Creating a new ggtheme

Jarred Robidoux

2023-02-07

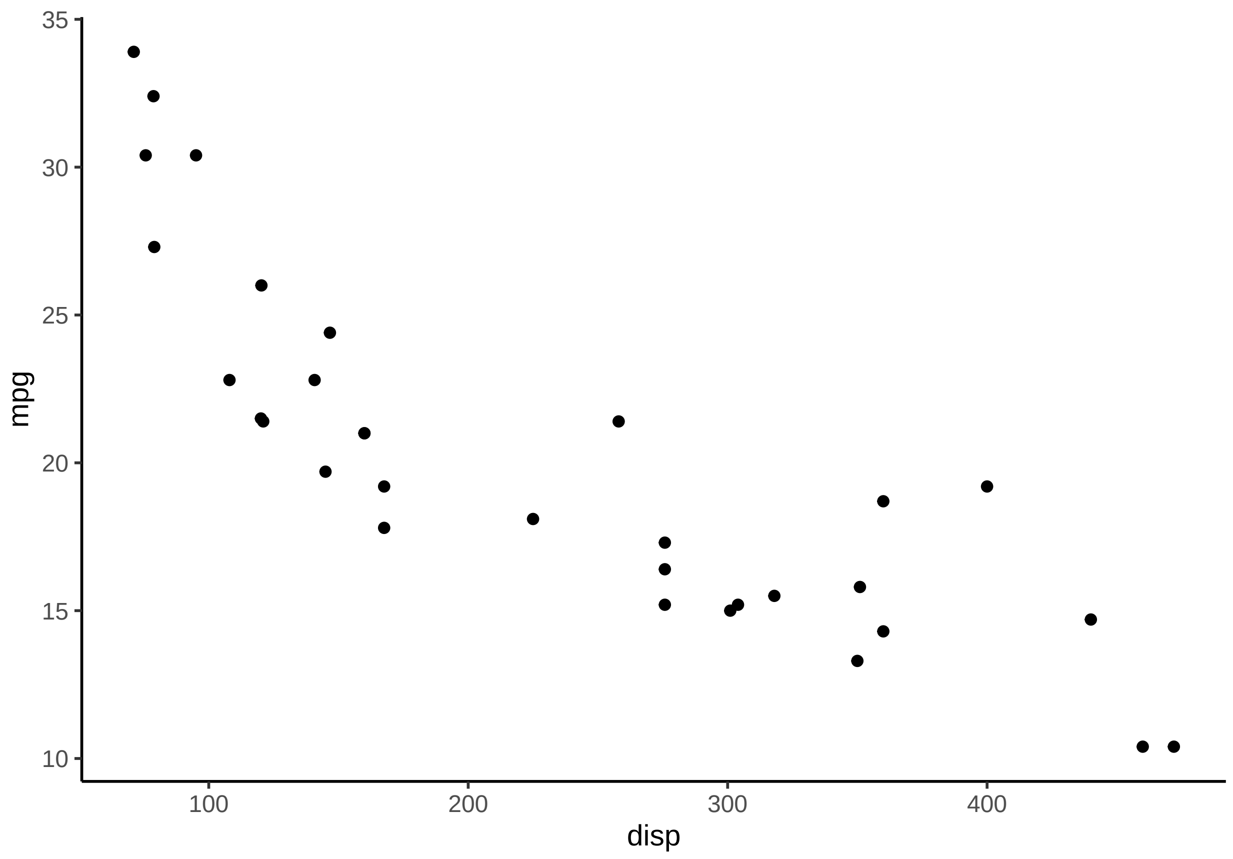
Building a New Theme

Building and modifying a theme in ggplot2 is a key feature of the ggplot2 package and system for building data graphics. The ggplot2 package implements the notion of a theme for its plots by allowing you to modify many different elements of a plot and to store all those modifications as a special "theme" object.

The default theme for ggplot2 is encapsulated by the theme_gray() function. Like other elements in the ggplot2 universe, themes can be "added" using the + operator to plot commands in order to change the look and feel of a plot.

For example, here is a plot that uses the theme_classic() function:

```r
knitr::opts_chunk$set(echo = TRUE)
mtcars %>%
  ggplot()+
  geom_point(mapping=aes(x=disp, y=mpg))+
  theme_classic()
```



Why Build a New Theme?

For many people, building a new theme is a matter of personal preference with respect to colors, shapes, fonts, positioning of labels, etc. Because plots, much like writing, are an expression of your ideas, it is often desirable to customize those plots so that they accurately represent your vision.

Developing themes is also a powerful branding tool. Plots that are distributed on the web or through marketing materials that have a common theme can be useful for reinforcing a brand.

Default Theme

As ntoed above, ggplot2 has a default theme, which is theme_gray(). This theme produces the familiar gray-background-white-grid-lines plot. You can obtain the default theme using the theme_get() function.

```
knitr::opts_chunk$set(echo = TRUE)
x <- theme_get()
class(x)
```

```
## [1] "theme" "gg"
```

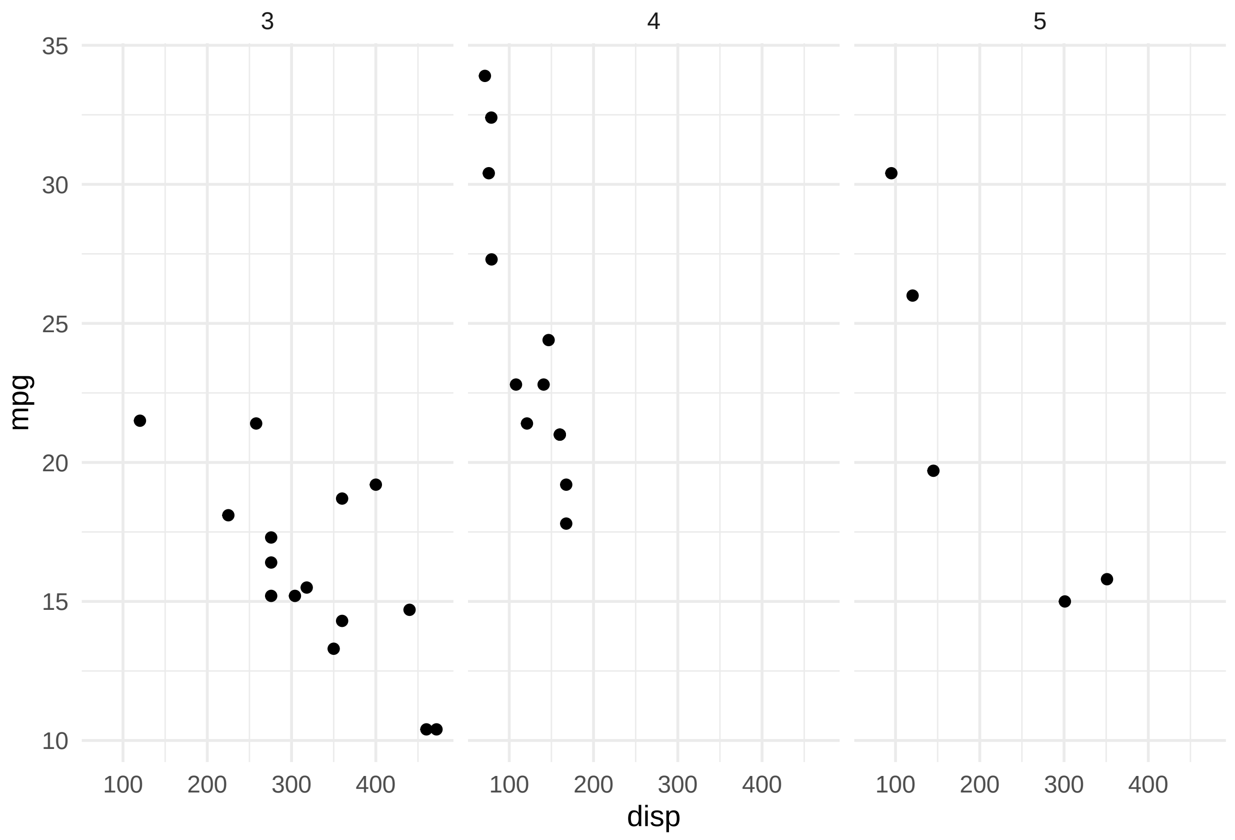
Notice that the object returned by theme_get() is an S3 object of class “theme” and “gg”. This is the kind of object you will need to create or modify in order to customize your theme.

You can modify the default theme by using the theme_set() function and passing it a theme object. For example, if we want all my plots to use the theme_minimal() theme, we could do

```
knitr::opts_chunk$set(echo = TRUE)
new_theme <- theme_minimal()
theme_set(new_theme)
```

Now your plots will use the theme_minimal() theme without you having to specify it

```
mtcars %>%
  ggplot()+
  geom_point(aes(x=disp, y=mpg))+
  facet_grid(.~gear)
```



Quitting R will erase the default theme setting. If you load ggplot2 in a future session it will revert to the default gray theme. If you'd like for ggplot2 to always use a different theme (either yours or one of the built-in ones), you can set a load hook and put it in your .Rprofile file. For example, the following hook sets the default theme to be theme_minimal() every time the ggplot2 package is loaded

```
# setHook(packageEvent("ggplot2", "onLoad"),
        # function(...)ggplot2::theme_set(ggplot2::theme_minimal))
```

Creating a New Theme

Perhaps the easiest thing to start with when customizing your own theme is to modify an existing theme (i.e. one that comes built-in to ggplot2). In case you're interested in thoroughly exploring this area and learning from others, there is also the ggthemes package on CRAN which provides a number of additional themes for ggplot2.

We will begin with the theme_bw() theme. This theme is a simple black and white theme that has little ornamentation and few features.

Modifying theme attributes

Suppose we want to make the default color for plot titles to be dark red. We can change just that element by adding a theme() modification to the exisiting theme.

```
newtheme <- theme_bw() + theme(plot.title = element_text(color = "darkred"))
```

Note thhat in our call to theme(), when we modify the plot.title attribute, we cannot simply say color = "darkred". This must be wrapped in a call to the element_text() function so that the elements of plot.title are appropriately modified. In the help page for theme(), you will see that each attribute of a theme is modified by using on or four element_* functions:

-element_text(): Specify the display of text elements -element_line(): Specify the disply of lines (i.e. axis lines) -element_rect(): Specify the display of borders and backgrounds -element_blank(): draw nothing

Let's change a few more things about our new theme. We can make the box surrounding the plot to look a little different by modifying the panel.border element of the theme. First let's take a look at what the value is by default.

```
newtheme$panel.border
```

```
## List of 5
##  $ fill         : logi NA
##  $ colour       : chr "grey20"
##  $ linewidth    : NULL
##  $ linetype     : NULL
##  $ inherit.blank: logi TRUE
##  - attr(*, "class")= chr [1:2] "element_rect" "element"
```

You can see that this is an object of class element_rect and there are 5 elements in this list, including the fill, color, sizer, and linetype. These attributes have the same meaning as they do in the usual ggplot2 context.

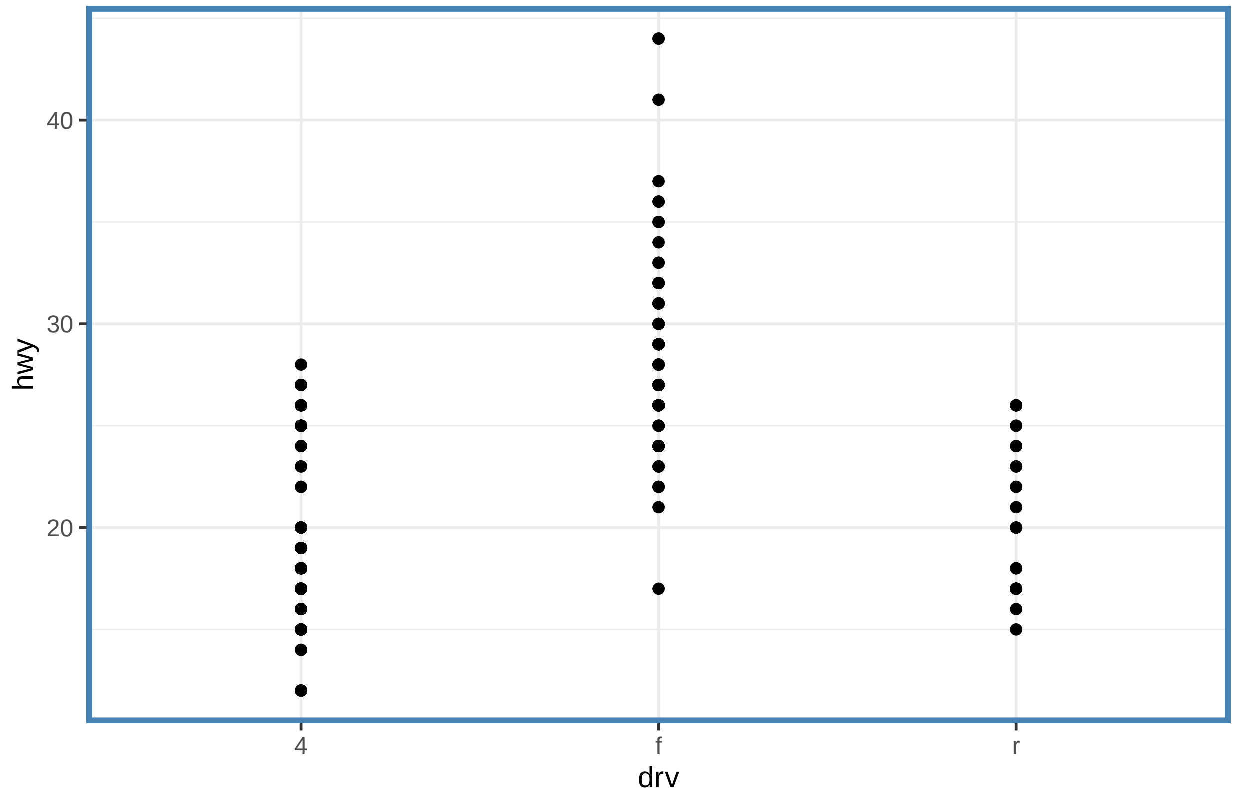
Let's modify the color attribute to make it "steelblue" and modify the size attribute to make it a little bigger

```
newtheme <- newtheme + theme(panel.border = element_rect(color = "steelblue", linewidth = 2))
```

Now let's see what this plot looks like

```
mpg %>%
  ggplot()+
  geom_point(aes(x=drv, y=hwy))+
  ggtitle("HWY MPG vs DRV")+
  newtheme
```

HWY MPG vs DRV


hwy
f
r
drv