Data Visualization with ggplot2

by Erika Siregar (@erikaris) Presented in R-Ladies Jakarta 4th Meetup

October 26, 2019

Why do we visualize?

- to get better understanding of the data.
- to understand relationship between variables.
- to help discovering insights.

How to visualize in R?

- basic R plot
- ggplot2

What is ggplot2?

- ggplot2 is a library for declaratively creating graphics
- it is based on The Grammar of Graphics.
  - Grammar of Graphics is a concept that defines a plot as a set of component layers: **aesthetic and geometry**.
  - You **provide the data**, tell ggplot2 how to map variables to **aesthetics**, what graphical **geometries** to use, and it takes care of the details.

Grammar of Graphic

A plot is basically a stack of layers consisting of:

- data

Grammar of Graphic

A plot is basically a stack of layers consisting of:

- data
- aesthetic –> aes()

Grammar of Graphic

A plot is basically a stack of layers consisting of:

- data
- aesthetic –> aes()
- geometry –> geom_

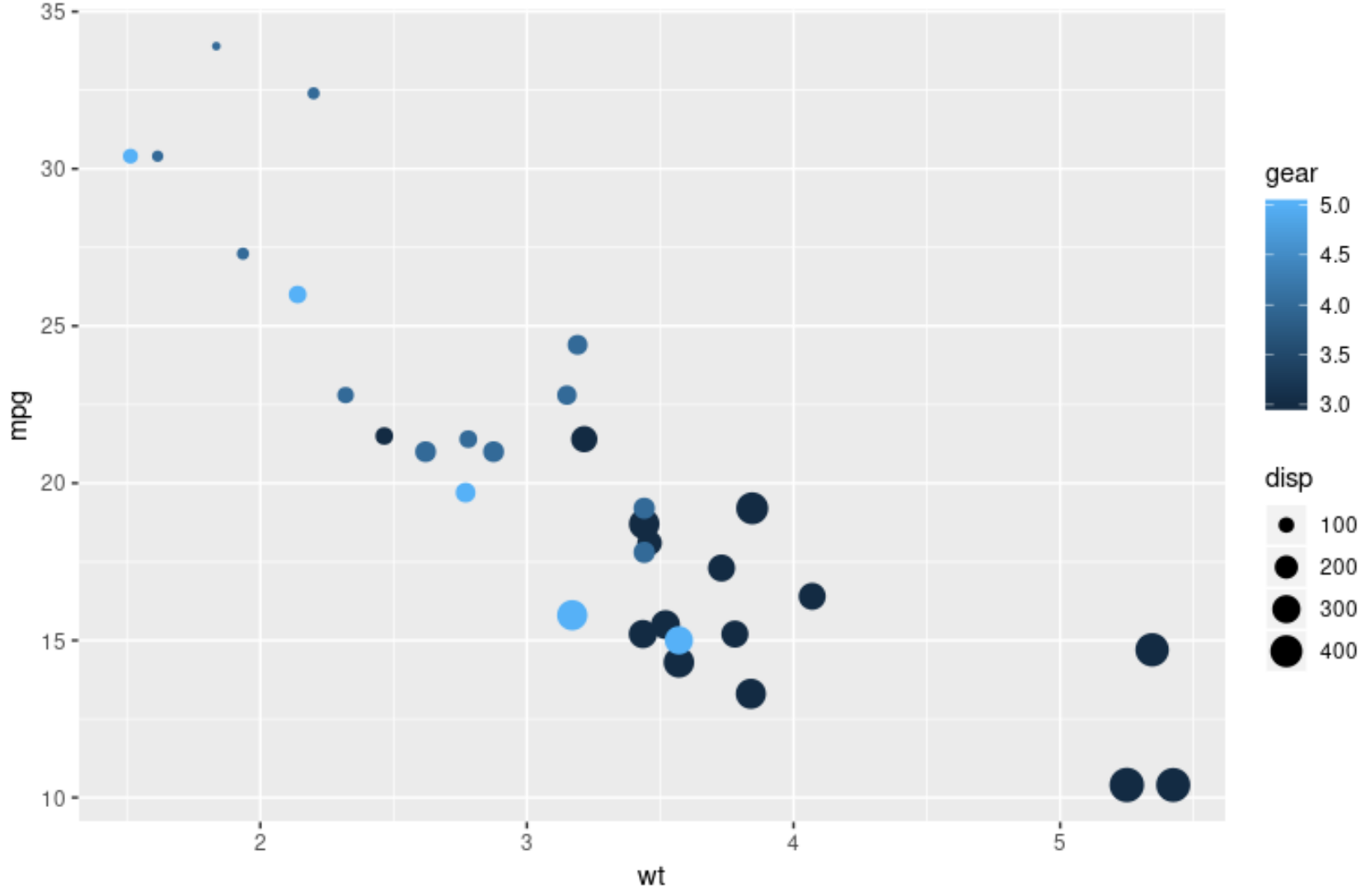
Grammar of Graphic

A plot is basically a stack of layers consisting of:

- data
- aesthetic –> aes()
- geometry –> geom_
- theme –> theme()

Take a look at this plot

A scatter plot of `mtcars` data with `x` and `y` axes are mapped to `wt` and `mpg` columns, respectively.

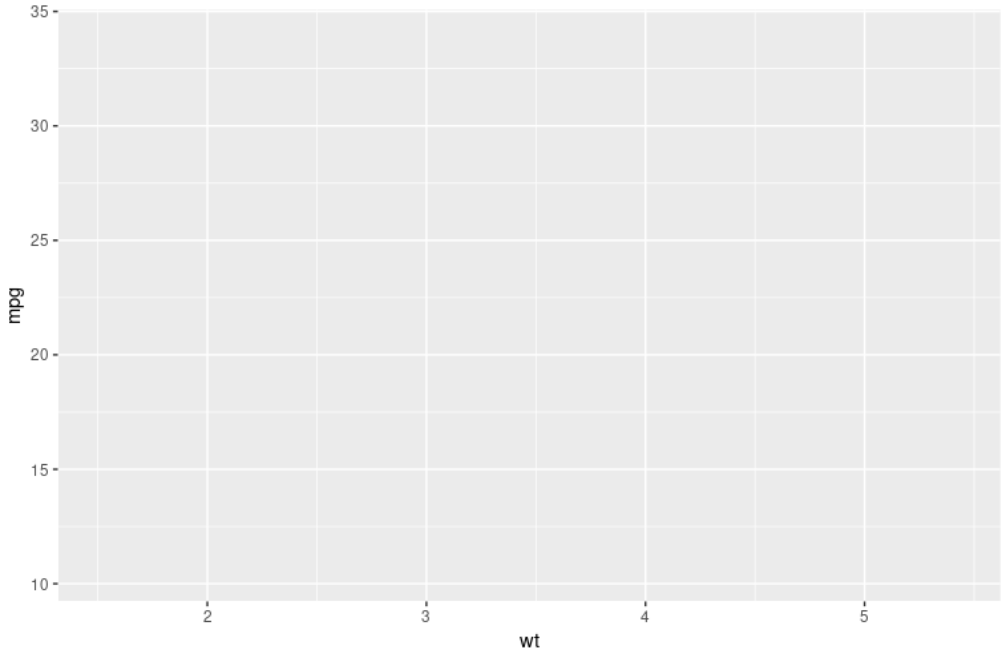


Now, look at the graphic in a 'Grammar of Graphics' way.

A plot is just a blank canvas

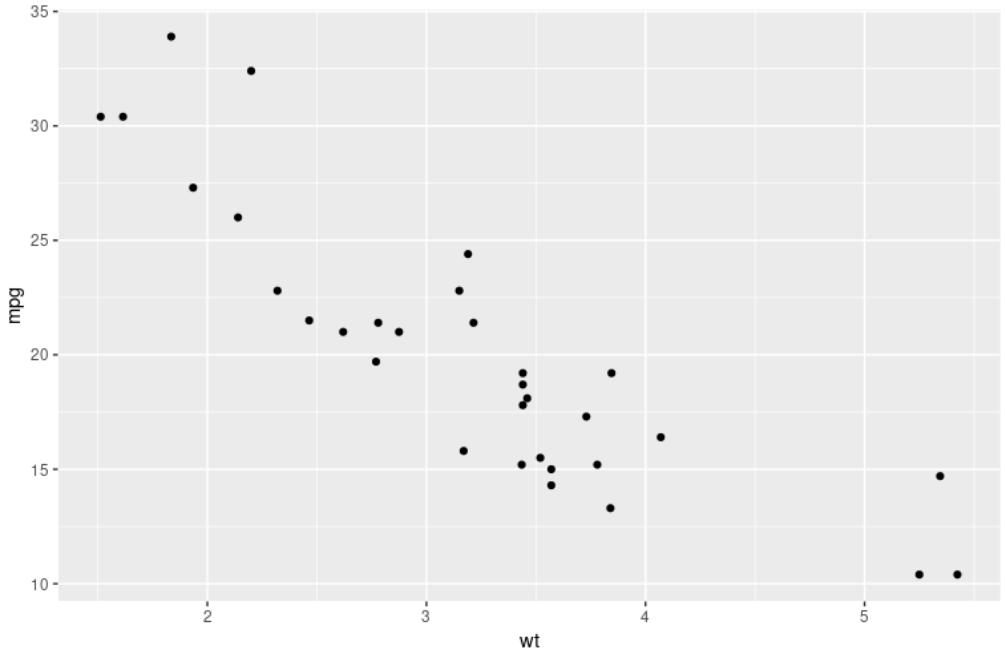
Now, look at the graphic in a 'Grammar of Graphics' way.

A plot is just a blank canvas + aesthetic



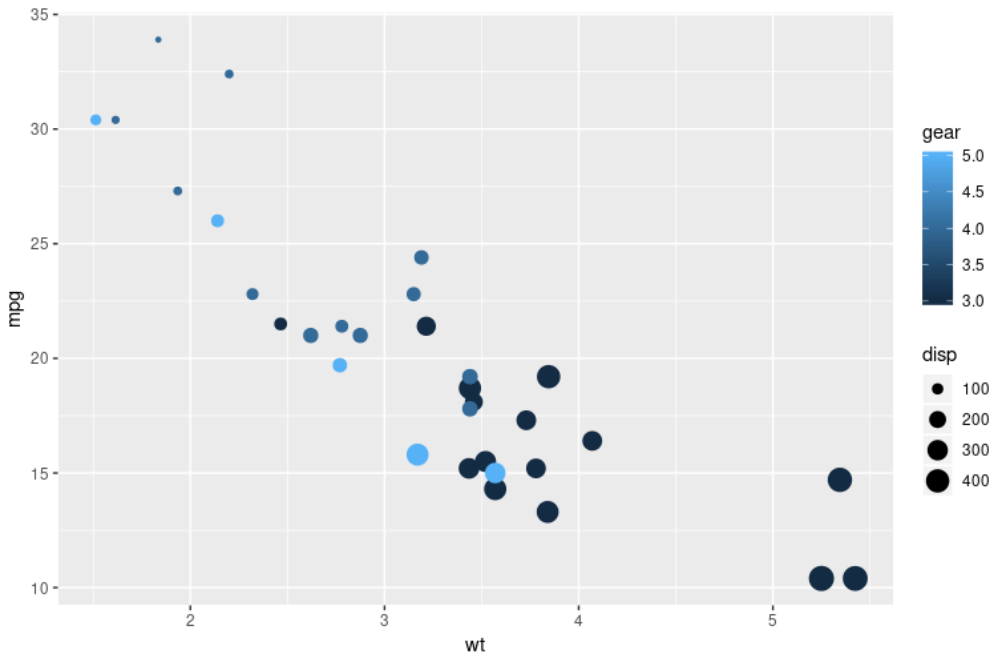
Now, look at the graphic in a 'Grammar of Graphics' way.

A plot is just a blank canvas + aesthetic + geometry



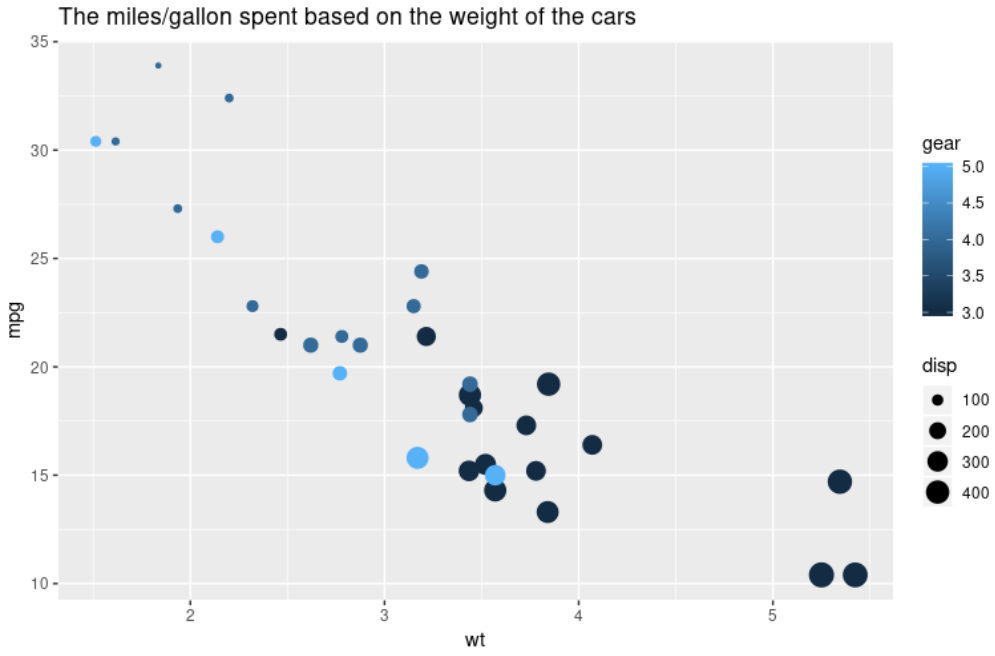
Now, look at the graphic in a 'Grammar of Graphics' way.

A plot is just a blank canvas + aesthetic + geometry + more aesthetics



Now, look at the graphic in a 'Grammar of Graphics' way.

A plot is just a blank canvas + aesthetic + geometry + more aesthetics + labels



Translate the Grammar of Graphics to ggplot

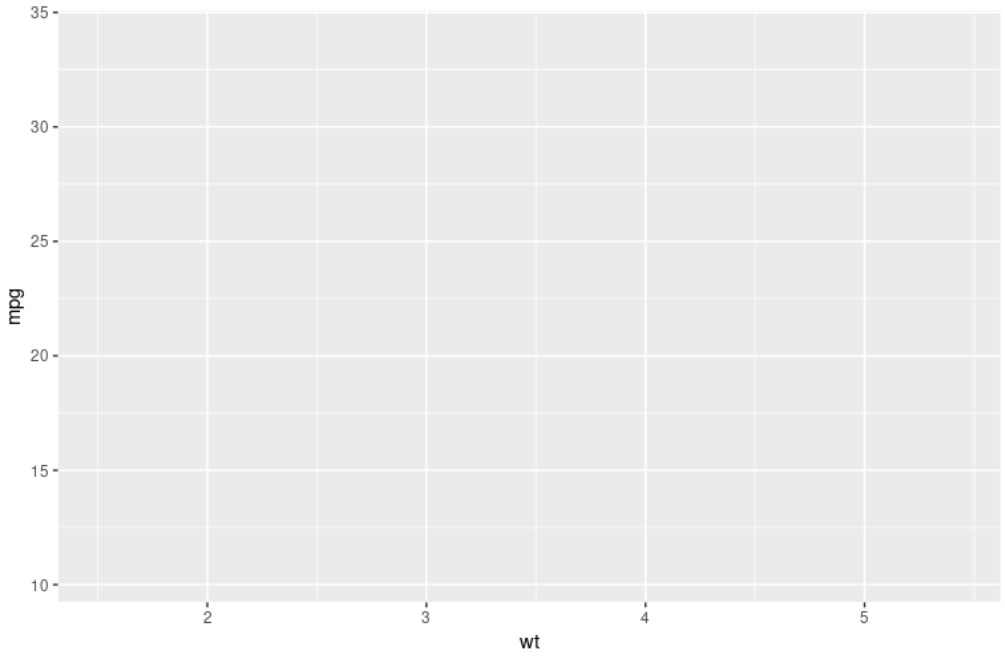
A plot is just a blank canvas

```
ggplot(mtcars)
```

Translate the Grammar of Graphics to ggplot

A plot is just a blank canvas + aesthetic

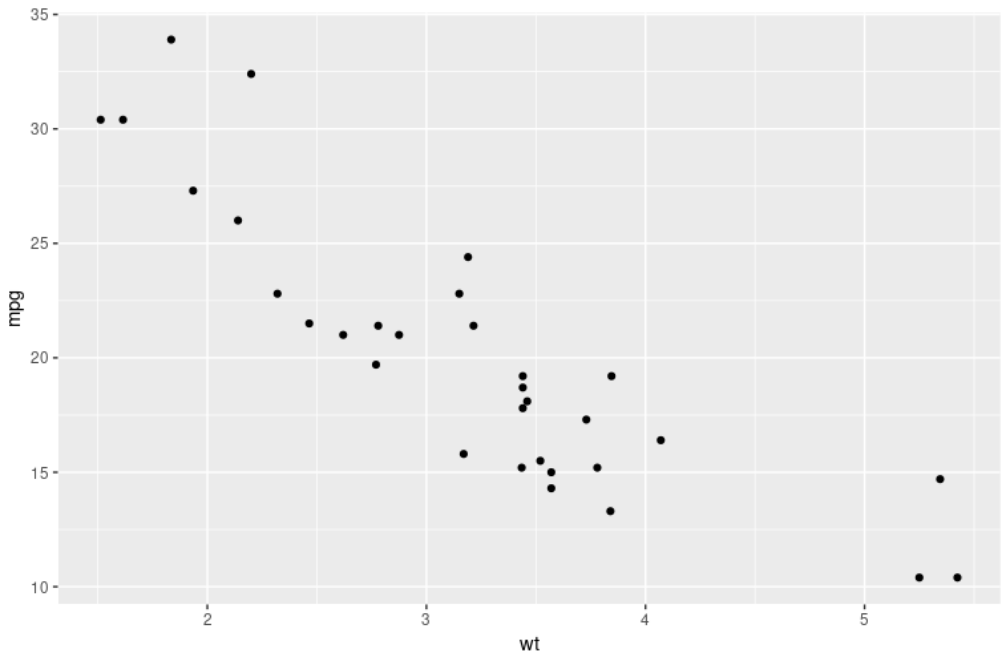
```
ggplot(mtcars, aes(x = wt, y = mpg))
```



Translate the Grammar of Graphics to ggplot

A plot is just a blank canvas + aesthetic + geometry

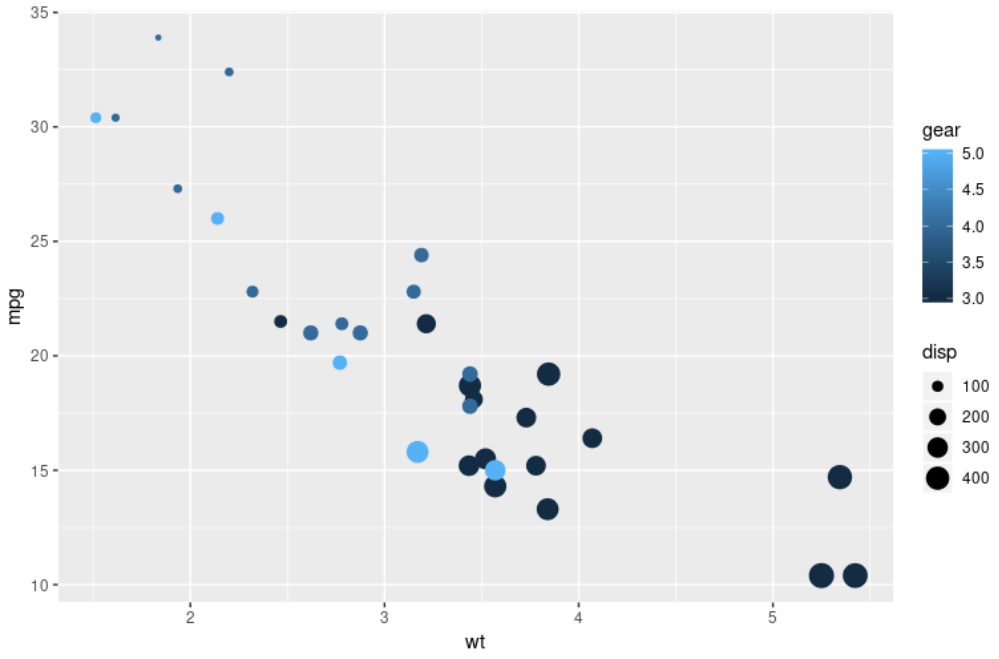
```
ggplot(mtcars, aes(x = wt, y = mpg, size = disp, color =
geom_point()
```



Translate the Grammar of Graphics to ggplot

A plot is just a blank canvas + aesthetic + geometry + more aesthetics

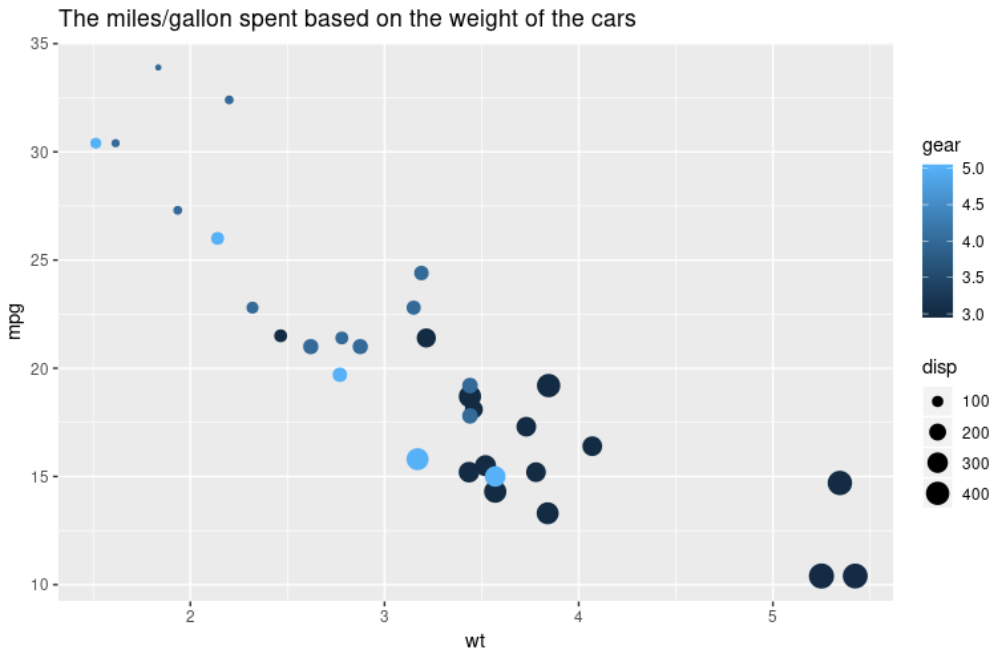
```
ggplot(mtcars, aes(x = wt, y = mpg, size = disp, color =
geom_point()
```



Translate the Grammar of Graphics to ggplot

A plot is just a blank canvas + aesthetic + geometry + more aesthetics + labels

```
ggplot(mtcars, aes(x = wt, y = mpg, size = disp, color =
geom_point() +
labs(title = "The miles/gallon spent based on the weight
```



Drawing your first plot

Preparation

- Make sure you have the library installed on your machine:

```
install.packages('ggplot2')
```

- load the ggplot library:

```
library(ggplot2)
```

- use the dataset `mtcars`, available in R Studio.
  - explore its structure:

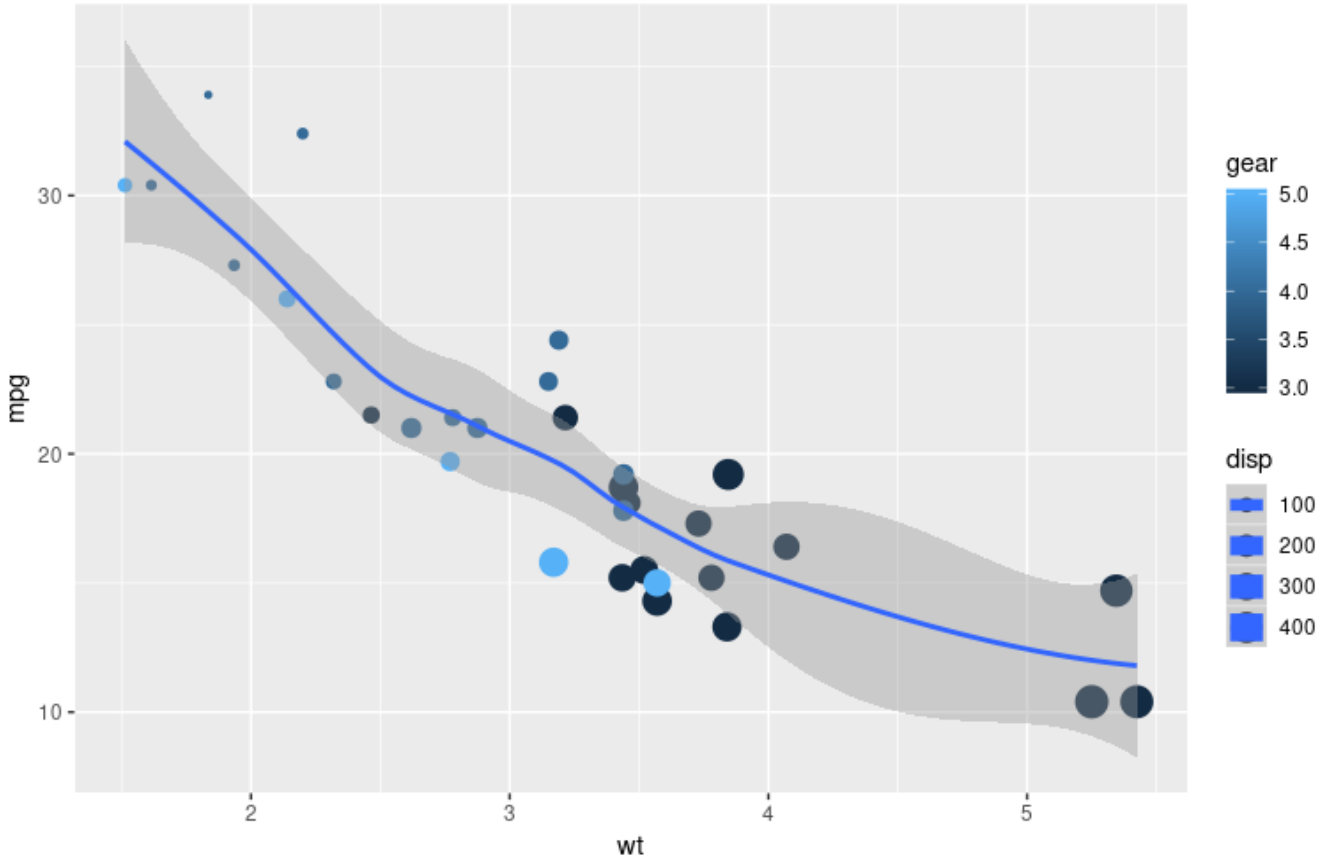
```
str(mtcars)
```

Now replicate this script on your own machine.

```
library(ggplot2)

ggplot(mtcars, aes(x = wt, y = mpg, size = disp, color =
  geom_point() +
  geom_smooth()
  labs(title = "The miles/gallon spent based on the weigh
```

Your output should look like this:



Types of Geoms (geom_*)

1. geom_point() –> for scatter plot
2. geom_bar() –> for bar chart
3. geom_area() –> for area chart
4. geom_line() –> for line chart

There are many other types of geoms. You can explore it yourself on ggplot cheatsheet.

Playing with geoms

Spend your time playing with different types of geoms by just simply changing the geom from the example that we have created earlier.

geom_bar

```
library(ggplot2)
ggplot(mtcars, aes(x = factor(cyl),
                   fill = factor(am))) +
  # try position = 'dodge', 'fill'
  geom_bar(position = 'dodge') +
  scale_x_discrete('Cylinders') +
  scale_y_continuous('Number') +
  scale_fill_manual('Transmission',
    values = c("#E41A1C", "#377EB8"),
    labels = c("Manual", "Automatic"))
```

histogram

1 - Make a univariate histogram

```
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram()
```

2 - Plot 1, plus set binwidth to 1 in the geom layer

```
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(binwidth = 1)
```

3 - Plot 2, plus MAP ..density.. to the y aesthetic (i.e. in a second aes() function)

```
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(binwidth = 1, aes(y = ..density..))
```

4 - plot 3, plus SET the fill attribute to "#377EB8"

```
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(binwidth = 1, fill = "#377EB8", aes(y = ..
```

Playing with aesthetics

Basic scatter plot: wt on x-axis and mpg on y-axis; map cyl to col

```
ggplot(mtcars, aes(x=wt, y=mpg, color=cyl)) + geom_point(s
```

Hollow circles

```
ggplot(mtcars, aes(x=wt, y=mpg, color=cyl)) + geom_point(s
```

Add transparency

```
ggplot(mtcars, aes(x=wt, y=mpg, color=cyl)) + geom_point(s
```

Playing with facet

- Facet = splitting a plot into several subplots.
- Let's play using the `iris` dataset.
- Let's take a brief look into the dataset

```
  str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.
##  $ Species     : Factor w/ 3 levels "setosa","versicolor
```

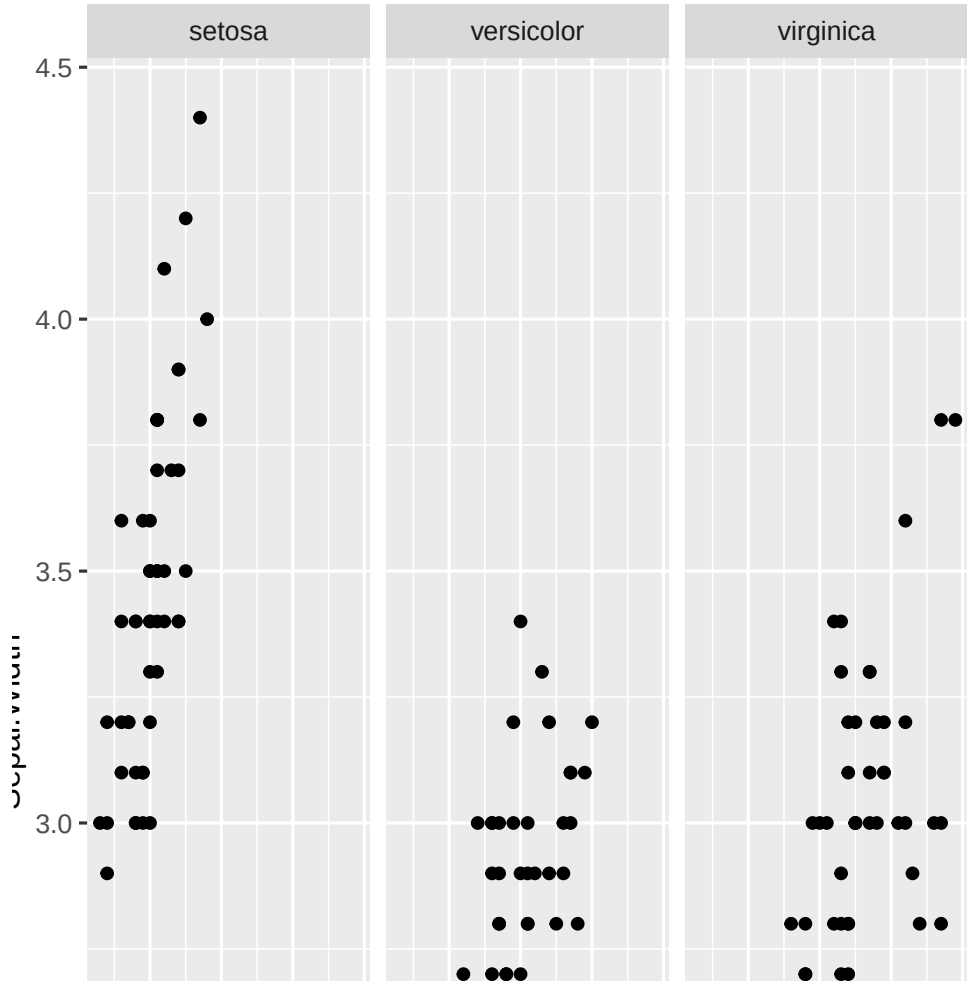
```
  names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.
## [5] "Species"
```

Playing with facet

Now, let's start playing

```
ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width))+
  geom_point() +
  facet_grid(. ~ Species)
```

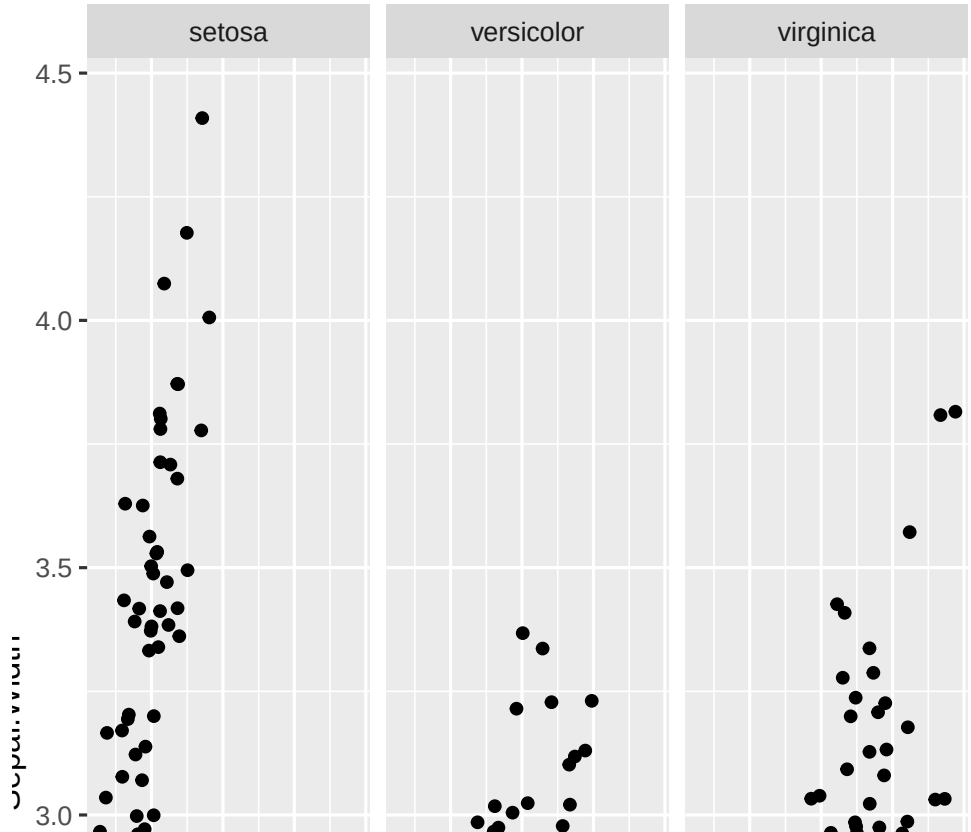


Playing with facet

use geom_jitter() instead of geom_point()

```
library(ggplot2)

ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width))+
  geom_point() +
  facet_grid(. ~ Species)
```

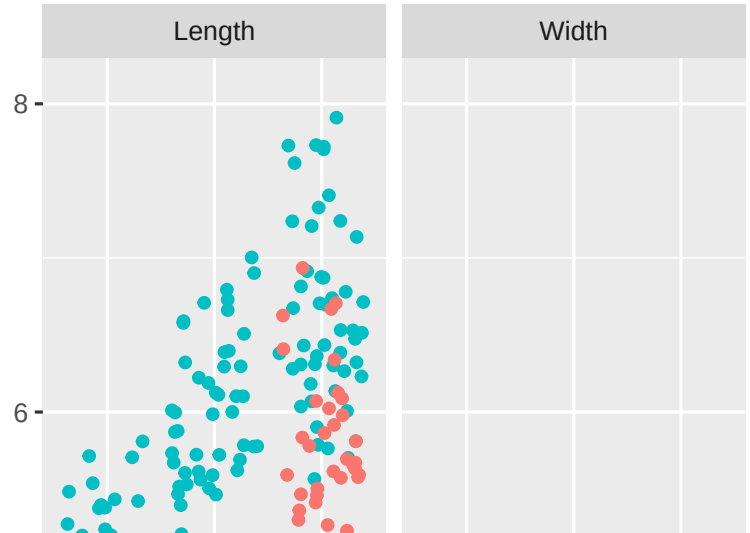


Playing with facet

Now, let's level up the game a bit. Modify the `iris` dataset to get a more interesting visualization.

```
library(tidyr)

iris %>%
  gather(key, value, -Species) %>%
  separate(key, c("Part","Measure"), sep="\\.") %>%
  ggplot(aes(x = Species, y = value, col = Part)) +
  geom_jitter() +
  facet_grid(. ~ Measure)
```

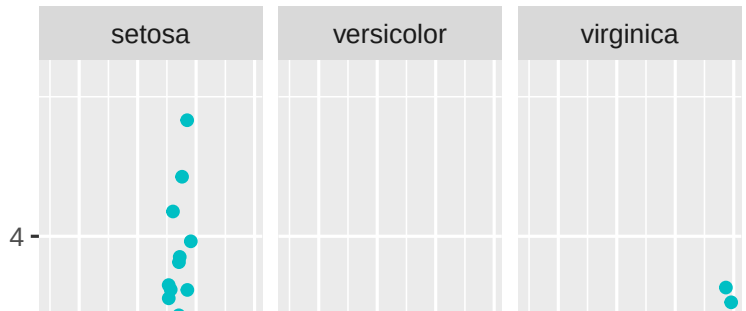


Playing with facet

Give it another try!

```
library(tidyr)
library(dplyr)
library(ggplot2)

iris %>%
  mutate(Flower = 1:nrow(iris)) %>%
  gather(key, value, -Species, -Flower) %>%
  separate(key,c("Part", "Measure"),sep="\\.")%>%
  spread(Measure, value) %>%
  ggplot(aes(x=Length, y=Width, color=Part)) +
  geom_jitter() +
  facet_grid(. ~ Species)
```



Questions?