Coding errors are an inevitable part of writing code—especially when you're first learning a new programming language. In this reading, you will learn how to recognize common coding errors when creating visualizations using **ggplot2**. You'll also find links to some resources that you can use to help address any coding problems you might encounter moving forward.

Common coding errors in ggplot2

When working with R code in ggplot2, many of the most common coding errors involve issues with syntax, such as misplaced characters. That is why paying attention to details is such an important part of writing code. When there's an error in your code that R is able to detect, it will generate an error message. Error messages can help point you in the right direction, but they won't always help you figure out the precise problem.

Now, explore a few of the most common coding errors you might encounter in ggplot2.

Case sensitivity

R code is case sensitive. If you accidentally capitalize the first letter in a certain function, it might affect your code. Here's an example:

```
Glimpse(penguins)
```

The error message lets you know that R cannot find a function named "Glimpse":

```
Error in Glimpse(penguins) : could not find function "Glimpse"
```

But you know that the function glimpse (lowercase "g") does exist. Notice that the error message doesn't explain exactly what is wrong, but does point you in a general direction.

Based on that, you can figure out that this is the correct code:

```
glimpse(penguins)
```

Balancing parentheses and quotation marks

Another common R coding error involves parentheses and quotation marks. In R, you need to make sure that every opening parenthesis in your function has a closing parenthesis, and every opening quotation mark has a closing quotation mark. For example, if you run the following code, nothing happens. R does not create the plot. That is because the second line of code is missing two closing parentheses:

```
ggplot(data = penguins) +
```

```
   geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g
```

RStudio does alert you to the problem. To the left of the line of code in your RStudio source editor, you might notice a red circle with a white X in the center. If you hover over the circle with your cursor, this message appears:

```
  405   ggplot(data = penguins) +
❌ 406     geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g
  4┌──
  4│ unmatched opening bracket '('
```

RStudio lets you know that you have an unmatched opening bracket. So, to correct the code, you know that you need to add a closing bracket to match each opening bracket.

Here's the correct code:

```
ggplot(data = penguins) +
```

```
 geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g))
```

Using the plus sign to add layers

In ggplot2, you need to add a plus sign (+) to your code when you add a new layer to your plot. Putting the plus sign in the wrong place is a common mistake. The plus sign should always be placed at the end of a line of code, not the beginning of a line.

Here's an example of code that includes incorrect placement of the plus sign:

```
ggplot(data = penguins)
```

```
  + geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g))
```

In this case, R's error message identifies the problem and prompts you to correct it:

```
Error: Cannot use `+.gg()` with a single argument. Did you
accidentally put + on a new line?
```

Here's the correct code:

```
ggplot(data = penguins) +
```

```
  geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g))
```

You also might accidentally use a pipe instead of a plus sign to add a new layer to your plot, like this:

```
ggplot(data = penguins)%>%

  geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g))
```

You then get the following error message:

```
Error: `data` must be a data frame, or other object coercible by
`fortify()`, not an S3 object with class gg/ggplot
```

Here's the correct code:

```
ggplot(data = penguins) +

  geom_point(mapping = aes(x = flipper_length_mm, y = body_mass_g))
```

Keep in mind these issues and pay attention to details when you write code will help you reduce errors and save time, so you can stay focused on your analysis.
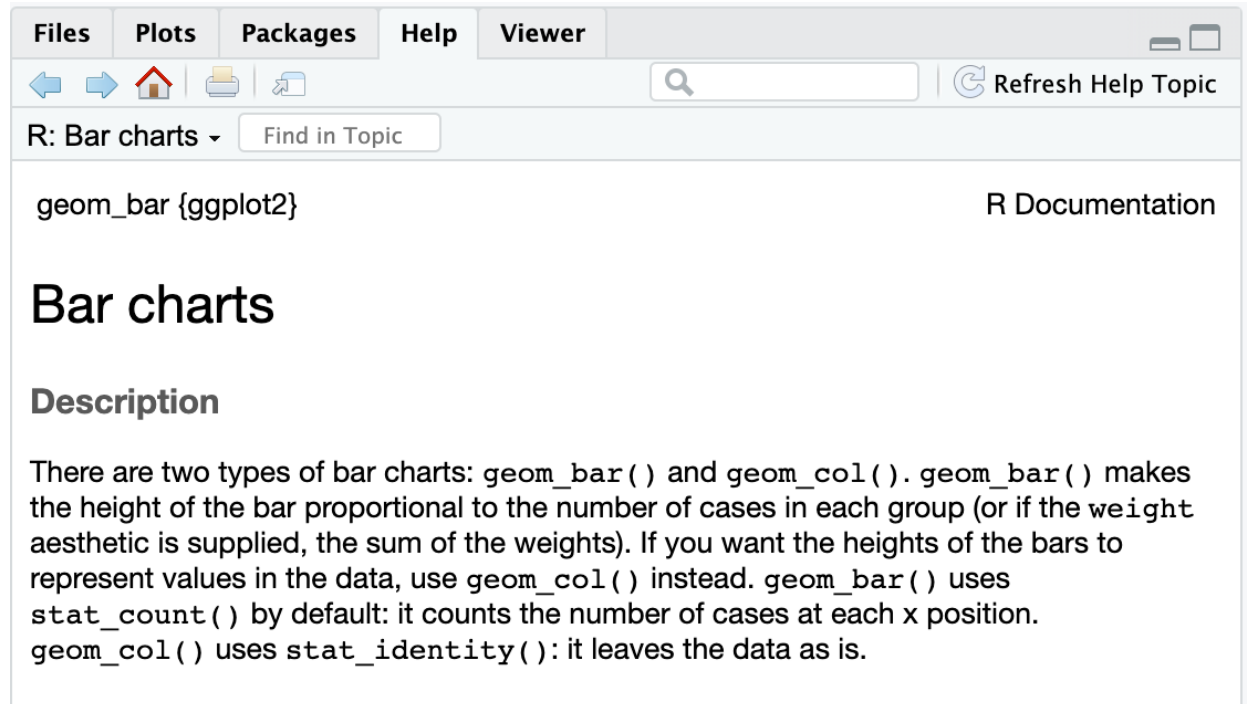
Help resources

Everyone makes mistakes when writing code—it's just part of the learning process. Fortunately, there are lots of helpful resources available in RStudio and online.

R documentation

R has built-in documentation for all functions and packages. To learn more about any R function, just run the code `?function_name`. For example, if you want to learn more about the geom_bar function, type:

```
?geom_bar
```

When you run the code, an entry on `geom_bar` appears in the Help viewer in the lower-right pane of your RStudio workspace. The entry begins with a Description section that discusses bar charts:

geom_bar {ggplot2}   R Documentation

Bar charts

Description

There are two types of bar charts: `geom_bar()` and `geom_col()`. `geom_bar()` makes the height of the bar proportional to the number of cases in each group (or if the `weight` aesthetic is supplied, the sum of the weights). If you want the heights of the bars to represent values in the data, use `geom_col()` instead. `geom_bar()` uses `stat_count()` by default: it counts the number of cases at each x position. `geom_col()` uses `stat_identity()`: it leaves the data as is.

The RDocumentation website contains much of the same content in a slightly different format, with additional examples and links.

ggplot2 documentation

The ggplot2 page, which is part of the official tidyverse documentation, is a great resource for all things related to ggplot2. It includes entries on key topics, useful examples of code, and links to other helpful resources.

Online search

Doing an online search for the error message you are encountering (and including "R" and the function or package name in your search terms) is another option. There's a good chance someone else has already encountered the same error and posted about it online.

The R community

If the other resources don't help, reach out to the Posit community online for advice about using RStudio. There are lots of useful online forums and websites:
- R for Data Science Online Learning Community
- Posit Community
- Stackoverflow
- Twitter (#rstats)