Package ‘ggpath’

September 6, 2022

Title Robust Image Rendering Support for 'ggplot2'

Version 1.0.0

Description A 'ggplot2' extension that enables robust image grobs in panels and theme elements.

License MIT + file LICENSE

URL `https://github.com/mrcaseb/ggpath`, `https://mrcaseb.github.io/ggpath/`

BugReports `https://github.com/mrcaseb/ggpath/issues`

Depends R (>= 3.5.0)

Imports cachem (>= 1.0.0), cli (>= 3.0.0), ggplot2 (>= 3.3.0), grid, magick (>= 2.7.3), memoise (>= 2.0.0), rappdirs (>= 0.3.0), rlang (>= 0.4.11)

Suggests covr (>= 3.5.1), rsvg (>= 2.0), testthat (>= 3.0.0), vdiffr (>= 1.0.2)

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.2.1

NeedsCompilation no

Author Sebastian Carl [aut, cre, cph]

Maintainer Sebastian Carl <`mrcaseb@gmail.com`>

Repository CRAN

Date/Publication 2022-09-06 06:10:02 UTC

R topics documented:

element_path 2
geom_from_path 3
geom_lines 6

Index **9**

element_path *Theme Element for Image Grobs*

Description

In conjunction with the ggplot2::theme system, the function `element_path()` enables images in non-data components of the plot, e.g. axis text. It draws images from valid image URLs, raster objects, or bitmap arrays.

Usage

```
element_path(
  alpha = NULL,
  colour = NA,
  hjust = NULL,
  vjust = NULL,
  color = NULL,
  angle = NULL,
  size = 0.5
)
```

Arguments

`alpha`	The alpha channel, i.e. transparency level, as a numerical value between 0 and 1.
`colour, color`	The image will be colorized with this color. Use the special character `"b/w"` to set it to black and white. For more information on valid color names in ggplot2 see `https://ggplot2.tidyverse.org/articles/ggplot2-specs.html?q=colour#colour-and-fill`.
`hjust, vjust`	The horizontal and vertical adjustment respectively. Must be a numerical value between 0 and 1.
`angle`	The angle of the element as a numerical value between 0° and 360°.
`size`	The output grob size in `cm` (!).

Value

An S3 object of class `element`.

See Also

`geom_from_path()` for more information.

Examples

```
library(ggplot2)
library(ggpath)

# compute path of an R logo file shipped with ggpath
local_image_path <- system.file("r_logo.svg", package = "ggpath")

# create dataframe with x-y-coordinates and the above local path
plot_data <- data.frame(x = c(-1, 1), y = 1, path = local_image_path)

# Replace title, subtitle, the caption, axis labels as well as y-axis text
# the the local image
ggplot(plot_data, aes(x = x, y = local_image_path)) +
  theme_minimal() +
  labs(
    title = local_image_path,
    subtitle = local_image_path,
    x = local_image_path,
    y = local_image_path,
    caption = local_image_path
  ) +
  theme(
    plot.caption = element_path(hjust = 1, size = 0.6),
    axis.text.y = element_path(size = 1),
    axis.title.x = element_path(),
    axis.title.y = element_path(vjust = 0.9),
    plot.title = element_path(hjust = 0, size = 2, alpha = 0.5),
    plot.subtitle = element_path(hjust = 0.9, angle = 45),
  )
```

`geom_from_path` *ggplot2 Layer for Visualizing Images from URLs or Local Paths*

Description

This geom is used to plot images instead of points in a ggplot. It requires x, y aesthetics as well as a path.

Usage

```
geom_from_path(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  ...,
  na.rm = FALSE,
  show.legend = FALSE,
  inherit.aes = TRUE
)
```

Arguments

`mapping` Set of aesthetic mappings created by `aes()` or `aes_()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

`data` The data to be displayed in this layer. There are three options:

If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.

A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.

A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).

`stat` The statistical transformation to use on the data for this layer, as a string.

`position` Position adjustment, either as a string, or the result of a call to a position adjustment function.

`...` Other arguments passed on to `ggplot2::layer()`. These are often aesthetics, used to set an aesthetic to a fixed value. See the below section "Aesthetics" for a full list of possible arguments.

`na.rm` If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.

`inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

Value

A ggplot2 layer (`ggplot2::layer()`) that can be added to a plot created with `ggplot2::ggplot()`.

Aesthetics

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

- **x** - The x-coordinate.
- **y** - The y-coordinate.
- **path** - a file path, url, raster object or bitmap array. See `magick::image_read()` for further information.
- `alpha = NULL` - The alpha channel, i.e. transparency level, as a numerical value between 0 and 1.
- `colour = NULL` - The image will be colorized with this colour. Use the special character `"b/w"` to set it to black and white. For more information on valid colour names in ggplot2 see `https://ggplot2.tidyverse.org/articles/ggplot2-specs.html?q=colour#colour-and-fill`

- angle = 0 - The angle of the image as a numerical value between 0° and 360°.
- hjust = 0.5 - The horizontal adjustment relative to the given x coordinate. Must be a numerical value between 0 and 1.
- vjust = 0.5 - The vertical adjustment relative to the given y coordinate. Must be a numerical value between 0 and 1.
- width = 1.0 - The desired width of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is width = 0.1 (see below examples).
- height = 1.0 - The desired height of the image in npc (Normalised Parent Coordinates). The default value is set to 1.0 which is *big* but it is necessary because all used values are computed relative to the default. A typical size is height = 0.1 (see below examples).

Examples

```
library(ggplot2)
library(ggpath)

# compute path of an R logo file shipped with ggpath
local_image_path <- system.file("r_logo.png", package = "ggpath")

# create dataframe with x-y-coordinates and the above local path
plot_data <- data.frame(x = c(-1, 1), y = 1, path = local_image_path)

# plot images directly from local path
ggplot(plot_data, aes(x = x, y = y)) +
  geom_from_path(aes(path = path), width = 0.2) +
  coord_cartesian(xlim = c(-2, 2)) +
  theme_minimal()

# plot images directly from local path and apply transparency
ggplot(plot_data, aes(x = x, y = y)) +
  geom_from_path(aes(path = path), width = 0.2, alpha = 0.5) +
  coord_cartesian(xlim = c(-2, 2)) +
  theme_minimal()

# It is also possible and recommended to use the underlying Geom inside a
# ggplot2 annotation
ggplot() +
  annotate(
    ggpath::GeomFromPath,
    x = 0,
    y = 0,
    path = local_image_path,
    width = 0.4
  ) +
  theme_minimal()
```

geom_lines *ggplot2 Layer for Horizontal and Vertical Reference Lines*

Description

These geoms can be used to draw horizontal or vertical reference lines in a ggplot. They use the data in the aesthetics `x0` and `y0` to compute their `median` or `mean` and draw them as a line.

Usage

```
geom_median_lines(
  mapping = NULL,
  data = NULL,
  ...,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)

geom_mean_lines(
  mapping = NULL,
  data = NULL,
  ...,
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE
)
```

Arguments

`mapping`	Set of aesthetic mappings created by `aes()` or `aes_()`.
`data`	The data to be displayed in this layer. There are three options: If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`. A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created. A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).
`...`	Other arguments passed on to `layer()`. These are often aesthetics, used to set an aesthetic to a fixed value, like `colour = "red"` or `size = 3`. They may also be parameters to the paired geom/stat.
`na.rm`	If `FALSE`, the default, missing values are removed with a warning. If `TRUE`, missing values are silently removed.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.

`inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn’t inherit behavior from the default plot specification.

Value

A ggplot2 layer (`ggplot2::layer()`) that can be added to a plot created with `ggplot2::ggplot()`.

Aesthetics

`geom_median_lines()` and `geom_mean_lines()` understand the following aesthetics (at least one of the bold aesthetics is required):

- **x0** - The variable for which to compute the median/mean that is drawn as vertical line.
- **y0** - The variable for which to compute the median/mean that is drawn as horizontal line.
- `alpha` = `NA` - The alpha channel, i.e. transparency level, as a numerical value between 0 and 1.
- `color` = `"red"` - The color of the drawn lines.
- `linetype` = `2` - The linetype of the drawn lines.
- `size` = `0.5` - The size of the drawn lines.

See Also

The underlying ggplot2 geoms `geom_hline()` and `geom_vline()`

Examples

```
library(ggplot2)

# inherit top level aesthetics
ggplot(mtcars, aes(x = disp, y = mpg, y0 = mpg, x0 = disp)) +
  geom_point() +
  geom_median_lines() +
  geom_mean_lines(color = "blue") +
  theme_minimal()

# draw horizontal line only
ggplot(mtcars, aes(x = disp, y = mpg, y0 = mpg)) +
  geom_point() +
  geom_median_lines() +
  geom_mean_lines(color = "blue") +
  theme_minimal()

# draw vertical line only
ggplot(mtcars, aes(x = disp, y = mpg, x0 = disp)) +
  geom_point() +
  geom_median_lines() +
  geom_mean_lines(color = "blue") +
  theme_minimal()

# choose your own value
ggplot(mtcars, aes(x = disp, y = mpg)) +
  geom_point() +
  geom_median_lines(x0 = 400, y0 = 15) +
  geom_mean_lines(x0 = 150, y0 = 30, color = "blue") +
  theme_minimal()
```

Index

aes(), *4*, *6*
aes_(), *4*, *6*

borders(), *4*

element_path, 2

fortify(), *4*, *6*

geom_from_path, 3
geom_from_path(), *2*
geom_hline(), *7*
geom_lines, 6
geom_mean_lines *(geom_lines)*, 6
geom_median_lines *(geom_lines)*, 6
geom_vline(), *7*
ggplot(), *4*, *6*
ggplot2::ggplot(), *4*, *7*
ggplot2::layer(), *4*, *7*
ggplot2::theme, *2*

layer(), *6*

magick::image_read(), *4*