Recall:

The **magrittr** package allows you to rewrite the previous function call as:

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

Pipe operator must be at the end of line, if using multiple lines

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

This following code LOOKS neat, but doesn't work.

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

Data Visualization with ggvis

- This style of programming (i.e. using the pipe operator) also allows gives you a lot of power when you start creating a lot of power.
- Also it allows you to seemlessly intermingle **ggvis** and **dplyr** code (*Next Slide*).

Data Visualization with ggvis

```
library(dplyr)
# convert engine displacment to litres

mtcars %>%
  ggvis(x = ~mpg, y = ~disp) %>%
      mutate(disp = disp / 61.0237) %>%
         layer_points()
```

Calling Formulas

- The format of the visual properties needs a little explanation.
- We use $\sim$ before the variable name to indicate that we dont want to literally use the value of the **mpg** variable (which doesnt exist), but instead we want we want to use the **mpg** variable inside in the dataset.
- This is a common pattern in **ggvis**: well always use formulas to refer to variables inside the dataset.

Data Visualization with ggvis

The first two arguments to `ggvis()` are usually the position, so by convention you can drop `x` and `y`:

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

(x for mpg, y for displacement)

All the mtcars variables

```
> names(mtcars)
 [1] "mpg"  "cyl"  "disp" "hp"   "drat"
 [6] "wt"   "qsec" "vs"   "am"   "gear"
[11] "carb"
>
```

Data Visualization with ggvis

You can add more variables to the plot by mapping them to other visual properties like **fill, stroke, size** and **shape**.

```
mtcars %>%
  ggvis(~mpg, ~disp, stroke = ~vs) %>%
     layer_points()
```

The "fill" property

```
mtcars %>%
    ggvis(~mpg, ~disp, fill = ~vs) %>%
       layer_points()
```

The "size" property

```
mtcars %>%
    ggvis(~mpg, ~disp, size = ~vs) %>%
        layer_points()
```

The "shape" property

```
mtcars %>%
    ggvis(~mpg, ~disp,
          shape = ~factor(cyl)) %>%
    layer_points()
```

The ":=" operator

- If you want to make the points a fixed colour or size, you need to use `:=` instead of `=`.
- The `:=` operator means to use a raw, unscaled value.
- This seems like something that `ggvis()` should be able to figure out by itself, but making it explicit allows you to create some useful plots that you couldnt otherwise.

Data Visualization with ggvis

```
mtcars  %>%
    ggvis(~wt, ~mpg, fill := "red",
          stroke := "black") %>%
       layer_points()
```

Data Visualization with ggvis

```
mtcars   %>%
     ggvis(~wt, ~mpg,
          size := 300,
          opacity := 0.4)    %>%
     layer_points()
```

Data Visualization with ggvis

```
mtcars  %>%
    ggvis(~wt, ~mpg,
        shape := "cross")   %>%
    layer_points()
```