

# Data Visualization with ggplot2

R for Data Science workshop

2019-05-01 (updated: 2019-07-15)

# Data Visualization with ggplot2

## Outline

- Overview of ggplot2
- Prerequisites
  - Tidyverse
  - Data frames
- Grammar of Graphics
- Exploratory vs Expository Plots

# Data science workflow



Image source: [R for Data Science](#) by Hadley Wickham & Garrett Golemund.

# Data science workflow

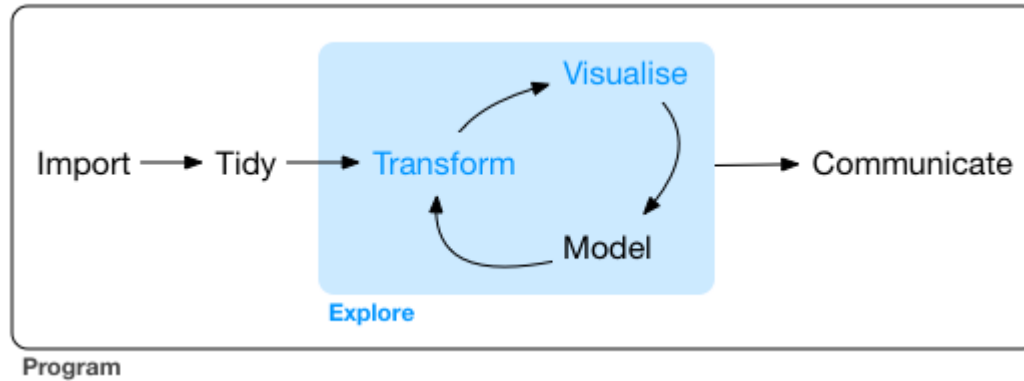


Image source: [R for Data Science](#) by Hadley Wickham & Garrett Golemund.

# ggplot2

From the official ggplot2 website:

**ggplot2** is a system for declaratively creating graphics, based on **The Grammar of Graphics**. You provide the **data**, tell ggplot2 how to map variables to **aesthetics**, what **graphical primitives** to use, and it takes care of the details.

- R package
- Developed by Hadley Wickham (PhD, Iowa State, 2008)
- Now part of the tidyverse
- Other R graphics systems: base, lattice, grid
- Extensible
- Worth the learning curve

# Prerequisites

## Tidyverse

- Opinionated collection of R packages
- Share underlying design philosophy, grammar, and data structures

## Data frames

- Rectangular data structure
- Columns represent variables
- Rows represent observations
- Tidyverse data frames are called "tibbles"

**Tidyverse packages work with tidy tibbles**

# Demo

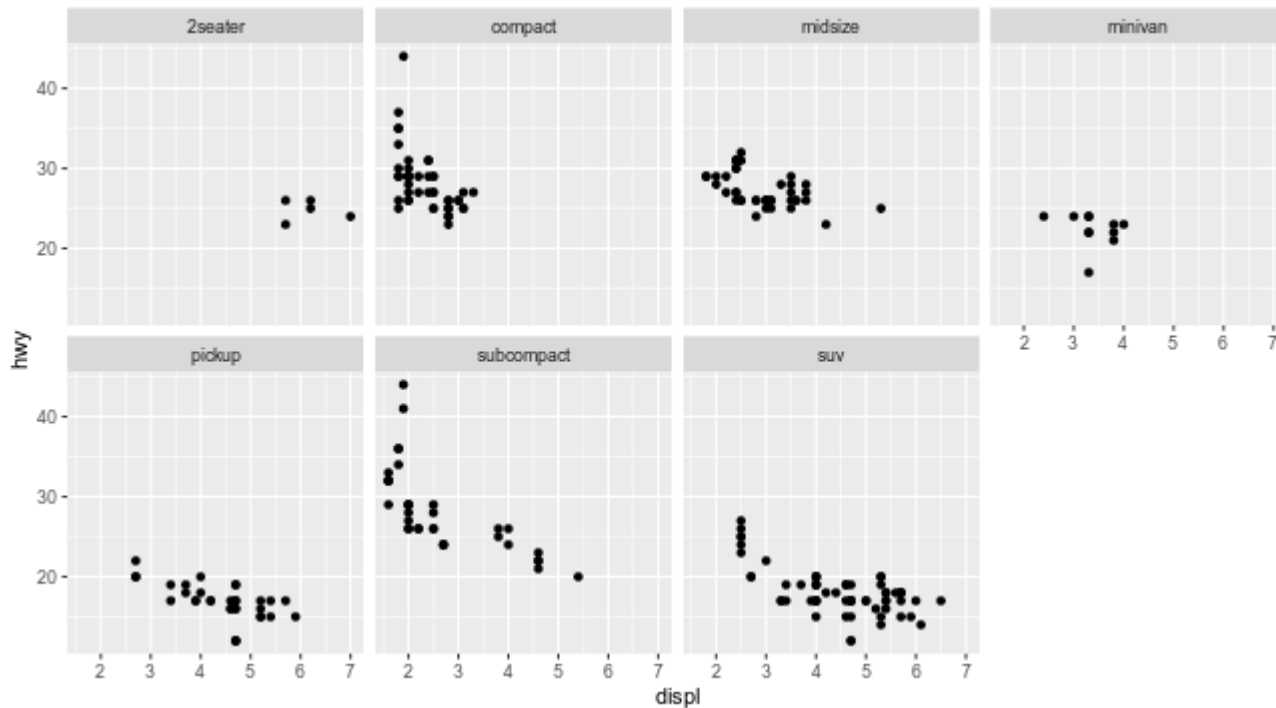
# Grammar of Graphics

- **Data** (data frame in tidy format)
- Layers of **geom**etric objects and **stat**istical transformations
- **Scales** that map data values to (and from) **aes**thetic space values (location, color, size, etc.)
- **Coord**inate system (Cartesian, map projections, polar)
- **Facet**ing (breaking data into subsets)
- **Themes**

# Facets are subplots

For example,

```
ggplot(data = mpg, mapping = aes(displ, hwy)) +  
  geom_point() +  
  facet_wrap(vars(class), nrow = 2)
```





# Grammar of Graphics

We will focus on 3 main elements of the grammar:

- **Data** (data frame in tidy format)
- Layers of **geom**etric objects
- Mappings of data values to visual properties (**aes**thetics) of geoms

# ggplot2 template

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>)) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>)) +  
  ...
```

If the geoms use the same aesthetics:

```
ggplot(data = <DATA>, mapping = aes(<MAPPINGS>)) +  
  <GEOM_FUNCTION>() +  
  <GEOM_FUNCTION>() +  
  ...
```

## Demo

# Full ggplot2 template

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(  
    mapping = aes(<MAPPINGS>),  
    stat = <STAT>,  
    position = <POSITION>  
  ) +  
  <COORDINATE_FUNCTION> +  
  <FACET_FUNCTION> +  
  <SCALE_FUNCTION> +  
  <THEME_FUNCTION>
```

Required elements: <DATA>, <GEOM\_FUNCTION>, <MAPPINGS>

Other elements have sensible defaults.

# Some named graphs and their geoms

Name	Geom
Scatterplot	geom_point()
Linegraph	geom_line()
Histogram	geom_histogram()
Boxplot	geom_boxplot()
Barplot	geom_barplot()

## Demo

# ggplot2 is extensible



Ryan Timpe  
@ryantimpe

Following

I have high hopes and low expectations for this [#brickr](#) branch. [#ggplot2](#) [#rstats](#)

```
~/bricks/brick - geom_brick - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function Addins
bricks_from.R geom_brick.R scale_image.R collect_bricks.R DESCRIPTION
1 # ggplot2 Bar Charts as Bricks
2 #
3 # 'geom_col', except bars look like LEGO(R) bricks.
4 #
5 # @inheritParams ggplot2::geom_col
6 # @export
7 geom_brick <- function(mapping = NULL, data = NULL,
8   position = "stack",
9   ...,
10   width = NULL,
11   na.rm = FALSE,
12   show.legend = NA,
13   inherit.aes = TRUE) {
14
```

3:22 PM - 11 May 2019

2 Retweets 9 Likes



1 2 9



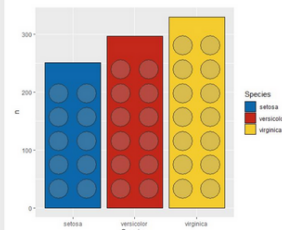
Tweet your reply



Ryan Timpe  
@ryantimpe · 24m

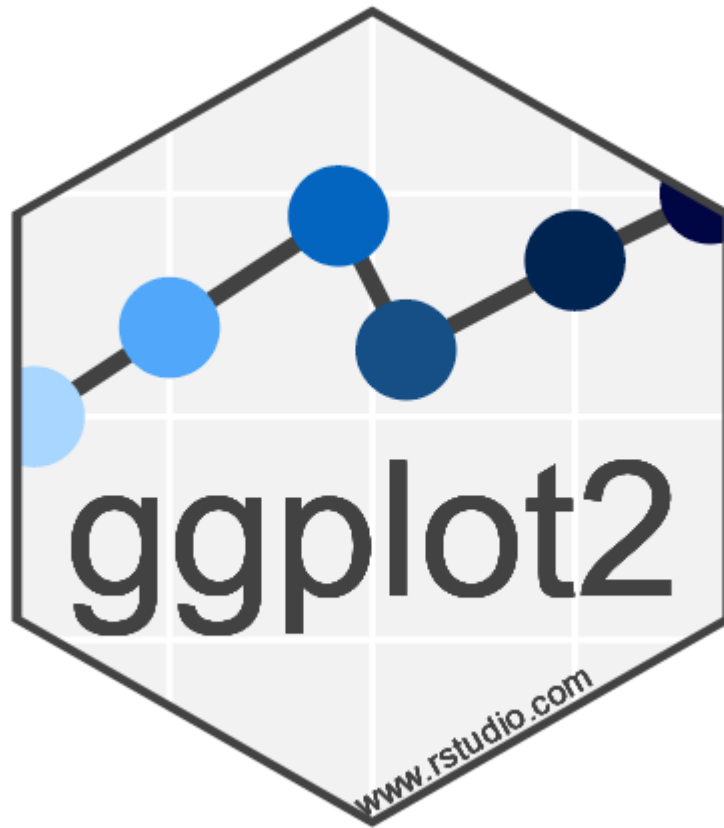
10% of the way there! At the very least, learning how the guts of ggplot work.

```
18 # (r_fig_width6, fig_height5)
19 library(ggplot2)
20 library(brickr)
21
22 fill_colors <- lego_colors %>%
23   dplyr::filter(color %in% c("bright red", "bright blue", "bright yellow")) %>%
24   dplyr::pull(hex)
25
26 iris %>%
27   dplyr::count(species, wt = sepal.length) %>%
28   ggplot(aes(x=species, y=n)) +
29   geom_brick(aes(fill = species)) +
30   scale_fill_manual(values = fill_colors)
31
```



2

You can create almost any 2-dimensional plot with ggplot2



15:00

# Your turn

Data visualization with ggplot2

[your-turn/01-data-visualization-with-ggplot2.Rmd](#)