Package: ggsvg (via r-universe)

September 11, 2024

Type Package

Title SVG Glyphs for Ggplot

Version 0.1.13

Maintainer Mike Cheng <mikefc@coolbutuseless.com>

Description Use SVG graphics as glyphs when plotting points with ggplot2.

License MIT + file LICENSE

URL https://github.com/coolbutuseless/ggsvg

BugReports https://github.com/coolbutuseless/ggsvg/issues

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.3.2

Imports grid, ggplot2, scales, glue, stringr, rlang, rsvg (>= 2.3.0)

Suggests knitr, rmarkdown, ambient, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

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

RemoteUrl https://github.com/coolbutuseless/ggsvg

RemoteRef HEAD

RemoteSha 340fd565fa9d6d832f44e8a07dc5ceed04cec5c5

Contents

`add_suffix` *Recursively update all names in a grobTree by adding a suffix to the name*

Description

If two grobs in a plot have the same name, then it is not guaranteed that both will be drawn. This function ensures that every grob in a tree has a unique name.

Usage

```
add_suffix(x, suffix)
```

Arguments

`x`	grobTree
`suffix`	string

Value

new grobTree

`create_new_GeomPointSVG` *Create a fresh instance of a GeomPointSVG ggproto object*

Description

These Geoms are created dynamically as each time `geom_point_svg()` is called, it wants to customize the `$default_aes` on this ggproto. Because ggproto objects are environments, then setting a default on the global copy would set the default for all references. This will get messy as things like CSS Aesthetics should not be shared between plots.

Usage

```
create_new_GeomPointSVG()
```

Details

So every geom_point_svg() gets a fresh GeomPointSVG by calling this function.

Value

ggproto object for GeomPointSVG

---

css *Specify an aesthetic that maps to a CSS Selector and Property*

Description

This function should only be used with an `aes()` call in `geom_point_svg()`, or as an argument to a ggsvg scale object e.g. `scale_svg_fill_discrete(aesthetics = css(...))`

Usage

```
css(selector, ..., format = NULL)
```

Arguments

`selector`	CSS selector as a single character string e.g. "circle .big"
`...`	single named argument of the form `css_property = value`. The `value` will remain unevaluated for passing into `ggplot2::aes()`. e.g. `stroke = cyl`, `"stroke-width" = mpg`
`format`	Advanced. This is a string specifying the formatting for the CSS property value. This is almost (but not quite) the equivalent of a formatting string for the `glue` package - however, the delimiters for the string are `[]` rather than `{}`. Default: NULL is equivalent to "[x]" and will insert just the bare value. For example, if the CSS property required an explicit "px" suffix on the value, the format would be "[x]px"

Value

a named amed list (with length = 1) where the name is the full name of this CSS aesthetic, and the value is the unevaluated value passed in to the ...

Examples

```
## Not run:
# circle .bit { stroke = XX; }
css("circle .big", stroke = as.factor(cyl))

# circle .bit { stroke = XXpx; }
css("circle .big", stroke = as.factor(cyl), format = "[x]px")

## End(Not run)
```

`draw_key_PointSVG` *Key for SVG points*

Description

Key for SVG points

Usage

```
draw_key_PointSVG(data, params, size)
```

Arguments

`data`, `params`, `size`
key stuff

`geom_point_svg` *Use SVG images as glyphs for points*

Description

Aesthetics

- **svg** SVG as a character string
- **svg_width,svg_height** Specify rendered width and/or height. If only one of these values is specified, then the other will be scaled to keep he aspect ratio. If neither value is specified (the default) then dimensions will be taken from the SVG itself. This value could be used to increase the resolution of the SVG so it does not appear blurry once rendered to an element in the plot e.g. `svg_width = 1000`
- **hjust,vjust** The justification of the SVG's bounding rectangle relative to its (x,y) position. Default value of 0.5 mean to centre the SVG at the specified location. Standard values for these variables are in the range [0,1]
- **x_abs,y_abs** Absolute positioning within the panel. Default: NULL means that the standard x,y positioning is to be used. Standard values for these variables are in the range [0,1]

Usage

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

Arguments

mapping, data, stat, position, ..., na.rm, show.legend, inherit.aes
see documentation for ggplot2::geom_point()

defaults
Advanced option. A named list of default values for new aesthetics. In general this is not necessary when using css() aesthetics, as a default value will be determined based upon the CSS property e.g. stroke property will have a default value of "black"

Set ‘options(GGSVG_DEBUG = TRUE)‘ for some verbose debugging which will cause ggsvg to output (to the console) the final SVG for each and every element in the plot.

Value

ggplot2 layer object

ggplot_add.ScaleSVGDefault

S3 method

Description

S3 method

Usage

```
## S3 method for class 'ScaleSVGDefault'
ggplot_add(object, plot, object_name)
```

Arguments

object, plot, object_name
see ggplot2 docs

scale_ggsvg_default *Automatically add a default scale for SVG CSS aesthetics*

Description

Automatically add a default scale for SVG CSS aesthetics

Usage

```
scale_ggsvg_default(p, verbose = FALSE)
```

Arguments

p	ggplot2 object
verbose	Be verbose about what scales are being created? Logical. Default: FALSE

Value

ggplot2 object with default scale_svg_*() added to the plot environment so that ggplot2 will find these scales.

Technical Details

If a scale for a particular aesthetic is not explicitly given, then ggplot2 guesses which scales to use based upon two things:

- The aesthetic being mapped e.g. fill
- The type of data that is being mapped e.g. continuous, discrete, date etc

For regular ggplot2 plots, the combination of all these factors have scales included in the ggplot2 package e.g. scale_fill_continous, scale_linetype_discrete.

However, for ggsvg, using css() aesthetics means that there are a very large number of aesthetics being mapped to i.e not just fill and colour, but also stroke-width, angle and many other CSS parameters.

Furthermore, CSS selectors are used to target the aesthetic, and since there are an infinite number of valid CSS selectors, there will be an infinite number of scales needed to be provided by ggsvg.

Fortunately, the majority of things we want to map to in SVG are either colours or numbers, which means that we can create a single colour scale for use with ggsvg and just change its name, then add it to the plot environment so that ggplot2 can find it when it renders the plot.

scale_svg_alpha *Scales for working with arbitrary named aesthetics*

Description

See the corresponding `scale_*()` function in `ggplot2` for more information on how scales operate.

Usage

```
scale_svg_alpha(aesthetics, ..., range = c(0.1, 1))

scale_svg_alpha_binned(aesthetics, ..., range = c(0.1, 1))

scale_svg_alpha_continuous(aesthetics, ..., range = c(0.1, 1))

scale_svg_alpha_discrete(aesthetics, ...)

scale_svg_alpha_identity(aesthetics, ..., guide = "none")

scale_svg_alpha_manual(
  aesthetics,
  ...,
  values,
  breaks = waiver(),
  na.value = NA
)

scale_svg_alpha_ordinal(aesthetics, ..., range = c(0.1, 1))

scale_svg_colour_binned(
  aesthetics,
  ...,
  type = getOption("ggplot2.binned.colour")
)

scale_svg_colour_brewer(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = 1
)

scale_svg_colour_continuous(
  aesthetics,
  ...,
  type = getOption("ggplot2.continuous.colour")
```

```
)

scale_svg_colour_discrete(
  aesthetics,
  ...,
  type = getOption("ggplot2.discrete.colour")
)

scale_svg_colour_fermenter(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = -1,
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_colour_grey(
  aesthetics,
  ...,
  start = 0.2,
  end = 0.8,
  na.value = "red"
)

scale_svg_colour_hue(
  aesthetics,
  ...,
  h = c(0, 360) + 15,
  c = 100,
  l = 65,
  h.start = 0,
  direction = 1,
  na.value = "grey50"
)

scale_svg_colour_identity(aesthetics, ..., guide = "none")

scale_svg_colour_manual(
  aesthetics,
  ...,
  values,
  breaks = waiver(),
  na.value = "grey50"
)

scale_svg_colour_ordinal(
  aesthetics,
  ...,
  type = getOption("ggplot2.ordinal.colour", getOption("ggplot2.ordinal.fill"))
)

scale_svg_colour_qualitative(
  aesthetics,
  ...,
  type = NULL,
  h = c(0, 360) + 15,
  c = 100,
  l = 65,
  h.start = 0,
  direction = 1,
  na.value = "grey50"
)

scale_svg_colour_steps(
  aesthetics,
  ...,
  low = "#132B43",
  high = "#56B1F7",
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_colour_steps2(
  aesthetics,
  ...,
  low = muted("red"),
  mid = "white",
  high = muted("blue"),
  midpoint = 0,
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_colour_stepsn(
  aesthetics,
  ...,
  colours,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps",
  colors
```

```
)

scale_svg_colour_viridis_b(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_colour_viridis_d(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D"
)

scale_svg_continuous_identity(aesthetics, ..., guide = "none")

scale_svg_discrete_identity(aesthetics, ..., guide = "none")

scale_svg_discrete_manual(aesthetics, ..., values, breaks = waiver())

scale_svg_fill_binned(aesthetics, ..., type = getOption("ggplot2.binned.fill"))

scale_svg_fill_brewer(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = 1
)

scale_svg_fill_continuous(
  aesthetics,
  ...,
  type = getOption("ggplot2.continuous.fill")
)
```

```
scale_svg_fill_discrete(
  aesthetics,
  ...,
  type = getOption("ggplot2.discrete.fill")
)

scale_svg_fill_fermenter(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = -1,
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_fill_grey(aesthetics, ..., start = 0.2, end = 0.8, na.value = "red")

scale_svg_fill_hue(
  aesthetics,
  ...,
  h = c(0, 360) + 15,
  c = 100,
  l = 65,
  h.start = 0,
  direction = 1,
  na.value = "grey50"
)

scale_svg_fill_identity(aesthetics, ..., guide = "none")

scale_svg_fill_manual(
  aesthetics,
  ...,
  values,
  breaks = waiver(),
  na.value = "grey50"
)

scale_svg_fill_ordinal(
  aesthetics,
  ...,
  type = getOption("ggplot2.ordinal.fill", getOption("ggplot2.ordinal.colour"))
)

scale_svg_fill_qualitative(
  aesthetics,
  ...,
  type = NULL,
  h = c(0, 360) + 15,
  c = 100,
  l = 65,
  h.start = 0,
  direction = 1,
  na.value = "grey50"
)

scale_svg_fill_steps(
  aesthetics,
  ...,
  low = "#132B43",
  high = "#56B1F7",
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_fill_steps2(
  aesthetics,
  ...,
  low = muted("red"),
  mid = "white",
  high = muted("blue"),
  midpoint = 0,
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_fill_stepsn(
  aesthetics,
  ...,
  colours,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps",
  colors
)

scale_svg_fill_viridis_b(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = "coloursteps"
)

scale_svg_fill_viridis_d(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D"
)

scale_svg_linetype(aesthetics, ..., na.value = "blank")

scale_svg_linetype_binned(aesthetics, ..., na.value = "blank")

scale_svg_linetype_continuous(aesthetics, ...)

scale_svg_linetype_discrete(aesthetics, ..., na.value = "blank")

scale_svg_linetype_identity(aesthetics, ..., guide = "none")

scale_svg_linetype_manual(
  aesthetics,
  ...,
  values,
  breaks = waiver(),
  na.value = "blank"
)

scale_svg_radius(
  aesthetics,
  name = waiver(),
  breaks = waiver(),
  labels = waiver(),
  limits = NULL,
  range = c(1, 6),
  trans = "identity",
  guide = "legend"
)

scale_svg_size(
  aesthetics,
  name = waiver(),
  breaks = waiver(),
  labels = waiver(),
  limits = NULL,
  range = c(5, 10),
  trans = "identity",
  guide = "legend"
)

scale_svg_size_area(aesthetics, ..., max_size = 6)

scale_svg_size_binned(
  aesthetics,
  name = waiver(),
  breaks = waiver(),
  labels = waiver(),
  limits = NULL,
  range = c(5, 10),
  n.breaks = NULL,
  nice.breaks = TRUE,
  trans = "identity",
  guide = "bins"
)

scale_svg_size_binned_area(aesthetics, ..., max_size = 6)

scale_svg_size_continuous(
  aesthetics,
  name = waiver(),
  breaks = waiver(),
  labels = waiver(),
  limits = NULL,
  range = c(5, 10),
  trans = "identity",
  guide = "legend"
)

scale_svg_size_discrete(aesthetics, ...)

scale_svg_size_identity(aesthetics, ..., guide = "none")

scale_svg_size_manual(
  aesthetics,
  ...,
  values,
  breaks = waiver(),
  na.value = NA
```

```
)

scale_svg_size_ordinal(aesthetics, ..., range = c(5, 10))
```

Arguments

`aesthetics` names of aesthethics to apply this scale to. e.g. `css("circle .big", fill = mpg)`, `'fill_rect'`

`...`, `range`, `guide`, `values`, `breaks`, `na.value`, `type`, `palette`
: See `ggplot2` documentation

`direction`, `start`, `end`, `h`, `c`, `l`, `h.start`, `low`, `high`, `space`
: See `ggplot2` documentation

`mid`, `midpoint`, `colours`, `colors`, `alpha`, `begin`, `option`, `name`
: See `ggplot2` documentation

`labels`, `limits`, `trans`, `max_size`, `n.breaks`, `nice.breaks`
: See `ggplot2` documentation

scale_svg_colour_gradient

Continuous scales for colour and fill aesthetics for `ggsvg`

Description

All these colour/fill scales use `guide_colourbar()` but by default, this guide will only accept aesthetics of `fill` and `colour`.

All these colour/fill scales use `guide_colourbar()` but by default, this guide will only accept aesthetics of `fill` and `colour`.

Usage

```
scale_svg_colour_gradient(
  aesthetics,
  ...,
  low = "#132B43",
  high = "#56B1F7",
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_colour_distiller(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = -1,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_colour_gradient2(
  aesthetics,
  ...,
  low = muted("red"),
  mid = "white",
  high = muted("blue"),
  midpoint = 0,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_colour_gradientn(
  aesthetics,
  ...,
  colours,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics),
  colors
)

scale_svg_colour_viridis_c(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_distiller(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = -1,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_gradient(
  aesthetics,
  ...,
  low = "#132B43",
  high = "#56B1F7",
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_gradient2(
  aesthetics,
  ...,
  low = muted("red"),
  mid = "white",
  high = muted("blue"),
  midpoint = 0,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_gradientn(
  aesthetics,
  ...,
  colours,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics),
  colors
)

scale_svg_fill_viridis_c(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_colour_gradient(
  aesthetics,
  ...,
  low = "#132B43",
  high = "#56B1F7",
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_colour_distiller(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = -1,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_colour_gradient2(
  aesthetics,
  ...,
  low = muted("red"),
  mid = "white",
  high = muted("blue"),
  midpoint = 0,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_colour_gradientn(
  aesthetics,
  ...,
  colours,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics),
  colors
)

scale_svg_colour_viridis_c(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_distiller(
  aesthetics,
  ...,
  type = "seq",
  palette = 1,
  direction = -1,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_gradient(
  aesthetics,
  ...,
  low = "#132B43",
  high = "#56B1F7",
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_gradient2(
  aesthetics,
  ...,
  low = muted("red"),
  mid = "white",
  high = muted("blue"),
  midpoint = 0,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)

scale_svg_fill_gradientn(
  aesthetics,
  ...,
  colours,
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics),
  colors
)

scale_svg_fill_viridis_c(
  aesthetics,
  ...,
  alpha = 1,
  begin = 0,
  end = 1,
  direction = 1,
  option = "D",
  values = NULL,
  space = "Lab",
  na.value = "grey50",
  guide = ggplot2::guide_colorbar(available_aes = aesthetics)
)
```

Arguments

`aesthetics` name of the aesthetic e.g. `fill_rect`

`...`, `guide`, `low`, `high`, `space`, `na.value`, `type`, `palette`, `direction`, `values`, `mid`, `midpoint`, `colours`, `colors`, `alpha`, `begin`, `end`, `option`
see `ggplot2` documentation

Details

The two key changes to these scales compared to their `ggplot2` originals are:

- Default `guide` argument is now a `guide_colourbar()` object which explicitly supports the current `aesthetics`
- `aesthetics` is now a required argument

The two key changes to these scales compared to their `ggplot2` originals are:

- Default `guide` argument is now a `guide_colourbar()` object which explicitly supports the current `aesthetics`
- `aesthetics` is now a required argument

`scale_svg_default` *Public facing method*

Description

Public facing method

Usage

```
scale_svg_default()
```

`svg_to_rasterGrob` *Convert SVG to a* `grid` `rasterGrob` *object*

Description

Convert SVG to a `grid` `rasterGrob` object

Usage

```
svg_to_rasterGrob(svg_text, width = NULL, height = NULL, css = NULL, ...)
```

Arguments

`svg_text`	character string containing valid SVG
`width, height`	output width,height in pixels or `NULL` (the default) which inputes the size from the SVG
`css`	character string containing CSS text. This requires your system has a recent version of librsvg.
`...`	other arguments passed to ‘grid::rasterGrob()‘.

Value

`grid::rasterGrob()` object containing the `nativeRaster` returned from `rsvg`

Index