Cheatography

Data Visualization in R: ggvis Cheat Sheet

by shanly3011 via cheatography.com/20988/cs/3865/

ggvis Grammar

The 4 essential components are -

1. Data
2. Coordinate system
3. Marks
4. Properties

Syntax Example-

```
faithful %>%
ggvis(waiting,eruptions) %>% layer_points() %>%
add_axis( " x", title = " Waiting period ", values = c(1,2,3,4,5,6,7), subdivide = 9)
```

We use the piping operator '%>%' for our syntaxes.

Mapping Vs Setting propeties

Mapping	Setting
= maps property to a data value	:= sets property to a specific size/coluor/width
Used for visualization of data	Used for customizing the appearance of plots
ggvis scales the values to a pre-defined scale of colour/sizes	ggvis sends the colour value to *vega* - a javascript package for further processing

Properties for points

The properties for points are -

```
fill,x, y, stroke, strokeWidth, strokeOpacity, fillOpacity, shape, size
```

Sample code:

```
faithful %>% ggvis(waiting, eruptions, fillOpacity = ~eruptions, size := 100, fill := "red")
```

Properties for lines

The properties for lines include -

```
x, y, fill, fillOpacity, opacity, stroke, strokeDash, strokeOpacity, and strokeWidth
```

Transformations

compute_smooth	compute_bin()
It transforms the data to generate a new dataframe.	It transforms the data to generate a new dataframe.

Transformations (cont)

It returns a dataset with 2 variables, one named pred_ and the other resp_	It returns a dataset with 4 variables, x, x2 ,y y2

Syntax: compute_smooth()

Long way:

```
mtcars %>% compute_smooth(mpg ~ wt) %>% ggvis(pred_, ~resp_) %>% layer_lines()
```

In-built:

```
mtcars %>% ggvis(wt,mpg) %>% layer_smooths()
```

Syntax: compute_bin()

Long way:

```
faithful %>% compute_bin(waiting, width = 5) %>% ggvis(x = xmin_, x2 = xmax_, y = 0, y2 = count_) %>% layer_rects()
```

In-built:

```
faithful %>% ggvis( ~waiting) %>% layer_histograms(width = 5)
```

Transformations

compute_density()

A density plot uses a line to display the density of a variable at each point in its range.

It returns a data frame with two columns: pred_, the x values of the variable's density line, and resp_, the y values of the variable's density line.

Transformations (cont)

Similarly, we have compute_count() or the in built function layer_bars()

Syntax: compute_smooth()

Long way:

```
faithful %>% compute_density(waiting) %>% ggvis(pred_, ~resp_) %>% layer_lines()
```

In-built:

```
faithful %>% ggvis( ~waiting, fill := "green") %>% layer_densities()
```

By **shanly3011**

cheatography.com/shanly3011/

Published 9th April, 2015.
Last updated 12th May, 2016.