

# Package: ggcharts (via r-universe)

September 10, 2024

**Type** Package

**Title** Get You to Your Desired Plot Faster

**Version** 0.2.1

**Description** Streamlines the creation of common charts by taking care of a lot of data preprocessing and plot customization for the user. Provides a high-level interface for creating plots using 'ggplot2'.

**Depends** R (>= 3.5.0), ggplot2 (>= 3.0.0)

**Imports** colorspace, dplyr, lifecycle, magrittr, patchwork, rlang, showtext, sysfonts

**Suggests** gapminder, knitr, lintr, rmarkdown, scales, spelling, tibble, tidy, testthat (>= 2.1.0)

**License** MIT + file LICENSE

**URL** <https://thomas-neitmann.github.io/ggcharts/index.html>,  
<https://github.com/thomas-neitmann/ggcharts>

**BugReports** <https://github.com/thomas-neitmann/ggcharts/issues>

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.0

**VignetteBuilder** knitr

**Language** en-US

**Repository** <https://thomas-neitmann.r-universe.dev>

**RemoteUrl** <https://github.com/thomas-neitmann/ggcharts>

**RemoteRef** HEAD

**RemoteSha** fa263d12878ece10d5eadfed26b27e2d233d28b1

## Contents

bar_chart	2
biomedicalrevenue	5
diverging_bar_chart	5
diverging_lollipop_chart	7
dumbbell_chart	8
ggcharts_get_default_color	10
ggcharts_get_theme	11
highlight_spec	12
line_chart	13
lollipop_chart	14
popch	16
popurope	17
pyramid_chart	18
revenue_wide	19
theme_coffee	20
theme_ggcharts	21
theme_hermit	23
theme_ng	24
theme_nightblue	26
<b>Index</b>	<b>28</b>

---

bar_chart	<i>Bar Chart</i>
-----------	------------------

---

### Description

Easily create a bar chart

### Usage

```
bar_chart(
  data,
  x,
  y,
  facet = NULL,
  ...,
  bar_color = "auto",
  highlight = NULL,
  sort = TRUE,
  horizontal = TRUE,
  top_n = NULL,
  threshold = NULL,
  other = FALSE,
  limit = NULL
)
```

```
column_chart(  
  data,  
  x,  
  y,  
  facet = NULL,  
  ...,  
  bar_color = "auto",  
  highlight = NULL,  
  sort = NULL,  
  horizontal = FALSE,  
  top_n = NULL,  
  threshold = NULL,  
  limit = NULL  
)
```

### Arguments

data	Dataset to use for the bar chart
x	character or factor column of data
y	numeric column of data representing the bar length. If missing, the bar length will be proportional to the count of each value in x.
facet	character or factor column of data defining the faceting groups
...	Additional arguments passed to aes()
bar_color	character. The color of the bars
highlight	character. One or more value(s) of x that should be highlighted in the plot
sort	logical. Should the data be sorted before plotting?
horizontal	logical. Should the plot be oriented horizontally?
top_n	numeric. If a value for top_n is provided only the top top_n records will be displayed
threshold	numeric. If a value for threshold is provided only records with y > threshold will be displayed
other	logical. Should all x with y < threshold be summarized in a group called 'other' and be displayed at the bottom of the chart?
limit	Deprecated. use top_n instead.

### Details

Both top\_n and threshold only work when sort = TRUE. Attempting to use them when sort = FALSE will result in an error. Furthermore, only top\_n or threshold can be used at a time. Providing a value for both top\_n and threshold will result in an error as well.

column\_chart() is a shortcut for bar\_chart() with horizontal = FALSE and sort = FALSE if x is numeric.

**Value**

An object of class `ggplot`

**Author(s)**

Thomas Neitmann

**See Also**

For more details have a look at these vignettes: `vignette("highlight", package = "ggcharts")`  
`vignette("customize", package = "ggcharts")`

**Examples**

```
data(biomedicalrevenue)
revenue2018 <- biomedicalrevenue[biomedicalrevenue$year == 2018, ]
revenue_roche <- biomedicalrevenue[biomedicalrevenue$company == "Roche", ]

## By default bar_chart() creates a horizontal and sorted plot
bar_chart(revenue2018, company, revenue)

## If the `y` argument is missing the count of each value in `x` is displayed
bar_chart(mtcars, cyl)

## Create a vertical, non-sorted bar chart
bar_chart(revenue_roche, year, revenue, horizontal = FALSE, sort = FALSE)

## column_chart() is a shortcut for the above
column_chart(revenue_roche, year, revenue)

## Limit the number of bars to the top 10
bar_chart(revenue2018, company, revenue, top_n = 10)

## Display only companies with revenue > 40B.
bar_chart(revenue2018, company, revenue, threshold = 40)

## Change the bar color
bar_chart(revenue2018, company, revenue, bar_color = "purple")

## Highlight a single bar
bar_chart(revenue2018, company, revenue, top_n = 10, highlight = "Roche")

## Use facets to show the top 5 companies over the years
bar_chart(biomedicalrevenue, company, revenue, facet = year, top_n = 5)
```

---

biomedicalrevenue      *Top Biomedical Companies Revenues*

---

**Description**

Annual revenues of top biomedical companies from 2011 to 2018

**Usage**

```
biomedicalrevenue
```

**Format**

A data frame with 224 rows and 3 variables:

**company** Name of the company

**year** Fiscal year

**revenue** Revenue in billion USD

**Source**

[https://en.wikipedia.org/wiki/List\\_of\\_largest\\_biomedical\\_companies\\_by\\_revenue](https://en.wikipedia.org/wiki/List_of_largest_biomedical_companies_by_revenue)

---

diverging\_bar\_chart      *Diverging Bar Chart*

---

**Description**

Easily create a diverging bar chart

**Usage**

```
diverging_bar_chart(  
  data,  
  x,  
  y,  
  bar_colors = c("#1F77B4", "#FF7F0E"),  
  text_color = "auto",  
  text_size = 10  
)
```

**Arguments**

<code>data</code>	Dataset to use for the diverging bar chart
<code>x</code>	character or factor column of data
<code>y</code>	numeric column of data representing the bar length
<code>bar_colors</code>	A character vector of length 2 containing the colors for the positive and negative bars
<code>text_color</code>	character. The color for the bar annotations
<code>text_size</code>	numeric. The size of the bar annotation text in pt

**Value**

An object of class `ggplot`

**Author(s)**

Thomas Neitmann

**See Also**

To learn how to further customize this plot have a look at the 'customize' vignette: `vignette("customize", package = "ggcharts")`

**Examples**

```
if (requireNamespace("tidyr")) {
  library(magrittr)
  data(biomedicalrevenue)
  biomedicalrevenue %>%
  dplyr::filter(year > 2016) %>%
  tidyr::pivot_wider(
    values_from = revenue,
    names_from = year,
    names_prefix = "revenue_"
  ) %>%
  dplyr::mutate(diff = revenue_2018 - revenue_2017) %>%
  diverging_bar_chart(company, diff)
}

data(mtcars)
mtcars_z <- dplyr::transmute(
  .data = mtcars,
  model = row.names(mtcars),
  hpz = scale(hp)
)

diverging_bar_chart(mtcars_z, model, hpz)

## Change the colors
diverging_bar_chart(mtcars_z, model, hpz, bar_color = c("darkgreen", "darkred"))
```

```
## Decrease the axis label font size
diverging_bar_chart(mtcars_z, model, hpz, text_size = 8)

## Display the axis label text in the same color as the bars
diverging_bar_chart(mtcars_z, model, hpz, text_color = c("#1F77B4", "#FF7F0E"))
```

---

diverging\_lollipop\_chart

*Diverging Lollipop Chart*

---

## Description

Easily create a diverging lollipop chart

## Usage

```
diverging_lollipop_chart(
  data,
  x,
  y,
  lollipop_colors = c("#1F77B4", "#FF7F0E"),
  line_size = 0.75,
  point_size = 3,
  text_color = "auto",
  text_size = 10
)
```

## Arguments

<code>data</code>	Dataset to use for the diverging lollipop chart
<code>x</code>	character or factor column of data
<code>y</code>	numeric column of data representing the lollipop length
<code>lollipop_colors</code>	A character vector of length 2 containing the colors for the positive and negative lollipops
<code>line_size</code>	numeric. Size of the lollipop 'stick'
<code>point_size</code>	numeric. Size of the lollipop 'head'
<code>text_color</code>	character. The color for the lollipop annotations
<code>text_size</code>	numeric The size of the lollipop annotation text in pt

## Value

An object of class `ggplot`

**Author(s)**

Thomas Neitmann

**See Also**

To learn how to further customize this plot have a look at the 'customize' vignette: `vignette("customize", package = "ggcharts")`

**Examples**

```

if (requireNamespace("tidyr")) {
  library(magrittr)
  data(biomedicalrevenue)
  biomedicalrevenue %>%
  dplyr::filter(year > 2016) %>%
  tidyr::pivot_wider(
    values_from = revenue,
    names_from = year,
    names_prefix = "revenue_"
  ) %>%
  dplyr::mutate(diff = revenue_2018 - revenue_2017) %>%
  diverging_lollipop_chart(company, diff)
}

data(mtcars)
mtcars_z <- dplyr::transmute(
  .data = mtcars,
  model = row.names(mtcars),
  hpz = scale(hp)
)

diverging_lollipop_chart(mtcars_z, model, hpz)

## Change the colors
diverging_lollipop_chart(mtcars_z, model, hpz, lollipop_colors = c("darkgreen", "darkred"))

## Decrease the axis label font size
diverging_lollipop_chart(mtcars_z, model, hpz, text_size = 8)

## Display the axis label text in the same color as the bars
diverging_lollipop_chart(mtcars_z, model, hpz, text_color = c("#1F77B4", "#FF7F0E"))

```

---

dumbbell\_chart

*Dumbbell Chart*


---

**Description**

Easily create a dumbbell chart



**Usage**

```
dumbbell_chart(  
  data,  
  x,  
  y1,  
  y2,  
  line_size = 1.5,  
  line_color = "lightgray",  
  point_size = 4,  
  point_colors = c("#1F77B4", "#FF7F0E"),  
  sort = TRUE,  
  horizontal = TRUE,  
  top_n = NULL,  
  legend = TRUE,  
  legend_labels = waiver(),  
  limit = NULL  
)
```

**Arguments**

data	Dataset to use for the dumbbell chart
x	character or factor column of data
y1	numeric column of data representing the dumbbell end
y2	numeric column of data representing the dumbbell start
line_size	numeric. Line width
line_color	character. Line color
point_size	numeric. Point size
point_colors	numeric. Point color
sort	logical. Should the data be sorted by y2 before plotting?
horizontal	logical. Should the plot be displayed horizontally?
top_n	integer. If a value for top_n is provided only the first top_n records will be displayed
legend	logical. Should a legend be displayed?
legend_labels	character. Custom labels to be displayed in the legend
limit	Deprecated. use top_n instead.

**Value**

An object of class `ggplot`

**Author(s)**

Thomas Neitmann

**See Also**

To learn how to further customize this plot have a look at the 'customize' vignette: `vignette("customize", package = "ggcharts")`

**Examples**

```
data(popeurope)

dumbbell_chart(popeurope, country, pop1952, pop2007)

# Display only the top 10 countries in terms of population in 2007
dumbbell_chart(popeurope, country, pop1952, pop2007, top_n = 10)

# Change line and point color
dumbbell_chart(popeurope, country, pop1952, pop2007, top_n = 10,
               line_color = "lightgray", point_color = c("lightgray", "black"))

# Add custom legend labels
dumbbell_chart(popeurope, country, pop1952, pop2007, top_n = 10,
               legend_labels = c("1952", "2007"))

# Increase line width and point size
dumbbell_chart(popeurope, country, pop1952, pop2007, top_n = 10,
               line_size = 2, point_size = 5)
```

---

ggcharts\_get\_default\_color

*Get the Default Color for a ggcharts Theme*

---

**Description**

Retrieve the color used by default for a given ggcharts theme

**Usage**

```
ggcharts_get_default_color(theme)
```

**Arguments**

theme                    character. The name of a ggcharts theme.

**Value**

The default color for the given theme as a character

**Author(s)**

Thomas Neitmann

## Examples

```
ggcharts_get_default_color("theme_hermit")
ggcharts_get_default_color("theme_ng")
```

---

ggcharts\_get\_theme      *Get and Set the Currently Active ggcharts Theme*

---

## Description

The current theme is automatically applied to any plot created with ggcharts. It does not affect plots created with ggplot2.

## Usage

```
ggcharts_get_theme()

ggcharts_set_theme(theme, ...)
```

## Arguments

theme	character. The name of the theme, e.g. "theme_hermit"
...	Additional argument passed onto the specified theme

## Value

ggchart\_set\_theme invisibly returns the name of the previously active theme as a character. ggchart\_get\_theme returns the name of the currently active theme as a character.

## Author(s)

Thomas Neitmann

## Examples

```
data("diamonds", package = "ggplot2")

## By default `theme_ggcharts()` is used
ggcharts_get_theme()
bar_chart(diamonds, cut)

ggcharts_set_theme("theme_hermit")
bar_chart(diamonds, cut)

ggcharts_set_theme("theme_ng")
bar_chart(diamonds, cut)

ggcharts_set_theme("theme_nightblue", base_size = 18, base_family = "serif")
```

```
bar_chart(diamonds, cut)

## Restore the default
ggcharts_set_theme("theme_ggcharts")
```

---

highlight_spec	<i>Highlight Specification</i>
----------------	--------------------------------

---

### Description

Create a highlight specification to pass on to a chart function

### Usage

```
highlight_spec(what, highlight_color = NULL, other_color = NULL)
```

### Arguments

what	character	The value(s) to highlight
highlight_color	character	The highlight color(s)
other_color	character	The color for the non-highlighted values

### Details

highlight\_color must be of length 1 or the same length as what. If it is of length 1 then all values in what are highlighted with the same color.

If highlight\_color is NULL (the default) then it is set to the default color of the currently active ggcharts theme, i.e. ggcharts\_get\_default\_color(ggcharts\_get\_theme()).

If other\_color is NULL is is automatically determined from the background color of the currently active ggcharts theme.

### Value

An object of class ggcharts\_highlight\_spec

### Author(s)

Thomas Neitmann

**Examples**

```

data("biomedicalrevenue")
revenue2018 <- biomedicalrevenue[biomedicalrevenue$year == 2018, ]

spec <- highlight_spec("Bayer")
bar_chart(revenue2018, company, revenue, highlight = spec)

spec <- highlight_spec("Bayer", "black", "gray")
bar_chart(revenue2018, company, revenue, highlight = spec)

spec <- highlight_spec(c("Bayer", "Novartis"))
bar_chart(revenue2018, company, revenue, highlight = spec)

spec <- highlight_spec(c("Bayer", "AstraZeneca"), c("darkgreen", "darkorange"))
bar_chart(revenue2018, company, revenue, highlight = spec)

ggcharts_set_theme("theme_ng")
spec <- highlight_spec("Novartis")
lollipop_chart(revenue2018, company, revenue, highlight = spec)

```

line\_chart

*Line Chart***Description**

Easily create a line chart

**Usage**

```
line_chart(data, x, y, group, line_color = "auto", line_size = 1)
```

**Arguments**

data	Dataset used for the line chart
x	numeric column of data
y	numeric column of data or a tdiyselect expression
group	character or factor column of data defining individual lines
line_color	character. Line color
line_size	character Line width in mm

**Details**

For plotting multiple lines `line_chart()` can handle data in long or wide format. If the data is in long format pass the variable that identifies individual lines to the `group` argument. If the data is in wide format pass a selection of variables to the `y` argument.

**Value**

An object of class ggplot

**Author(s)**

Thomas Neitmann

**Examples**

```
library(dplyr, warn.conflicts = FALSE)
data("biomedicalrevenue")
data("revenue_wide")

line_chart(revenue_wide, year, Roche)

line_chart(revenue_wide, year, Roche, line_size = 1.5)

line_chart(revenue_wide, year, Roche, line_color = "darkorange")

## Plot multiple lines (data is in long format)
biomedicalrevenue %>%
  filter(company %in% c("Roche", "Novartis", "Bayer")) %>%
  line_chart(year, revenue, group = company)

## Plot multiple lines (data in wide format, i.e. one column per line)
## Select multiple columns with `c()`
line_chart(revenue_wide, year, c(Roche, Novartis, Bayer))

## Select all columns from Novartis to Sanofi suing `:`
line_chart(revenue_wide, year, Novartis:Sanofi)

## Select all columns starting with "B"
line_chart(revenue_wide, year, starts_with("B"))
```

---

lollipop\_chart

*Lollipop Chart*

---

**Description**

Easily create a lollipop chart

**Usage**

```
lollipop_chart(
  data,
  x,
  y,
  facet = NULL,
```

```

    ...,
    line_size = 0.75,
    line_color = "auto",
    point_size = 4,
    point_color = line_color,
    highlight = NULL,
    sort = TRUE,
    horizontal = TRUE,
    top_n = NULL,
    threshold = NULL,
    other = FALSE,
    limit = NULL
  )

```

### Arguments

<code>data</code>	Dataset to use for the bar chart
<code>x</code>	character or factor column of data
<code>y</code>	numeric column of data representing the lollipop length. If missing, the lollipop length will be proportional to the count of each value in <code>x</code> .
<code>facet</code>	character or factor column of data defining the faceting groups
<code>...</code>	Additional arguments passed to <code>aes()</code>
<code>line_size</code>	numeric. Size of the lollipop 'stick'
<code>line_color</code>	character. Color of the lollipop 'stick'
<code>point_size</code>	numeric. Size of the lollipop 'head'
<code>point_color</code>	character. Color of the lollipop 'head'
<code>highlight</code>	character. One or more value(s) of <code>x</code> that should be highlighted in the plot
<code>sort</code>	logical. Should the data be sorted before plotting?
<code>horizontal</code>	logical. Should the plot be oriented horizontally?
<code>top_n</code>	numeric. If a value for <code>top_n</code> is provided only the top <code>top_n</code> records will be displayed
<code>threshold</code>	numeric. If a value for <code>threshold</code> is provided only records with <code>y &gt; threshold</code> will be displayed
<code>other</code>	logical. Should all <code>x</code> with <code>y &lt; threshold</code> be summarized in a group called 'other' and be displayed at the bottom of the chart?
<code>limit</code>	Deprecated. use <code>top_n</code> instead.

### Details

Both `top_n` and `threshold` only work when `sort = TRUE`. Attempting to use them when `sort = FALSE` will result in an error. Furthermore, only `top_n` or `threshold` can be used at a time. Providing a value for both `top_n` and `threshold` will result in an error as well.

### Value

An object of class `ggplot`

**Author(s)**

Thomas Neitmann

**See Also**

For more details have a look at these vignettes: `vignette("highlight", package = "ggcharts")`  
`vignette("customize", package = "ggcharts")`

**Examples**

```
data(biomedicalrevenue)
revenue2016 <- biomedicalrevenue[biomedicalrevenue$year == 2016, ]
revenue_bayer <- biomedicalrevenue[biomedicalrevenue$company == "Bayer", ]

## By default lollipop_chart() creates a horizontal and sorted plot
lollipop_chart(revenue2016, company, revenue)

## If the `y` argument is missing the count of each value in `x` is displayed
lollipop_chart(mtcars, cyl)

## Create a vertical, non-sorted lollipop chart
lollipop_chart(revenue_bayer, year, revenue, horizontal = FALSE, sort = FALSE)

## Limit the number of lollipops to the top 15
lollipop_chart(revenue2016, company, revenue, top_n = 15)

## Display only companies with revenue > 50B.
lollipop_chart(revenue2016, company, revenue, threshold = 50)

## Change the color of the whole lollipop
lollipop_chart(revenue2016, company, revenue, line_color = "purple")

## Change the color of the lollipop stick and head individually
lollipop_chart(revenue2016, company, revenue, point_color = "darkgreen", line_color = "gray")

## Decrease the lollipop head size
lollipop_chart(revenue2016, company, revenue, point_size = 2.5)

## Highlight a single lollipop
lollipop_chart(revenue2016, company, revenue, top_n = 15, highlight = "Roche")

## Use facets to show the top 10 companies over the years
lollipop_chart(biomedicalrevenue, company, revenue, facet = year, top_n = 10)
```



**Description**

Swiss population in 2020 by five-year age groups

**Usage**

popch

**Format**

A data frame with 42 rows and 3 variables:

**age** Five-year age group

**sex** Sex

**pop** Population

**Source**

US Census International Data Base

---

pop europe

*European Population*

---

**Description**

Population of European countries in 1952 and 2007

**Usage**

pop europe

**Format**

A data frame with 30 rows and 3 variables:

**country** Name of the country

**pop1952** Population in 1952 (in millions)

**pop2007** Population in 2007 (in millions)

**Source**

<http://www.gapminder.org/data/>

---

pyramid_chart	<i>Pyramid Chart</i>
---------------	----------------------

---

**Description**

Easily create a pyramid chart

**Usage**

```
pyramid_chart(  
  data,  
  x,  
  y,  
  group,  
  bar_colors = c("#1F77B4", "#FF7F0E"),  
  sort = "no",  
  xlab = NULL,  
  title = NULL  
)
```

**Arguments**

data	Dataset to use for the pyramid chart
x	character or factor column of data
y	numeric column of data
group	character or factor column of data
bar_colors	character vector of length 2 containing colors
sort	character. Should the bars be sorted? By default "no".
xlab	character. X axis label
title	character. Plot title. By default no title is displayed.

**Value**

An object of class `ggplot`

**Author(s)**

Thomas Neitmann

**Examples**

```
data(popch)  
  
pyramid_chart(popch, age, pop, sex)  
  
## Change bar colors
```

```

pyramid_chart(popch, age, pop, sex, bar_colors = c("darkgreen", "darkorange"))

## Change x axis label and add title
pyramid_chart(popch, age, pop, sex, xlab = "Population", title = "Switzerland 2020")

```

---

revenue\_wide

*Top Biomedical Companies Revenues*


---

### Description

Annual revenues of top biomedical companies from 2011 to 2018 in wide format

### Usage

```
revenue_wide
```

### Format

A data frame with 8 rows and 29 variables:

**year** Fiscal year

**Johnson & Johnson** Revenue of Johnson & Johnson in billion USD

**Roche** Revenue of Roche in billion USD

**Pfizer** Revenue of Pfizer in billion USD

**Novartis** Revenue of Novartis in billion USD

**Bayer** Revenue of Bayer in billion USD

**GlaxoSmithKline** Revenue of GlaxoSmithKline in billion USD

**Merck & Co.** Revenue of Merck & Co. in billion USD

**Sanofi** Revenue of Sanofi in billion USD

**AbbVie** Revenue of AbbVie in billion USD

**Abbott Laboratories** Revenue of Abbott Laboratories in billion USD

**Eli Lilly & Co** Revenue of Eli Lilly & Co in billion USD

**Amgen** Revenue of Amgen in billion USD

**Bristol-Myers Squibb** Revenue of Bristol-Myers Squibb in billion USD

**Gilead Sciences** Revenue of Gilead Sciences in billion USD

**AstraZeneca** Revenue of AstraZeneca in billion USD

**/ Teva Pharmaceutical Industries** Revenue of / Teva Pharmaceutical Industries in billion USD

**Boehringer Ingelheim** Revenue of Boehringer Ingelheim in billion USD

**Takeda Pharmaceutical** Revenue of Takeda Pharmaceutical in billion USD

**Merck Group** Revenue of Merck Group in billion USD

**Novo Nordisk** Revenue of Novo Nordisk in billion USD

**Allergan plc** Revenue of Allergan plc in billion USD  
**Danaher Corporation** Revenue of Danaher Corporation in billion USD  
**Celgene** Revenue of Celgene in billion USD  
**Biogen** Revenue of Biogen in billion USD  
**Astellas Pharma** Revenue of Astellas Pharma in billion USD  
**Labcorp** Revenue of Labcorp in billion USD  
**Baxter International** Revenue of Baxter International in billion USD  
**Mylan** Revenue of Mylan in billion USD

### Source

[https://en.wikipedia.org/wiki/List\\_of\\_largest\\_biomedical\\_companies\\_by\\_revenue](https://en.wikipedia.org/wiki/List_of_largest_biomedical_companies_by_revenue)

---

theme\_coffee

*Theme Coffee*

---

### Description

A theme inspired by coffee

### Usage

```
theme_coffee(
  base_size = 13,
  base_family = "Cooper Hewitt",
  header_family = "Cooper Hewitt",
  axis = "",
  ticks = "",
  grid = ""
)
```

### Arguments

base_size	numeric. Base font size in pt
base_family	character. Base font family
header_family	character. Font family for title and subtitle
axis	character. Where to draw an axis line
ticks	character. Where to draw axis ticks
grid	character. Where to draw grid lines

### Value

An object of class theme

**Author(s)**

Thomas Neitmann

**See Also**

For more details see the 'theme' vignette: `vignette("theme", package = "ggcharts")`

**Examples**

```
library(ggplot2)
library(dplyr)

scatter <- ggplot(mtcars, aes(hp, mpg)) +
  geom_point(color = "#F4C95D")

scatter + theme_coffee()

scatter + theme_coffee(grid = "XY")

scatter + theme_coffee(axis = "xy", ticks = "xy")

bar_chart(ggplot2::diamonds, cut, bar_color = "#F4C95D") +
  theme_coffee(axis = "y", grid = "Y")

column_chart(ggplot2::diamonds, cut, bar_color = "#F4C95D") +
  theme_coffee(axis = "x", grid = "X")

ggcharts::biomedicalrevenue %>%
  filter(company == "Roche") %>%
  ggplot(aes(year, revenue)) +
  geom_line(color = "#F4C95D", size = 1) +
  scale_y_continuous(expand = expand_scale(c(0, .05))) +
  theme_coffee(grid = "X", axis = "x", ticks = "x")
```

---

theme\_ggcharts

*Theme ggcharts*

---

**Description**

The default ggcharts theme

**Usage**

```
theme_ggcharts(
  base_size = 13,
  base_family = "Cooper Hewitt",
  header_family = "Cooper Hewitt",
  axis = "",
```

```

    ticks = "",
    grid = ""
  )

```

### Arguments

base_size	numeric. Base font size in pt
base_family	character. Base font family
header_family	character. Font family for title and subtitle
axis	character. Where to draw an axis line
ticks	character. Where to draw axis ticks
grid	character. Where to draw grid lines

### Details

theme\_ggcharts is the default theme used when creating any plot with ggcharts.

### Value

An object of class theme

### Author(s)

Thomas Neitmann

### See Also

For more details see the 'theme' vignette: vignette("theme", package = "ggcharts")

### Examples

```

library(ggplot2)
library(dplyr)

scatter <- ggplot(mtcars, aes(hp, mpg)) +
  geom_point(color = "steelblue")

scatter + theme_ggcharts()

scatter + theme_ggcharts(grid = "XY")

scatter + theme_ggcharts(axis = "xy", ticks = "xy")

bar_chart(ggplot2::diamonds, cut) +
  theme_ggcharts(axis = "y", grid = "Y")

column_chart(ggplot2::diamonds, cut) +
  theme_ggcharts(axis = "x", grid = "X")

ggcharts::biomedicalrevenue %>%

```

```
filter(company == "Roche") %>%
  ggplot(aes(year, revenue)) +
  geom_line(color = "steelblue", size = 1) +
  scale_y_continuous(expand = expand_scale(c(0, .05))) +
  theme_ggcharts(grid = "X", axis = "x", ticks = "x")
```

---

theme\_hermit

*Theme Hermit*

---

## Description

A ggplot2 theme inspired by the 'hermit' Hugo theme

## Usage

```
theme_hermit(  
  base_size = 13,  
  base_family = "Cooper Hewitt",  
  header_family = "Cooper Hewitt",  
  axis = "",  
  ticks = "",  
  grid = ""  
)
```

## Arguments

base_size	numeric. Base font size in pt
base_family	character. Base font family
header_family	character. Font family for title and subtitle
axis	character. Where to draw an axis line
ticks	character. Where to draw axis ticks
grid	character. Where to draw grid lines

## Value

An object of class theme

## Author(s)

Thomas Neitmann

## See Also

For more details see the 'theme' vignette: `vignette("theme", package = "ggcharts")`

**Examples**

```

library(ggplot2)
library(dplyr)

scatter <- ggplot(mtcars, aes(hp, mpg)) +
  geom_point(color = "yellow")

scatter + theme_hermit()

scatter + theme_hermit(grid = "XY")

scatter + theme_hermit(axis = "xy", ticks = "xy")

bar_chart(ggplot2::diamonds, cut, bar_color = "darkorange") +
  theme_hermit(axis = "y", grid = "Y")

column_chart(ggplot2::diamonds, cut, bar_color = "darkorange") +
  theme_hermit(axis = "x", grid = "X")

ggcharts::biomedicalrevenue %>%
  filter(company == "Roche") %>%
  ggplot(aes(year, revenue)) +
  geom_line(color = "yellow", size = 1) +
  scale_y_continuous(expand = expand_scale(c(0, .05))) +
  theme_hermit(grid = "X", axis = "x", ticks = "x")

```

---

 theme\_ng

*Theme NG*


---

**Description**

A ggplot2 theme inspired with the 'hello friend ng' Hugo theme

**Usage**

```

theme_ng(
  base_size = 13,
  base_family = "Cooper Hewitt",
  header_family = "Cooper Hewitt",
  axis = "",
  ticks = "",
  grid = ""
)

```

**Arguments**

base\_size      numeric. Base font size in pt



base_family	character. Base font family
header_family	character. Font family for title and subtitle
axis	character. Where to draw an axis line
ticks	character. Where to draw axis ticks
grid	character. Where to draw grid lines

**Value**

An object of class theme

**Author(s)**

Thomas Neitmann

**See Also**

For more details see the 'theme' vignette: `vignette("theme", package = "ggcharts")`

**Examples**

```
library(ggplot2)
library(dplyr)

scatter <- ggplot(mtcars, aes(hp, mpg)) +
  geom_point(color = "yellow")

scatter + theme_ng()

scatter + theme_ng(grid = "XY")

scatter + theme_ng(axis = "xy", ticks = "xy")

bar_chart(ggplot2::diamonds, cut, bar_color = "darkorange") +
  theme_ng(axis = "y", grid = "Y")

column_chart(ggplot2::diamonds, cut, bar_color = "darkorange") +
  theme_ng(axis = "x", grid = "X")

ggcharts::biomedicalrevenue %>%
  filter(company == "Roche") %>%
  ggplot(aes(year, revenue)) +
  geom_line(color = "yellow", size = 1) +
  scale_y_continuous(expand = expand_scale(c(0, .05))) +
  theme_ng(grid = "X", axis = "x", ticks = "x")
```

---

theme_nightblue	<i>Theme Nightblue</i>
-----------------	------------------------

---

## Description

A theme inspired by the RStudio 'Tomorrow Night Blue' editor theme

## Usage

```
theme_nightblue(  
  base_size = 13,  
  base_family = "Cooper Hewitt",  
  header_family = "Cooper Hewitt",  
  axis = "",  
  ticks = "",  
  grid = ""  
)
```

## Arguments

base_size	numeric. Base font size in pt
base_family	character. Base font family
header_family	character. Font family for title and subtitle
axis	character. Where to draw an axis line
ticks	character. Where to draw axis ticks
grid	character. Where to draw grid lines

## Value

An object of class theme

## Author(s)

Thomas Neitmann

## See Also

For more details see the 'theme' vignette: `vignette("theme", package = "ggcharts")`

## Examples

```
library(ggplot2)  
library(dplyr)  
  
scatter <- ggplot(mtcars, aes(hp, mpg)) +  
  geom_point(color = "#EBBBFF")
```

```
scatter + theme_nightblue()

scatter + theme_nightblue(grid = "XY")

scatter + theme_nightblue(axis = "xy", ticks = "xy")

bar_chart(ggplot2::diamonds, cut, bar_color = "darkorange") +
  theme_nightblue(axis = "y", grid = "Y")

column_chart(ggplot2::diamonds, cut, bar_color = "darkorange") +
  theme_nightblue(axis = "x", grid = "X")

ggcharts::biomedicalrevenue %>%
  filter(company == "Roche") %>%
  ggplot(aes(year, revenue)) +
  geom_line(color = "yellow", size = 1) +
  scale_y_continuous(expand = expand_scale(c(0, .05))) +
  theme_nightblue(grid = "X", axis = "x", ticks = "x")
```

# Index

## \* datasets

- biomedicalrevenue, 5
- popch, 16
- pop europe, 17
- revenue\_wide, 19

bar\_chart, 2

biomedicalrevenue, 5

column\_chart (bar\_chart), 2

diverging\_bar\_chart, 5

diverging\_lollipop\_chart, 7

dumbbell\_chart, 8

ggcharts\_get\_default\_color, 10

ggcharts\_get\_theme, 11

ggcharts\_set\_theme  
(ggcharts\_get\_theme), 11

highlight\_spec, 12

line\_chart, 13

lollipop\_chart, 14

popch, 16

pop europe, 17

pyramid\_chart, 18

revenue\_wide, 19

theme\_coffee, 20

theme\_ggcharts, 21

theme\_hermit, 23

theme\_ng, 24

theme\_nightblue, 26