

The **pander** ships with a default colorblind and printer-friendly color palette borrowed from <http://jfly.iam.u-tokyo.ac.jp/color/>.

Usage

```
scale_color_pander(...)  
  
scale_colour_pander(...)  
  
scale_fill_pander(...)
```

Arguments

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

[theme_pander](#)

Other colour pander: [palette_pander](#)

scale_colour_economist

Economist color scales

Description

Color scales using the colors in the Economist graphics.

Usage

```
scale_colour_economist(stata = FALSE, ...)
```

```
scale_color_economist(stata = FALSE, ...)
```

```
scale_fill_economist(stata = FALSE, ...)
```

Arguments

stata Use the palette in the Stata economist scheme.

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

[theme_economist](#) for examples.

Other colour economist: [economist_pal](#)

scale_colour_few *Color scales from Few "Practical Rules for Using Color in Charts"*

Description

See [few_pal](#).

Usage

```
scale_colour_few(palette = "medium", ...)
```

```
scale_color_few(palette = "medium", ...)
```

```
scale_fill_few(palette = "light", ...)
```

Arguments

palette One of "medium", "dark", or "light"

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

Other colour few: [few_pal](#)

scale_colour_fivethirtyeight
fivethirtyeight.com color scales

Description

Color scales using the colors in the fivethirtyeight graphics.

Usage

```
scale_colour_fivethirtyeight(...)
```

```
scale_color_fivethirtyeight(...)
```

```
scale_fill_fivethirtyeight(...)
```

Arguments

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

[theme_fivethirtyeight](#) for examples.

Other colour fivethirtyeight: [fivethirtyeight_pal](#)

scale_colour_gradient2_tableau

Tableau diverging colour scales (continuous)

Description

Tableau diverging colour scales (continuous)

Usage

```
scale_colour_gradient2_tableau(palette = "Red-Blue", ..., space = "rgb",
  na.value = "grey50", guide = "colourbar")
```

```
scale_fill_gradient2_tableau(palette = "Red-Blue", ..., space = "rgb",
  na.value = "grey50", guide = "colourbar")
```

```
scale_color_gradient2_tableau(palette = "Red-Blue", ..., space = "rgb",
  na.value = "grey50", guide = "colourbar")
```

Arguments

palette	Palette name: One of .
...	Other arguments passed on to discrete_scale to control name, limits, breaks, labels and so forth.
space	Colour space in which to calculate gradient.
na.value	Colour to use for missing values
guide	Type of legend. Use "colourbar" for continuous colour bar, or "legend" for discrete colour legend.

See Also

Other colour tableau: [scale_color_continuous_tableau](#), [scale_color_gradient_tableau](#), [scale_colour_gradient_tableau](#), [scale_fill_continuous_tableau](#), [scale_fill_gradient_tableau](#); [scale_color_tableau](#), [scale_colour_tableau](#), [scale_fill_tableau](#); [tableau_color_pal](#); [tableau_div_gradient_pal](#); [tableau_seq_gradient_pal](#)

Examples

```
dsub <- subset(diamonds, x > 5 & x < 6 & y > 5 & y < 6)
dsub$diff <- with(dsub, sqrt(abs(x-y))* sign(x-y))
d <- qplot(x, y, data=dsub, colour=diff)
d + scale_colour_gradient2_tableau()
d + scale_colour_gradient2_tableau("Orange-Blue")
d + scale_colour_gradient2_tableau("Temperature")
```

scale_colour_gradient_tableau

Tableau sequential colour scale (continuous)

Description

Tableau sequential colour scale (continuous)

Usage

```
scale_colour_gradient_tableau(palette = "Red", ..., space = "Lab",
  na.value = "grey50", guide = "colourbar")
```

```
scale_fill_gradient_tableau(palette = "Red", ..., space = "Lab",
  na.value = "grey50", guide = "colourbar")
```

```
scale_color_gradient_tableau(palette = "Red", ..., space = "Lab",
  na.value = "grey50", guide = "colourbar")
```

```
scale_color_continuous_tableau(palette = "Red", ..., space = "Lab",
  na.value = "grey50", guide = "colourbar")
```

```
scale_fill_continuous_tableau(palette = "Red", ..., space = "Lab",
  na.value = "grey50", guide = "colourbar")
```

Arguments

palette	Palette name: One of "Red", "Green", "Blue", "Orange", "Gray", "Red Light", "Green Light", "Blue Light", "Orange Light", "Area Red", "Area Green", "Area Brown", "Blue-Green Sequential", "Brown Sequential", "Purple Sequential", "Grey Sequential".
...	Other arguments passed on to discrete_scale to control name, limits, breaks, labels and so forth.
space	Colour space in which to calculate gradient.
na.value	Colour to use for missing values
guide	Type of legend. Use "colourbar" for continuous colour bar, or "legend" for discrete colour legend.

See Also

Other colour tableau: [scale_color_gradient2_tableau](#), [scale_colour_gradient2_tableau](#), [scale_fill_gradient2_tableau](#); [scale_color_tableau](#), [scale_colour_tableau](#), [scale_fill_tableau](#); [tableau_color_pal](#); [tableau_div_gradient_pal](#); [tableau_seq_gradient_pal](#)

Examples

```
dsub <- subset(diamonds, x > 5 & x < 6 & y > 5 & y < 6)
d <- qplot(x, y, data=dsub, colour=z)
d + scale_colour_gradient_tableau("Red", limits=c(3, 4))
d + scale_colour_gradient_tableau("Blue", limits=c(3, 4))
d + scale_colour_gradient_tableau("Green", limits=c(3, 4))
```

scale_colour_hc *Highcharts color and fill scales*

Description

Colour and fill scales which use the palettes in [hc_pal](#) and are meant for use with [theme_hc](#).

Usage

```
scale_colour_hc(palette = "default", ...)
scale_color_hc(palette = "default", ...)
scale_fill_hc(palette = "default", ...)
```

Arguments

palette	character	The color palette to use. This must be a name in ggthemes_data\$hc\$palettes .
...		Other arguments passed on to discrete_scale to control name, limits, breaks, labels and so forth.

scale_colour_stata *Stata color scales*

Description

See [stata_pal](#) for details.

Usage

```
scale_colour_stata(scheme = "s2color", ...)

scale_fill_stata(scheme = "s2color", ...)

scale_color_stata(scheme = "s2color", ...)
```

Arguments

`scheme` character. One of "s2color", "s1rcolor", "s1color", or "mono".

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
(d <- qplot(carat, price, data=dsamp, colour=clarity)
  + theme_bw()
  + scale_color_stata())
(d <- qplot(carat, price, data=dsamp, colour=clarity)
  + theme_bw()
  + scale_color_stata("s1color"))
```

scale_colour_tableau *Tableau color scales.*

Description

See [tableau_color_pal](#) for details.

Usage

```
scale_colour_tableau(palette = "tableau10", ...)

scale_fill_tableau(palette = "tableau10", ...)

scale_color_tableau(palette = "tableau10", ...)
```

Arguments

`palette` Palette name. One of "tableau20", "tableau10medium", "gray5", "colorblind10", "trafficlight", "purplegray12", "bluered12", "greenorange12", "cyclic", "tableau10", "tableau10light", "purplegray6", "bluered6", "greenorange6".

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

[tableau_color_pal](#) for references.

Other colour tableau: [scale_color_continuous_tableau](#), [scale_color_gradient_tableau](#), [scale_colour_gradient_tableau](#), [scale_fill_continuous_tableau](#), [scale_fill_gradient_tableau](#); [scale_color_gradient2_tableau](#), [scale_colour_gradient2_tableau](#), [scale_fill_gradient2_tableau](#); [tableau_color_pal](#); [tableau_div_gradient_pal](#); [tableau_seq_gradient_pal](#)

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
p <- qplot(carat, price, data=dsamp, colour=clarity) + theme_igray()
p + scale_colour_tableau()
p + scale_colour_tableau("tableau20")
p + scale_colour_tableau("tableau10medium")
p + scale_colour_tableau("tableau10light")
p + scale_colour_tableau("colorblind10")
p + scale_colour_tableau("trafficlight")
p + scale_colour_tableau("purplegray12")
p + scale_colour_tableau("bluered12")
p + scale_colour_tableau("greenorange12")
p + scale_colour_tableau("cyclic")
```

scale_colour_wsj

Wall Street Journal color and fill scales

Description

Colour and fill scales which use the palettes in [wsj_pal](#) and are meant for use with [theme_wsj](#).

Usage

```
scale_colour_wsj(palette = "colors6", ...)
```

```
scale_color_wsj(palette = "colors6", ...)
```

```
scale_fill_wsj(palette = "colors6", ...)
```

Arguments

palette	character	The color palette to use. This must be a name in <code>ggthemes_data\$wsj\$palettes</code> .
...		Other arguments passed on to discrete_scale to control name, limits, breaks, labels and so forth.

See Also

Other colour wsj: [wsj_pal](#)

scale_fill_calc	<i>LibreOffice Calc color scales</i>
-----------------	--------------------------------------

Description

Color scales from LibreOffice Calc.

Usage

```
scale_fill_calc(...)
```

```
scale_colour_calc(...)
```

```
scale_color_calc(...)
```

Arguments

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

See [theme_calc](#) for examples.

Other colour calc: [calc_pal](#)

scale_fill_excel	<i>Excel color scales</i>
------------------	---------------------------

Description

Color scales from both old and new Excel.

Usage

```
scale_fill_excel(palette = "fill", ...)
```

```
scale_colour_excel(palette = "line", ...)
```

```
scale_color_excel(palette = "line", ...)
```

Arguments

palette One of "old", "fill", or "new".

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

See [theme_excel](#) for examples.

Other colour excel: [excel_pal](#)

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
(qplot(carat, price, data=dsamp, colour=clarity)
+ theme_igray()
+ scale_colour_excel("new"))
```

scale_fill_gdocs

Google Docs color scales

Description

Color scales from Google Docs.

Usage

```
scale_fill_gdocs(...)
```

```
scale_colour_gdocs(...)
```

```
scale_color_gdocs(...)
```

Arguments

... Other arguments passed on to [discrete_scale](#) to control name, limits, breaks, labels and so forth.

See Also

See [theme_gdocs](#) for examples.

Other colour gdocs: [gdocs_pal](#)

scale_fill_solarized *Solarized color scales*

Description

See [solarized_pal](#) for details.

Usage

```
scale_fill_solarized(accent = "blue", ...)
scale_colour_solarized(accent = "blue", ...)
scale_color_solarized(accent = "blue", ...)
```

Arguments

accent	character Starting color. One of "yellow", "orange", "red", "magenta", "violet", "blue", "cyan", "green"
...	Other arguments passed on to discrete_scale to control name, limits, breaks, labels and so forth.

See Also

Other solarized colour: [solarized_pal](#)

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
(d <- qplot(carat, price, data=dsamp, colour=clarity)
  + theme_solarized()
  + scale_colour_solarized() )
```

scale_linetype_stata *Stata linetype palette (discrete)*

Description

See [stata_linetype_pal](#) for details.

Usage

```
scale_linetype_stata(...)
```

Arguments

...	common discrete scale parameters: name, breaks, labels, na.value, limits and guide. See discrete_scale for more details
-----	---

See Also

Other linetype stata: [stata_linetype_pal](#)

Examples

```
library(reshape2) # for melt
library(plyr) # for ddply
library(ggplot2)
ecm <- melt(economics, id = "date")
rescale01 <- function(x) {(x - min(x)) / diff(range(x))}
ecm <- ddply(ecm, "variable", transform, value = rescale01(value))
qplot(date, value, data=ecm, geom="line", linetype=variable) + scale_linetype_stata()
```

scale_shape_calc *Calc shape scale*

Description

See [calc_shape_pal](#) for details.

Usage

```
scale_shape_calc(...)
```

Arguments

... common discrete scale parameters: name, breaks, labels, na.value, limits and guide. See [discrete_scale](#) for more details

See Also

[theme_calc](#) for examples.

Other shapes calc: [calc_shape_pal](#)

scale_shape_circlefill
Filled Circle Shape palette (discrete)

Description

Filled Circle Shape palette (discrete)

Usage

```
scale_shape_circlefill(...)
```

Arguments

... common discrete scale parameters: name, breaks, labels, na.value, limits and guide. See [discrete_scale](#) for more details

See Also

[circlefill_shape_pal](#) for a description of the palette.

Other shapes: [circlefill_shape_pal](#); [cleveland_shape_pal](#); [scale_shape_cleveland](#); [scale_shape_tremmel](#); [tremmel_shape_pal](#)

scale_shape_cleveland *Shape scales from Cleveland "Elements of Graphing Data"*

Description

Shape scales from Cleveland "Elements of Graphing Data"

Usage

```
scale_shape_cleveland(overlap = TRUE, ...)
```

Arguments

overlap logical Use the scale for overlapping points?

... common discrete scale parameters: name, breaks, labels, na.value, limits and guide. See [discrete_scale](#) for more details

References

Cleveland WS. The Elements of Graphing Data. Revised Edition. Hobart Press, Summit, NJ, 1994, pp. 154-164, 234-239.

See Also

[cleveland_shape_pal](#) for a description of the palette.

Other shapes: [circlefill_shape_pal](#); [cleveland_shape_pal](#); [scale_shape_circlefill](#); [scale_shape_tremmel](#); [tremmel_shape_pal](#)

scale_shape_stata *Stata shape scale*

Description

See [stata_shape_pal](#) for details.

Usage

```
scale_shape_stata(...)
```

Arguments

... common discrete scale parameters: name, breaks, labels, na.value, limits and guide. See [discrete_scale](#) for more details

Examples

```
dsmall <- diamonds[sample(nrow(diamonds), 100), ]
(d <- qplot(carat, price, data=dsmall, shape=cut)
+ scale_shape_stata())
```

scale_shape_tableau *Tableau shape scales*

Description

See [tableau_shape_pal](#) for details.

Usage

```
scale_shape_tableau(palette = "default", ...)
```

Arguments

palette Palette name. One of "proportions", "default", "filled", "gender", "kpi", "thin_arrows", "weather".

... common discrete scale parameters: name, breaks, labels, na.value, limits and guide. See [discrete_scale](#) for more details

See Also

Other shape tableau: [tableau_shape_pal](#)

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
p <- qplot(carat, price, data=dsamp, shape=clarity)
p + scale_shape_tableau()
```

scale_shape_tremmel *Shape scales from Tremmel (1995)*

Description

Shape scales from Tremmel (1995)

Usage

```
scale_shape_tremmel(overlap = FALSE, n3alt = TRUE, ...)
```

Arguments

overlap	use an empty circle instead of a solid circle when $n == 2$.
n3alt	If TRUE then use a solid circle, plus sign and empty triangle, else use a solid circle, empty circle, and empty triangle.
...	common discrete scale parameters: name, breaks, labels, na.value, limits and guide. See discrete_scale for more details

See Also

[tremmel_shape_pal](#) for a description of the palette.

Other shapes: [circlefill_shape_pal](#); [cleveland_shape_pal](#); [scale_shape_circlefill](#); [scale_shape_cleveland](#); [tremmel_shape_pal](#)

Examples

```
(ggplot(mtcars, aes(x=mpg, y=hp, shape=factor(cyl)))
+ geom_point() + scale_shape_tremmel())
(ggplot(mtcars, aes(x=mpg, y=hp, shape=factor(cyl)))
+ geom_point() + scale_shape_tremmel(n3alt=FALSE))
(ggplot(mtcars, aes(x=mpg, y=hp, shape=factor(am)))
+ geom_point() + scale_shape_tremmel())
(ggplot(mtcars, aes(x=mpg, y=hp, shape=factor(am)))
+ geom_point() + scale_shape_tremmel(overlap=TRUE))
```

scale_x_tufte	<i>Axis breaks inclusive of extreme values</i>
---------------	--

Description

These scales draw pretty axis breaks that always include the extreme values of the data.

Usage

```
scale_x_tufte(breaks = scales_extended_range_breaks(expand), ...,  
             expand = c(0.04, 0))
```

```
scale_y_tufte(breaks = scales_extended_range_breaks(expand), ...,  
             expand = c(0.04, 0))
```

Arguments

breaks	see scale_x_continuous
...	additional parameters passed to continuous_scale
expand	see scale_x_continuous

Author(s)

Baptiste Auguie

See Also

[range_breaks](#)

Examples

```
(ggplot(mtcars, aes(x = wt + runif(1), y = mpg))  
  + geom_point()  
  + geom_rangeframe()  
  + theme_tufte()  
  + scale_x_tufte()  
  + scale_y_tufte()  
  )
```

show_linetypes	<i>Show linetypes</i>
----------------	-----------------------

Description

A quick and dirty way to show linetypes.

Usage

```
show_linetypes(linetypes, labels = TRUE)
```

Arguments

linetypes	A character vector of linetypes. See par .
labels	Label each line with its linetype (lty) value.

See Also

[show_col](#), [show_linetypes](#)

Examples

```
library(scales)
show_linetypes(linetype_pal()(3))
show_linetypes(linetype_pal()(3), labels=TRUE)
```

show_shapes	<i>Show shapes</i>
-------------	--------------------

Description

A quick and dirty way to show shapes.

Usage

```
show_shapes(shapes, labels = TRUE)
```

Arguments

shapes	A numeric or character vector of shapes. See par .
labels	Include the plotting character value of the symbol.

See Also

[show_col](#), [show_linetypes](#)

Examples

```
library(scales)
show_shapes(shape_pal()(5))
show_shapes(shape_pal()(3), labels=TRUE)
```

smart_digits*Format numbers with automatic number of digits*

Description

Format numbers with automatic number of digits

Usage

```
smart_digits(x, ...)
smart_digits_format(x, ...)
```

Arguments

x	A numeric vector to format
...	Parameters passed to <code>format</code>

Value

`smart_digits` returns a character vector. `smart_digits_format` returns a function with a single argument `x`, a numeric vector, that returns a character vector.

Author(s)

Josh O'Brien, Baptise Auguie, Jeffrey B. Arnold

References

Josh O'Brien, <http://stackoverflow.com/questions/23169938/select-accuracy-to-display-additional-axis-23171858#23171858>.

solarized_pal	<i>Solarized color palette (discrete)</i>
---------------	---

Description

Qualitative color palate based on the Ethan Schoonover's Solarized palette, <http://ethanschoonover.com/solarized>.

Usage

```
solarized_pal(accent = "blue")
```

Arguments

accent character Starting color. One of "yellow", "orange", "red", "magenta", "violet", "blue", "cyan", "gr

Note

For a given starting color and number of colors in the palette, the other colors are the combination of colors that maximizes the total Euclidean distance between colors in L*a*b space.

See Also

Other solarized colour: [scale_color_solarized](#), [scale_colour_solarized](#), [scale_fill_solarized](#)

Examples

```
library(scales)
show_col(solarized_pal()(2))
show_col(solarized_pal()(3))
show_col(solarized_pal("red")(4))
```

stata_linetype_pal	<i>Stata linetype palette (discrete)</i>
--------------------	--

Description

Linetype palette based on the linepattern scheme in Stata.

Usage

```
stata_linetype_pal()
```

See Also

[scale_linetype_stata](#)

Other linetype stata: [scale_linetype_stata](#)

stata_pal *Stata color palettes (discrete)*

Description

Stata color palettes. See Stata documentation for a description of the schemes, <http://www.stata.com/help.cgi?schemes>.

Usage

```
stata_pal(scheme = "s2color")
```

Arguments

scheme character. One of "s2color", "s1rcolor", "s1color", or "mono".

Examples

```
library(scales)
show_col(stata_pal("s2color")(15))
show_col(stata_pal("s1rcolor")(15))
show_col(stata_pal("s1color")(15))
show_col(stata_pal("mono")(15))
```

stata_shape_pal *Stata shape palette (discrete)*

Description

Shape palette based on the symbol palette in Stata, specifically that for the scheme s2mono.

Usage

```
stata_shape_pal()
```

See Also

See [scale_shape_stata](#) for examples.

stat_fivenumber	<i>Calculate components of a five-number summary</i>
-----------------	--

Description

The five number summary of a sample is the minimum, first quartile, median, third quartile, and maximum.

Usage

```
stat_fivenumber(mapping = NULL, data = NULL, geom = "boxplot",
  position = "dodge", na.rm = FALSE, ...)
```

Arguments

mapping	The aesthetic mapping, usually constructed with aes or aes_string . Only needs to be set at the layer level if you are overriding the plot defaults.
data	A layer specific dataset - only needed if you want to override the plot defaults.
geom	The geometric object to use display the data
position	The position adjustment to use for overlapping points on this layer
na.rm	If FALSE (the default), removes missing values with a warning. If TRUE silently removes missing values.
...	other arguments passed on to layer . This can include aesthetics whose values you want to set, not map. See layer for more details.

Value

A data frame with additional columns:

width	width of boxplot
ymin	minimum
lower	lower hinge, 25% quantile
notchlower	lower edge of notch = median - 1.58 * IQR / sqrt(n)
middle	median, 50% quantile
notchupper	upper edge of notch = median + 1.58 * IQR / sqrt(n)
upper	upper hinge, 75% quantile
ymax	maximum

Aesthetics

stat_fivenumber understands the following aesthetics (required aesthetics are in bold):

- **x**
- **y**

See Also

[stat_boxplot](#)

tableau_color_pal	<i>Color Palettes based on Tableau (discrete)</i>
-------------------	---

Description

Color palettes used in [Tableau](#).

Usage

```
tableau_color_pal(palette = "tableau10")
```

Arguments

palette Palette name. One of "tableau20", "tableau10medium", "gray5", "colorblind10", "trafficlight", "purplegray12", "bluered12", "greenorange12", "cyclic", "tableau10", "tableau10light", "purplegray6", "bluered6", "greenorange6".

References

<http://vis.stanford.edu/color-names/analyzer/>

Maureen Stone, "Designing Colors for Data" (slides), at the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging, Banff, AB, Canada, June 22, 2007 <http://www.stonesc.com/slides/CompAe%202007.pdf>.

Heer, Jeffrey and Maureen Stone, 2012 "Color Naming Models for Color Selection, Image Editing and Palette Design", ACM Human Factors in Computing Systems (CHI) <http://vis.stanford.edu/files/2012-ColorNameModels-CHI.pdf>.

See Also

Other colour tableau: [scale_color_continuous_tableau](#), [scale_color_gradient_tableau](#), [scale_colour_gradient_tableau](#), [scale_fill_continuous_tableau](#), [scale_fill_gradient_tableau](#); [scale_color_gradient2_tableau](#), [scale_colour_gradient2_tableau](#), [scale_fill_gradient2_tableau](#); [scale_color_tableau](#), [scale_colour_tableau](#), [scale_fill_tableau](#); [tableau_div_gradient_pal](#); [tableau_seq_gradient_pal](#)

Examples

```
library(scales)
show_col(tableau_color_pal("tableau20")(20))
show_col(tableau_color_pal("tableau10")(10))
show_col(tableau_color_pal("tableau10medium")(10))
show_col(tableau_color_pal("tableau10light")(10))
show_col(tableau_color_pal("colorblind10")(10))
show_col(tableau_color_pal("trafficlight")(10))
```

```
show_col(tableau_color_pal("purplegray12")(12))
show_col(tableau_color_pal("bluered12")(12))
show_col(tableau_color_pal("greenorange12")(12))
show_col(tableau_color_pal("cyclic")(20))
```

tableau_div_gradient_pal

Tableau diverging colour gradient palettes (continuous)

Description

Tableau diverging colour gradient palettes (continuous)

Usage

```
tableau_div_gradient_pal(palette = "Red-Blue", space = "Lab")
```

Arguments

palette	Palette name: One of .
space	Colour space in which to calculate gradient.

See Also

Other colour tableau: [scale_color_continuous_tableau](#), [scale_color_gradient_tableau](#), [scale_colour_gradient_tableau](#), [scale_fill_continuous_tableau](#), [scale_fill_gradient_tableau](#); [scale_color_gradient2_tableau](#), [scale_colour_gradient2_tableau](#), [scale_fill_gradient2_tableau](#); [scale_color_tableau](#), [scale_colour_tableau](#), [scale_fill_tableau](#); [tableau_color_pal](#); [tableau_seq_gradient_pal](#)

Examples

```
x <- seq(-1, 1, length = 100)
r <- sqrt(outer(x^2, x^2, "+"))
image(r, col = tableau_div_gradient_pal()(seq(0, 1, length = 12)))
image(r, col = tableau_div_gradient_pal("Orange-Blue")(seq(0, 1, length = 12)))
image(r, col = tableau_div_gradient_pal("Temperature")(seq(0, 1, length = 12)))
```

Tableau sequential colour gradient palettes (continuous)

Description

Tableau sequential colour gradient palettes (continuous)

Usage

```
tableau_seq_gradient_pal(palette = "Red", space = "Lab")
```

Arguments

palette	Palette name: One of "Red", "Green", "Blue", "Orange", "Gray", "Red Light", "Green Light", "Blue Light", "Orange Light", "Area Red", "Area Green", "Area Brown", "Blue-Green Sequential", "Brown Sequential", "Purple Sequential", "Grey Sequential".
space	Colour space in which to calculate gradient.

See Also

Other colour tableau: [scale_color_continuous_tableau](#), [scale_color_gradient_tableau](#), [scale_colour_gradient_tableau](#), [scale_fill_continuous_tableau](#), [scale_fill_gradient_tableau](#); [scale_color_gradient2_tableau](#), [scale_colour_gradient2_tableau](#), [scale_fill_gradient2_tableau](#); [scale_color_tableau](#), [scale_colour_tableau](#), [scale_fill_tableau](#); [tableau_color_pal](#); [tableau_div_gradient_pal](#)

Examples

```
library(scales)
x <- seq(0, 1, length = 25)
show_col(tableau_seq_gradient_pal("Red")(x))
show_col(tableau_seq_gradient_pal("Blue")(x))
show_col(tableau_seq_gradient_pal("Purple Sequential")(x))
```

Tableau Shape Palettes (discrete)

Description

Shape palettes used by [Tableau](#).

Usage

```
tableau_shape_pal(palette = "default")
```

Arguments

palette Palette name. One of "proportions", "default", "filled", "gender", "kpi", "thin_arrows", "weather".

See Also

Other shape tableau: [scale_shape_tableau](#)

Examples

```
show_shapes(tableau_shape_pal()(5))
```

theme_calc

Theme Calc

Description

Theme similar to the default settings of LibreOffice Calc charts.

Usage

```
theme_calc(base_size = 10, base_family = "sans")
```

Arguments

base_size base font size
base_family base font family

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]  
(d <- qplot(carat, price, data=dsamp, colour=clarity)  
+ theme_calc()  
+ scale_color_calc())  
(d <- qplot(carat, price, data=dsamp, shape=cut)  
+ theme_calc()  
+ scale_shape_calc())
```

theme_economist	<i>ggplot color theme based on the Economist</i>
-----------------	--

Description

Style plots similar to those in *The Economist*.

Usage

```
theme_economist(base_size = 10, base_family = "sans", horizontal = TRUE,
  dkpanel = FALSE, stata = FALSE)
```

```
theme_economist_white(base_size = 11, base_family = "sans",
  gray_bg = TRUE, horizontal = TRUE)
```

Arguments

base_size	numeric base font size
base_family	character base font family
horizontal	logical. Horizontal axis lines?
dkpanel	logical Darker background for panel region?
stata	logical Use RGB values from Stata's economist scheme.
gray_bg	logical If TRUE, use gray background, else use white background.

Details

theme_economist implements the standard bluish-gray background theme in the print *The Economist* and economist.com. theme_economist_white implements a variant with a white panel and light gray (or white) background used by *The Economist* blog [Graphic Detail](#).

The Economist uses "ITC Officina Sans" as its font for graphs. If you have access to this font, you can use it with the **extrafont** package. "Verdana" is a good substitute.

Value

An object of class [theme](#).

References

- [The Economist](#)
- [Spiekerblog, "ITC Officina Display", January 1, 2007.](#)
- <http://www.economist.com/help/about-us>

See Also

[theEconomist.theme](#) for an Economist theme for lattice plots.

Examples

```

dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
q <- (qplot(carat, price, data=dsamp, colour=clarity)
      + ggtitle("Diamonds Are Forever"))

## Standard
q + theme_economist() + scale_colour_economist()

## Stata colors
q + theme_economist(stata=TRUE) + scale_colour_economist(stata=TRUE)

## Darker plot region
q + theme_economist(dkpanel=TRUE) + scale_colour_economist(stata=TRUE)

## Darker plot region is best for for facets
dkblue <- ggthemes_data$economist$fg['blue_dark']
(ggplot(data=dsamp, aes(x=carat, y=price))
 + geom_point(colour=dkblue)
 + facet_grid(. ~ cut )
 + theme_economist(dkpanel=TRUE))

##' ## Change axis lines to vertical
(q + theme_economist(horizontal=FALSE)
 + scale_colour_economist() + coord_flip())

## White panel/light gray background
(q + theme_economist_white()
 + scale_colour_economist())

## All white variant
(q + theme_economist_white(gray_bg=FALSE)
 + scale_colour_economist())
## Not run:
## The Economist uses ITC Officina Sans
library(extrafont)
(q + theme_economist(base_family="ITC Officina Sans")
 + scale_colour_economist())

## Verdana is a widely available substitute
(q + theme_economist(base_family="Verdana")
 + scale_colour_economist())

## End(Not run)

```

theme_excel

ggplot color theme based on old Excel plots

Description

Theme to replicate the ugly monstrosity that was the old gray-background Excel chart. Please never use this.

Usage

```
theme_excel(base_size = 12, base_family = "", horizontal = TRUE)
```

Arguments

```
base_size      numeric base font size
base_family    character base font family
horizontal     logical. Horizontal axis lines?
```

Value

An object of class `theme`.

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
# Old line color palette
(qplot(carat, price, data=dsamp, colour=clarity)
+ theme_excel()
+ scale_colour_excel() )
# Old fill color palette
(ggplot(diamonds, aes(clarity, fill=cut))
+ geom_bar()
+ scale_fill_excel("fill")
+ theme_excel())
```

theme_few

Theme based on Few's "Practical Rules for Using Color in Charts"

Description

Theme based on the rules and examples in "[Practical Rules for Using Color in Charts](#)".

Usage

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

Arguments

```
base_size      base font size
base_family    base font family
```

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
(qplot(carat, price, data=dsamp, colour=clarity)
+ theme_few()
+ scale_colour_few())
(qplot(carat, price, data=dsamp, colour=clarity)
+ theme_few()
+ scale_colour_few("dark"))
(ggplot(diamonds, aes(clarity, fill=cut))
+ geom_bar()
+ theme_few()
+ scale_fill_few("light"))
```

theme_fivethirtyeight *Theme inspired by fivethirtyeight.com plots*

Description

Theme inspired by the plots on <http://fivethirtyeight.com>.

Usage

```
theme_fivethirtyeight(base_size = 12, base_family = "sans")
```

Arguments

base_size	Base font size.
base_family	Plot text font family.

Examples

```
(qplot(hp, mpg, data= subset(mtcars, cyl != 5), geom="point", color = factor(cyl))
+ ggtitle("Horsepower, mpg and cylinders")
+ geom_smooth(method = "lm", se = FALSE)
+ scale_color_fivethirtyeight()
+ theme_fivethirtyeight())
```

theme_foundation *Foundation Theme*

Description

This theme is designed to be a foundation from which to build new themes, and not meant to be used directly. `theme_foundation` is a complete theme with only minimal number of elements defined.

Usage

```
theme_foundation(base_size = 12, base_family = "", use_sizes = TRUE)
```

Arguments

<code>base_size</code>	Base font size.
<code>base_family</code>	Base font family.
<code>use_sizes</code>	If TRUE, then define sizes and locations with reasonable defaults taken from theme_gray .

Details

It is easier to create new themes by extending this one rather than `theme_gray` or `theme_bw`, because those themes those themes define elements deep in the hierarchy.

See Also

Other themes: [theme_igray](#); [theme_solid](#)

theme_gdocs *Theme with Google Docs Chart defaults*

Description

Theme similar to the default look of charts in Google Docs.

Usage

```
theme_gdocs(base_size = 12, base_family = "sans")
```

Arguments

<code>base_size</code>	base font size
<code>base_family</code>	base font family

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
(d <- qplot(carat, price, data=dsamp, colour=clarity)
+ theme_gdocs()
+ ggtitle("Diamonds")
+ scale_color_gdocs())
```

 theme_hc

Highcharts JS theme

Description

Theme based on the plots in *Highcharts JS*.

Usage

```
theme_hc(base_size = 12, base_family = "sans", bgcolor = "default")
```

Arguments

base_size	base font size
base_family	base font family
bgcolor	The background color of plot. One of "default", "darkunica", the names of values in ggthemes_data\$hc\$bg.

References

<http://www.highcharts.com/demo/line-basic>

<https://github.com/highslide-software/highcharts.com/tree/master/js/themes>

Examples

```
(qplot(hp, mpg, data = mtcars, geom = "point")
+ scale_colour_hc()
+ ggtitle("Diamond Prices")
+ theme_hc())
## Use a Dark-Unica theme
(qplot(hp, mpg, data = mtcars, geom = "point")
+ scale_colour_hc("darkunica")
+ ggtitle("Diamond Prices")
+ theme_hc(bgcolor = "darkunica"))
```

theme_igray	<i>Inverse gray theme</i>
-------------	---------------------------

Description

Theme with white panel and gray background.

Usage

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

Arguments

base_size	base font size
base_family	base font family

Details

This theme inverts the colors in the [theme_gray](#), a white panel and a light gray area around it. This keeps a white background for the color scales like [theme_bw](#). But by using a gray background, the plot is closer to the typographical color of the document, which is the motivation for using a gray panel in [theme_gray](#). This is similar to the style of plots in Stata and Tableau.

See Also

[theme_gray](#), [theme_bw](#)

Other themes: [theme_foundation](#); [theme_solid](#)

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
(d <- qplot(carat, price, data=dsamp, colour=clarity)
  + theme_igray())
```

theme_pander	<i>A ggplot theme originated from the pander package</i>
--------------	--

Description

The **pander** ships with a default theme when the "unify plots" option is enabled via `panderOptions`, which is now also available outside of **pander** internals, like `evals`, `eval.msgs` or `Pandoc.brew`.

Usage

```
theme_pander(base_size = 12, base_family = "sans", nomargin = TRUE,
             ff = NULL, fc = "black", fs = NULL, gM = TRUE, gm = TRUE,
             gc = "grey", gl = "dashed", boxes = FALSE, bc = "white",
             pc = "transparent", lp = "right", axis = 1)
```

Arguments

base_size	base font size
base_family	base font family
nomargin	suppress the white space around the plot (boolean)
ff	font family, like sans. Deprecated: use base_family instead.
fc	font color (name or hexa code)
fs	font size (integer). Deprecated: use base_size instead.
gM	major grid (boolean)
gm	minor grid (boolean)
gc	grid color (name or hexa code)
gl	grid line type (lty)
boxes	to render a border around the plot or not
bc	background color (name or hexa code)
pc	panel background color (name or hexa code)
lp	legend position
axis	axis angle as defined in par(las)

Examples

```
## Not run:
p <- qplot(mpg, wt, data = mtcars)
p + theme_pander()

panderOptions('graph.grid.color', 'red')
p + theme_pander()

p <- ggplot(mtcars, aes(wt, mpg, colour = factor(cyl))) + geom_point()
p + theme_pander() + scale_color_pander()

## standard examples of the ggtheme package
qplot(carat, price, data = diamonds, colour = cut) + theme_pander() + scale_colour_pander()
ggplot(diamonds, aes(clarity, fill = cut)) + geom_bar() + scale_fill_pander() + theme_pander()

## End(Not run)
```

theme_solarized *ggplot color themes based on the Solarized palette*

Description

See <http://ethanschoonover.com/solarized> for a description of the Solarized palette.

Usage

```
theme_solarized(base_size = 12, base_family = "", light = TRUE)
```

```
theme_solarized_2(base_size = 12, base_family = "", light = TRUE)
```

Arguments

base_size	base font size
base_family	base font family
light	logical. Light or dark theme?

Details

Plots made with this theme integrate seamlessly with the Solarized Beamer color theme. <https://github.com/jrnold/beamercolorthemesolarized>. There are two variations: theme_solarized is similar to [theme_bw](#), while theme_solarized_2 is similar to [theme_gray](#).

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
(qplot(carat, price, data=dsamp, colour=clarity)
+ theme_solarized()
+ scale_colour_solarized("blue"))
## Dark version
(qplot(carat, price, data=dsamp, colour=clarity)
+ theme_solarized(light=FALSE)
+ scale_colour_solarized("blue"))
## With panels
(ggplot(dsamp, aes(x = carat, y = price))
+ geom_point(color = ggthemes_data$solarized$base['base00'])
+ facet_wrap(~ clarity)
+ theme_solarized())
## Alternative version
(qplot(carat, price, data=dsamp, color=clarity)
+ theme_solarized_2(light=FALSE)
+ scale_colour_solarized("blue"))
```

theme_solid	<i>Theme with nothing other than a background color</i>
-------------	---

Description

Theme that removes all non-geom elements (lines, text, etc), This theme is when only the geometric objects are desired.

Usage

```
theme_solid(base_size = 12, base_family = "", fill = NA)
```

Arguments

base_size	Base font size.
base_family	Ignored, kept for consistency with theme.
fill	Background color of the plot.

See Also

Other themes: [theme_foundation](#); [theme_igray](#)

Examples

```
(ggplot(mtcars, aes(wt, mpg))
+ geom_point()
+ theme_solid(fill = "white"))
```

theme_stata	<i>Themes based on Stata graph schemes</i>
-------------	--

Description

Themes based on Stata graph schemes

Usage

```
theme_stata(base_size = 11, base_family = "sans", scheme = "s2color")
```

Arguments

base_size	base font size
base_family	base font family
scheme	One of "s2color", "s2mono", "s1color", "s1rcolor", or "s1mono", "s2manual", "s1manual", or "sj"

Note

Stata graph schemes include what **ggplot2** separates into themes and scales, as well as defaults specific to different graph types (which ggplot does not support).

References

<http://www.stata.com/help.cgi?schemes>

Examples

```
dsamp <- diamonds[sample(nrow(diamonds), 1000), ]
q1 <- (qplot(carat, price, data=dsamp, colour=clarity)
      + ggtitle("Diamonds"))
q2 <- (qplot(carat, price, data=dsamp)
      + facet_wrap(~ clarity)
      + ggtitle("Diamonds"))
q1mono <- (qplot(carat, price, shape=clarity, color=clarity,
                data=dsamp)
          + scale_shape_stata()
          + ggtitle("Diamonds"))

## s2color
(q1 + theme_stata() + scale_colour_stata(scheme = "s2color"))
(q2 + theme_stata())

## s2mono
(q1mono + theme_stata(scheme = "s2mono") + scale_colour_stata("mono"))
(q2 + theme_stata(scheme = "s2mono"))

## s1color
(q1 + theme_stata(scheme = "s1color") + scale_colour_stata("s1color"))
(q2 + theme_stata(scheme = "s1color"))

## s1rcolor
(q1 + theme_stata(scheme = "s1rcolor") + scale_colour_stata("s1rcolor"))
(ggplot(dsamp, aes(x=carat, y=price)) + geom_point(colour="white")
 + facet_wrap(~ clarity) + scale_colour_stata("s1rcolor")
 + ggtitle("Diamonds"))

## s1mono
(q1mono + theme_stata(scheme = "s1mono") + scale_colour_stata("mono"))
(q2 + theme_stata(scheme = "s1mono"))
```

theme_tufte

Tufte Maximal Data, Minimal Ink Theme

Description

Theme based on Chapter 6 "Data-Ink Maximization and Graphical Design" of Edward Tufte *The Visual Display of Quantitative Information*. No border, no axis lines, no grids. This theme works best in combination with [geom_rug](#) or [geom_rangeframe](#).

Usage

```
theme_tufte(base_size = 11, base_family = "serif", ticks = TRUE)
```

Arguments

base_size	Base font size
base_family	Base font family
ticks	logical Show axis ticks?

Note

The default font family is set to "serif" as he uses serif fonts for labels in "The Visual Display of Quantitative Information". The serif font used by Tufte in his books is a variant of Bembo, while the sans serif font is Gill Sans. If these fonts are installed on your system, then you can use them with the package **extrafont**.

References

Tufte, Edward R. (2001) The Visual Display of Quantitative Information, Chapter 6.

Examples

```
# with ticks and range frames
(ggplot(mtcars, aes(wt, mpg))
 + geom_point() + geom_rangeframe()
 + theme_tufte())
# with geom_rug
(ggplot(mtcars, aes(wt, mpg))
 + geom_point() + geom_rug()
 + theme_tufte(ticks=FALSE))
## Not run:
## Using the Bembo serif family
library(extrafont)
(ggplot(mtcars, aes(wt, mpg))
 + geom_point() + geom_rangeframe()
 + theme_tufte(base_family="BemboStd"))
## Using the Gill Sans sans serif family
(ggplot(mtcars, aes(wt, mpg))
 + geom_point() + geom_rangeframe()
 + theme_tufte(base_family="GillSans"))

## End(Not run)
```

 theme_wsj

 Wall Street Journal theme

Description

Theme based on the plots in *The Wall Street Journal*.

Usage

```
theme_wsj(base_size = 12, color = "brown", base_family = "sans",
          title_family = "mono")
```

Arguments

base_size	Base font size.
color	The background color of plot. One of "brown", "gray", "green", "blue", the names of values in ggthemes_data\$wsj\$bg.
base_family	Plot text font family.
title_family	Plot title font family.

References

<https://twitter.com/WSJGraphics>

<http://pinterest.com/wsjgraphics/wsj-graphics/>

Examples

```
(qplot(hp, mpg, data=mtcars, geom="point")
+ scale_colour_wsj("colors6", "")
+ ggtitle("Diamond Prices")
+ theme_wsj())
## Use a gray background instead
(qplot(hp, mpg, data=mtcars, geom="point")
+ scale_colour_wsj("colors6", "")
+ ggtitle("Diamond Prices")
+ theme_wsj(color="gray"))
```

tremmel_shape_pal	<i>Shape palette from Tremmel (1995) (discrete)</i>
-------------------	---

Description

Based on experiments Tremmel (1995) suggests the following shape palettes:

Usage

```
tremmel_shape_pal(overlap = FALSE, n3alt = TRUE)
```

Arguments

overlap	use an empty circle instead of a solid circle when $n == 2$.
n3alt	If TRUE then use a solid circle, plus sign and empty triangle, else use a solid circle, empty circle, and empty triangle.

Details

If two symbols, then use a solid circle and plus sign.

If three symbols, then use a solid circle, empty circle, and an empty triangle. However, that set of symbols does not satisfy the requirement that each symbol should differ from the other symbols in the same feature dimension. A set of three symbols that satisfies this is a circle (curvature), plus sign (number of terminators), triangle (line orientation).

If more than three groups of data, then separate the groups into different plots.

References

Tremmel, Lothar, (1995) "The Visual Separability of Plotting Symbols in Scatterplots" Journal of Computational and Graphical Statistics, <http://www.jstor.org/stable/1390760>

See Also

Other shapes: [circlefill_shape_pal](#); [cleveland_shape_pal](#); [scale_shape_circlefill](#); [scale_shape_cleveland](#); [scale_shape_tremmel](#)

wsj_pal

Wall Street Journal color palette (discrete)

Description

The Wall Street Journal uses many different color palettes in its plots. This collects a few of them, but is by no means exhaustive. Collections of these plots can be found on the WSJ Graphics [Twitter](#) feed and [Pinterest](#).

Usage

```
wsj_pal(palette = "colors6")
```

Arguments

`palette` character The color palette to use. This must be a name in `ggthemes_datawsjpalettes`.

Palettes

The following palettes are defined,

rgby Red/Green/Blue/Yellow theme. Examples: <http://twitpic.com/b2e3v2>.

green_red Green/red two-color scale for good/bad. Examples: <http://twitpic.com/b1avj6>, <http://twitpic.com/a4kxcl>.

green_black Black-green 4-color scale for "Very negative", "Somewhat negative", "somewhat positive", "very positive". Examples: <http://twitpic.com/awbua0>.

dem_rep Democrat/Republican/Undecided blue/red/gray scale. Examples: <http://twitpic.com/awbua0>.

colors6 Red,blue,gold,green,orange, and black palette. Examples: <http://twitpic.com/9gfg5q>.

See Also

Other colour wsj: [scale_color_wsj](#), [scale_colour_wsj](#), [scale_fill_wsj](#)

Index

*Topic **datasets**

- ggthemes_data, 17

- aes, 14, 15, 41
- aes_string, 14, 15, 41

- bank_slopes, 3
- banking, 5

- calc_pal, 6, 29
- calc_shape_pal, 6, 32
- circlefill_shape_pal, 7, 8, 33, 35, 59
- cleveland_shape_pal, 7, 7, 33, 35, 59
- colorblind_pal, 8
- continuous_scale, 36

- dichromat_pal, 9
- discrete_scale, 9, 22–35

- economist_pal, 9, 22
- excel_pal, 10, 30
- extended_range_breaks, 10

- few_pal, 12, 23
- fivethirtyeight_pal, 12, 24
- format, 38

- gdocs_pal, 13, 30
- geom_boxplot, 16
- geom_rangeframe, 13, 16, 56
- geom_rug, 56
- geom_tufteboxplot, 14, 15
- ggthemes, 16
- ggthemes-package (ggthemes), 16
- ggthemes_data, 17
- ggthemes_data\$hc\$palettes, 20, 26
- ggthemes_data\$wsj\$palettes, 28, 59

- hc_pal, 20, 26

- layer, 14, 15, 41

- nlm, 4

- palette_pander, 21, 22
- par, 37

- scale_color_calc, 6
- scale_color_calc (scale_fill_calc), 29
- scale_color_colorblind (colorblind_pal), 8
- scale_color_continuous_tableau, 24, 28, 42–44
- scale_color_continuous_tableau (scale_colour_gradient_tableau), 25
- scale_color_economist, 9
- scale_color_economist (scale_colour_economist), 22
- scale_color_excel, 10
- scale_color_excel (scale_fill_excel), 29
- scale_color_few, 12
- scale_color_few (scale_colour_few), 23
- scale_color_fivethirtyeight, 13
- scale_color_fivethirtyeight (scale_colour_fivethirtyeight), 23
- scale_color_gdocs, 13
- scale_color_gdocs (scale_fill_gdocs), 30
- scale_color_gradient2_tableau, 26, 28, 42–44
- scale_color_gradient2_tableau (scale_colour_gradient2_tableau), 24
- scale_color_gradient_tableau, 24, 28, 42–44
- scale_color_gradient_tableau (scale_colour_gradient_tableau), 25
- scale_color_hc (scale_colour_hc), 26
- scale_color_pander, 21, 21
- scale_color_solarized, 39

- scale_color_solarized
(scale_fill_solarized), 31
- scale_color_stata (scale_colour_stata),
26
- scale_color_tableau, 9, 24, 26, 42–44
- scale_color_tableau
(scale_colour_tableau), 27
- scale_color_wsj, 60
- scale_color_wsj (scale_colour_wsj), 28
- scale_colour_calc, 6
- scale_colour_calc (scale_fill_calc), 29
- scale_colour_colorblind
(colorblind_pal), 8
- scale_colour_economist, 9, 22
- scale_colour_excel, 10
- scale_colour_excel (scale_fill_excel),
29
- scale_colour_few, 12, 23
- scale_colour_fivethirtyeight, 13, 23
- scale_colour_gdocs, 13
- scale_colour_gdocs (scale_fill_gdocs),
30
- scale_colour_gradient2_tableau, 24, 26,
28, 42–44
- scale_colour_gradient_tableau, 24, 25,
28, 42–44
- scale_colour_hc, 26
- scale_colour_pander, 21
- scale_colour_pander
(scale_color_pander), 21
- scale_colour_solarized, 39
- scale_colour_solarized
(scale_fill_solarized), 31
- scale_colour_stata, 26
- scale_colour_tableau, 24, 26, 27, 42–44
- scale_colour_wsj, 28, 60
- scale_fill_calc, 6, 29
- scale_fill_colorblind (colorblind_pal),
8
- scale_fill_continuous_tableau, 24, 28,
42–44
- scale_fill_continuous_tableau
(scale_colour_gradient_tableau),
25
- scale_fill_economist, 9
- scale_fill_economist
(scale_colour_economist), 22
- scale_fill_excel, 10, 29
- scale_fill_few, 12
- scale_fill_few (scale_colour_few), 23
- scale_fill_fivethirtyeight, 13
- scale_fill_fivethirtyeight
(scale_colour_fivethirtyeight),
23
- scale_fill_gdocs, 13, 30
- scale_fill_gradient2_tableau, 26, 28,
42–44
- scale_fill_gradient2_tableau
(scale_colour_gradient2_tableau),
24
- scale_fill_gradient_tableau, 24, 28,
42–44
- scale_fill_gradient_tableau
(scale_colour_gradient_tableau),
25
- scale_fill_hc (scale_colour_hc), 26
- scale_fill_pander, 21
- scale_fill_pander (scale_color_pander),
21
- scale_fill_solarized, 31, 39
- scale_fill_stata (scale_colour_stata),
26
- scale_fill_tableau, 24, 26, 42–44
- scale_fill_tableau
(scale_colour_tableau), 27
- scale_fill_wsj, 60
- scale_fill_wsj (scale_colour_wsj), 28
- scale_linetype_stata, 31, 39
- scale_shape_calc, 6, 32
- scale_shape_circlefill, 7, 8, 32, 33, 35, 59
- scale_shape_cleveland, 7, 8, 33, 33, 35, 59
- scale_shape_stata, 34, 40
- scale_shape_tableau, 34, 45
- scale_shape_tremmel, 7, 8, 33, 35, 59
- scale_x_continuous, 11, 36
- scale_x_tufte, 11, 36
- scale_y_tufte, 11
- scale_y_tufte (scale_x_tufte), 36
- scales_extended_range_breaks
(extended_range_breaks), 10
- show_col, 37
- show_linetypes, 37, 37
- show_shapes, 37
- smart_digits, 38
- smart_digits_format (smart_digits), 38
- solarized_pal, 31, 39

stat_boxplot, 42
stat_fivenumber, 41
stata_linetype_pal, 31, 32, 39
stata_pal, 26, 40
stata_shape_pal, 34, 40

tableau_color_pal, 24, 26–28, 42, 43, 44
tableau_div_gradient_pal, 24, 26, 28, 42,
43, 44
tableau_seq_gradient_pal, 24, 26, 28, 42,
43, 44
tableau_shape_pal, 34, 44
theEconomist.theme, 46
theme, 46, 48
theme_bw, 52, 54
theme_calc, 29, 32, 45
theme_economist, 22, 46
theme_economist_white
 (theme_economist), 46
theme_excel, 30, 47
theme_few, 48
theme_fivethirtyeight, 24, 49
theme_foundation, 50, 52, 55
theme_gdocs, 30, 50
theme_gray, 50, 52, 54
theme_hc, 26, 51
theme_igray, 50, 52, 55
theme_pander, 22, 52
theme_solarized, 54
theme_solarized_2 (theme_solarized), 54
theme_solid, 50, 52, 55
theme_stata, 55
theme_tufte, 56
theme_wsj, 28, 57
tremmel_shape_pal, 7, 8, 33, 35, 58

wsj_pal, 28, 59