Changing the plot type

ggvis Layers

- In ggplot2 you use **geom** functions to determine the type of plot that you create
- In ggvis you can use `layer` functions
- N.B. Not all geoms are currently available as layers

Layers

So far, you seen two layer functions: `layer_points()` and `layer_histograms()`. There are many other layers, and they can be roughly categorised into two types:

- Simple, which include primitives like points, lines and rectangles.
- Compound, which combine data transformations with one or more simple layers.

All layer functions use the plural, not the singular.

> *Think the verb, not the noun: Im going to layer some points onto my plot.*

Layers

Function	Description
layer_points	Adds data as points
layer_histograms	Adds data as a histogram
layer_boxplots	Draws as a boxplot
layer_lines	Adds data as lines
layer_smooths	Adds a smoothing line
layer_paths	Joins data as a single path
layer_text	Adds text
layer_model_predictions	Adds lines for model predictions, such as lm

Layers

Simple layers

There are five simple layers:

1. **Points** -`layer_points`
2. **Paths** and **polygons**, - `layer_paths()`.
3. **Filled Areas** - `layer_ribbons()`
4. **Rectangles** - `layer_rects()`
5. **Text** - `layer_text()`

1. **Points,** `layer_points()`

properties: `x`, `y`, `shape`, `stroke`, `fill`, `strokeOpacity`, `fillOpacity`, and `opacity`.

```
mtcars  %>%
   ggvis(~wt, ~mpg) %>%
      layer_points()
```

2. Paths and polygons, layer_paths().

```
df <- data.frame(x = 1:10,
        y = runif(10))

df %>%
    ggvis(~x, ~y) %>%
    layer_paths()
```

Layers

If you supply a `fill`, youll get a polygon

```
t <- seq(0, 2 * pi, length = 100)
df <- data.frame(x = sin(t), y = cos(t))

df %>%
    ggvis(~x, ~y) %>%
    layer_paths(fill := "red")
```

3. Filled areas, `layer_ribbons()`

Use properties y and y2 to control the extent of the area.

```
df <- data.frame(x = 1:10,
     y = runif(10))
df %>%
   ggvis(~x, ~y) %>%
   layer_ribbons()
```

Layers

```
df %>% ggvis(~x, ~y + 0.1,
             y2 = ~y - 0.1) %>%
       layer_ribbons()
```

Layers

4. Rectangles, `layer_rects()`.

The location and size of the rectangle is controlled by the x, x2, y and y2 properties.

```
set.seed(1014)
df <- data.frame(x1 = runif(5), x2 = runif(5),
          y1 = runif(5), y2 = runif(5))

df %>% ggvis(~x1, ~y1,
             x2 = ~x2, y2 = ~y2,
             fillOpacity := 0.1)  %>%
       layer_rects()
```

Layers

5. Text, layer_text()..

The `text` layer has many new options to control the apperance of the text:

- `text` (the label),
- `dx` and `dy` (margin in pixels between text and anchor point),
- `angle` (rotate the text),
- `font` (font name) and fontSize (size in pixels),
- `fontWeight` (e.g. bold or normal),
- `fontStyle` (e.g. italic or normal.)

Layers

```
df <- data.frame(x = 3:1,
                 y = c(1, 3, 2),
                 label = c("a", "b", "c"))

df %>% ggvis(~x, ~y, text := ~label)
   %>% layer_text()
```

Layers

```
df %>%
  ggvis(~x, ~y, text := ~label) %>%
  layer_text(fontSize := 50)
```

Layers

```
df %>%
    ggvis(~x, ~y, text := ~label) %>%
    layer_text(angle := 45)
```

Compound layers

The four most common compound layers are:

1. `layer_paths()`
2. `layer_histograms()`
3. `layer_polygons()`
4. `layer_smooths()`

Layers

`layer_lines()` which automatically orders by the x variable:

```
t <- seq(0, 2 * pi, length = 20)
df <- data.frame(x = sin(t), y = cos(t))

df %>%
   ggvis(~x, ~y) %>%
   layer_paths()
```

Compound layers

```
df %>%
   ggvis(~x, ~y) %>%
   layer_lines()
```

Layers

`layer_lines()` is equivalent to `arrange()` + `layer_paths()`:

```
df %>%
   ggvis(~x, ~y) %>%
   arrange(x) %>%
   layer_paths()
```

Layers

`layer_histograms()` and `layer_freqpolys()`

- `layer_histograms()` and `layer_freqpolys()` which allows you to explore the distribution of continuous.
- Both layers first bin the data with `compute_bin()` then display the results with either rects or lines.

Layers

```
mtcars  %>%
    ggvis(~mpg) %>%
    layer_histograms()

# Guessing width = 1
# range / 24
```

```
# Or equivalently
binned <- mtcars %>% compute_bin(~mpg)

#  Guessing width = 1
#  range / 24
binned   %>% ggvis(x = ~xmin_,
         x2 = ~xmax_,
         y2 = 0, y = ~count_,
         fill := "black") %>%
  layer_rects()
```

Compound Layers

`layer_smooths()` fits a smooth model to the data, and displays predictions with a line.
Its used to highlight the trend in noisy data:

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

You can control the degree of *wiggliness* with the span parameter:

```
span <- input_slider(0.2, 1, value = 0.75)
mtcars  %>%
  ggvis(~wt, ~mpg) %>%
  layer_smooths(span = span)
```

Vignette

- You can learn more about layers in the **layers** vignette.