**Changing plot style and color**

So far we've covered how to create a variety of different plot types. Now let's learn how to customize them.

**Why customize?**

By default, Seaborn plots are pleasing to look at, but there are several reasons you may want to change the appearance. Changing the style of a plot can be motivated by personal preference, but it can also help improve its readability or help orient an audience more quickly to the key takeaway.

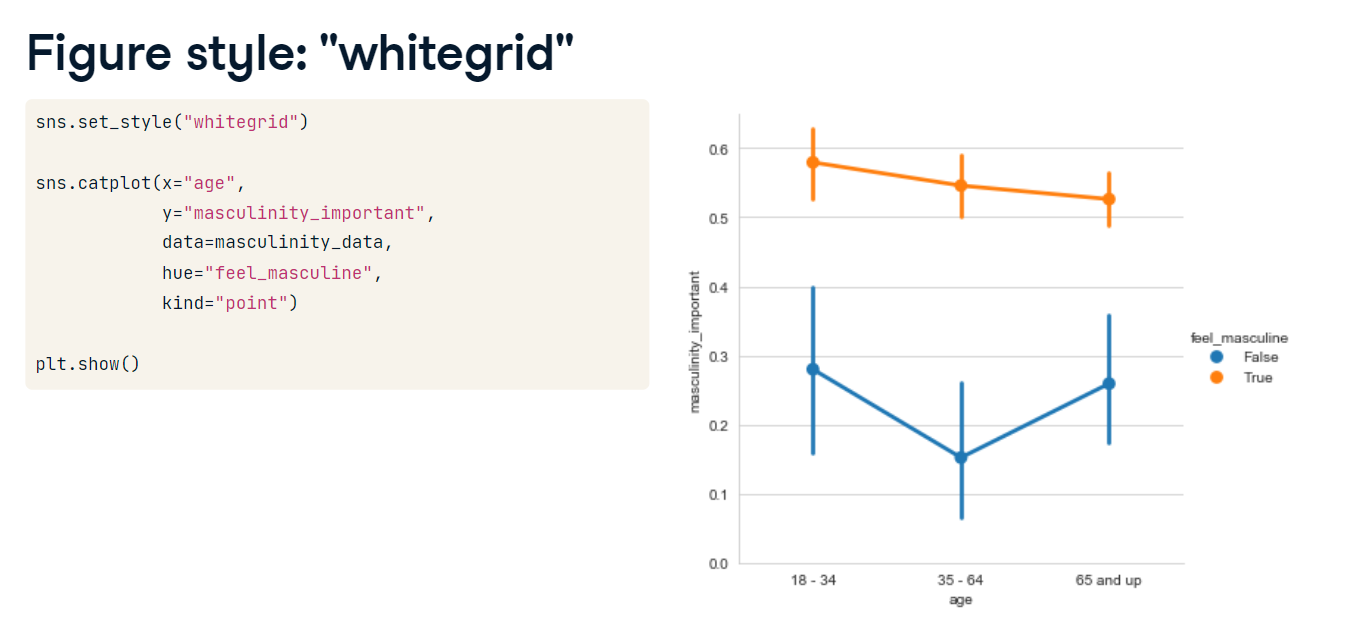
**Changing the figure style**

Seaborn has five preset figure styles which change the background and axes of the plot. You can refer to them by name: "white", "dark", "whitegrid", "darkgrid", and "ticks". To set one of these as the global style for all of your plots, use the "set style" function.

**Default figure style ("white")**

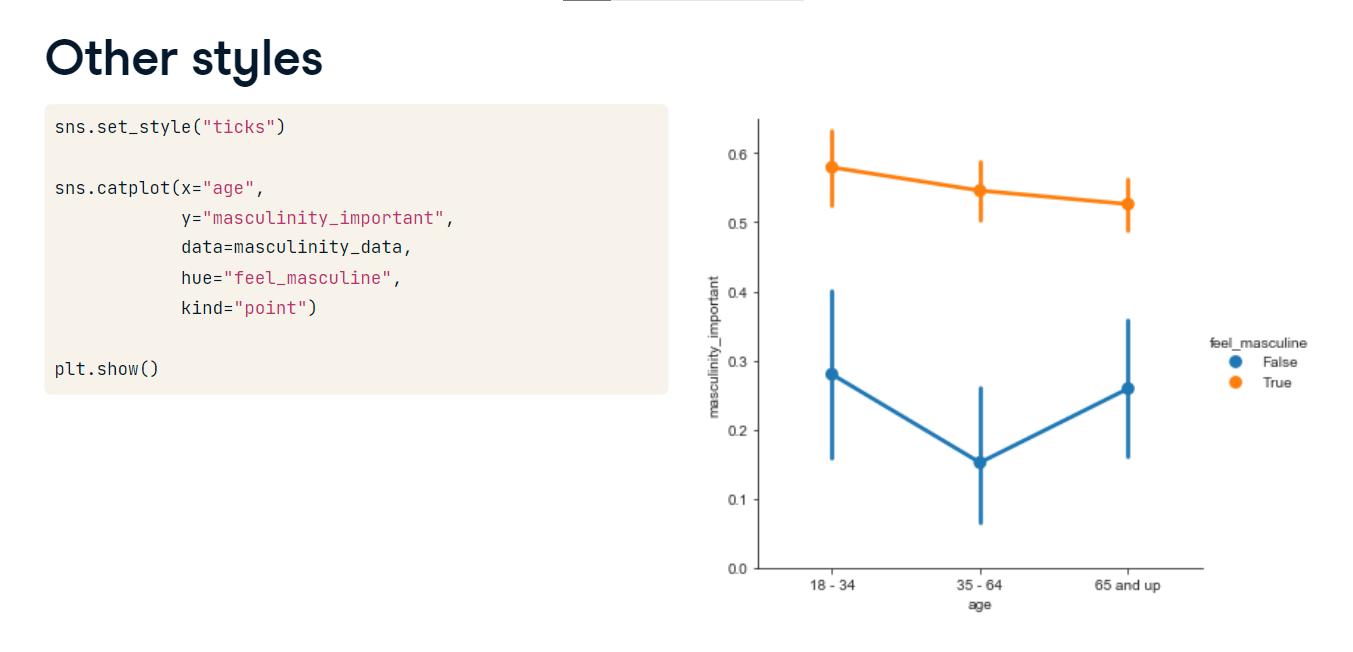
This is a plot we've seen before, showing the percentage of men reporting that masculinity was important to them, stratified by their age and whether or not they feel masculine. The default style is called "white" and provides clean axes with a solid white background. If we only care about the comparisons between groups or the general trend across age groups instead of the specific values, this is a good choice.

**Figure style: "whitegrid"**

Changing the style to "whitegrid" will add a gray grid in the background. This is useful if you want your audience to be able to determine the specific values of the plotted points instead of making higher level observations.  


**Other styles**

The other styles are variants on these. "ticks" is similar to "white", but adds small tick marks to the x- and y-axes."dark" provides a gray background,and "darkgrid" provides a gray background with a white grid.

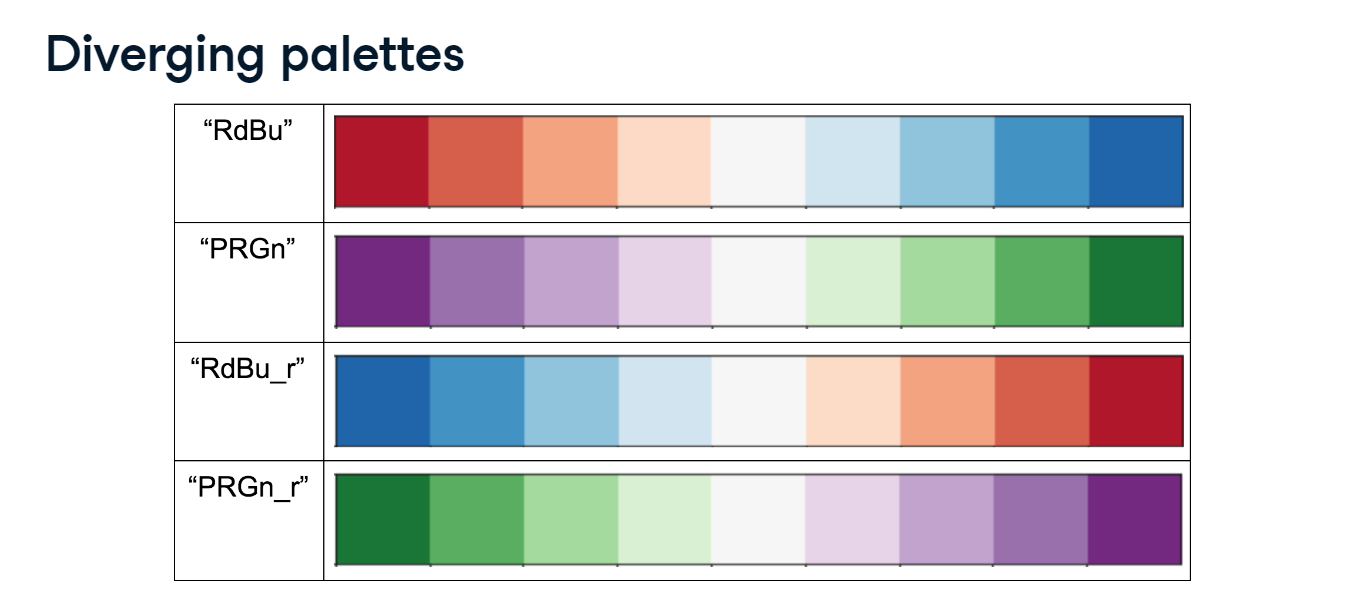


**Changing the palette**

You can change the color of the main elements of the plot with Seaborn's "set palette" function. Seaborn has many preset color palettes that you can refer to by name, or you can create your own custom palette. Let's see an example.

**Diverging palettes**

Seaborn has a group of preset palettes called diverging palettes that are great to use if your visualization deals with a scale where the two ends of the scale are opposites and there is a neutral midpoint. Here are some examples of diverging palettes - red/blue and purple/green. Note that if you append the palette name with "\_r", you can reverse the palette.

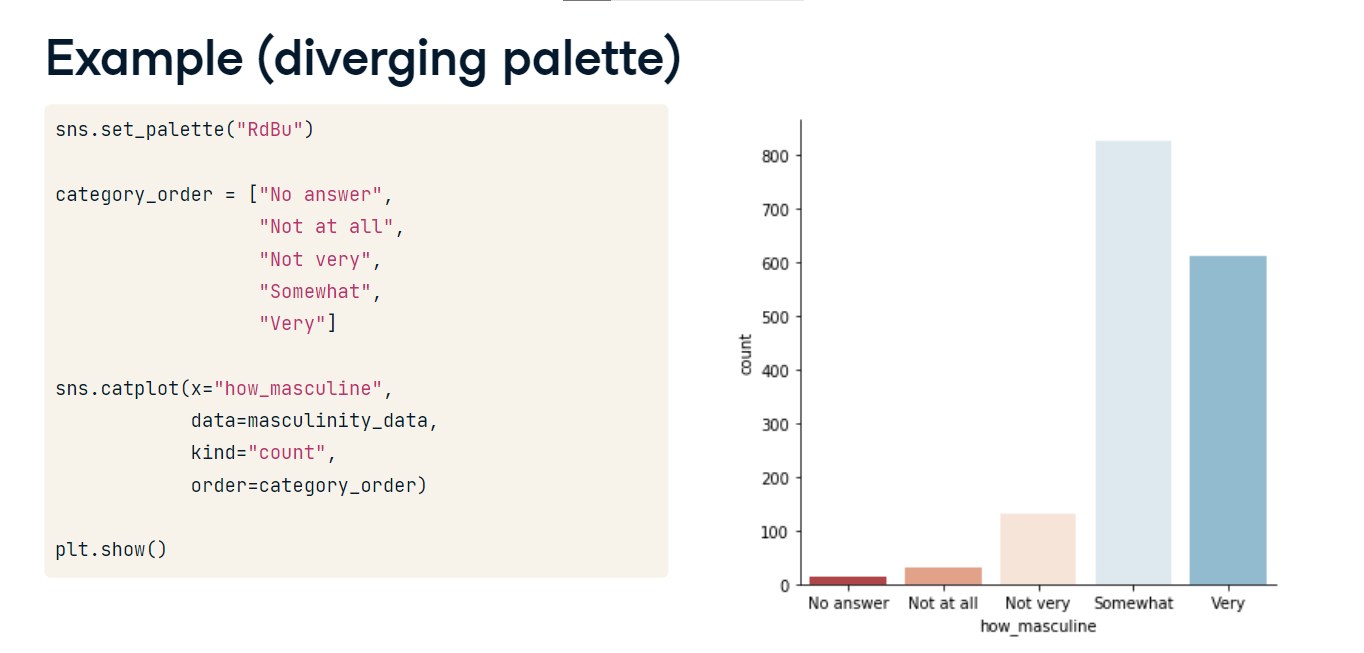


**Example (default palette)**

To see this in action, let's return to a count plot we've seen before of the responses of men reporting how masculine they feel.

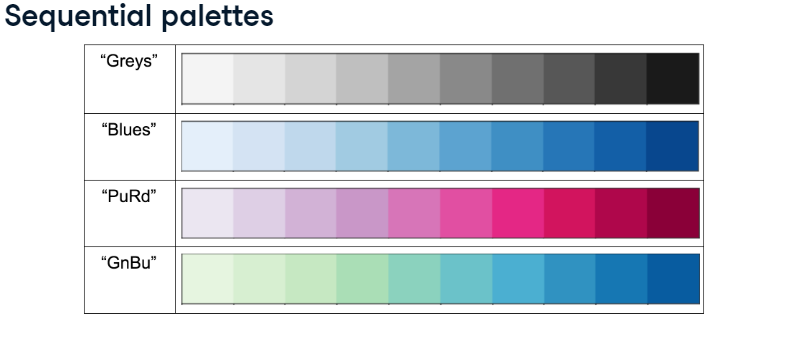
**Example (diverging palette)**

Setting this plot's palette to red/blue diverging provides a clearer contrast between the men who do not feel masculine and the men who do.



**Sequential palettes**

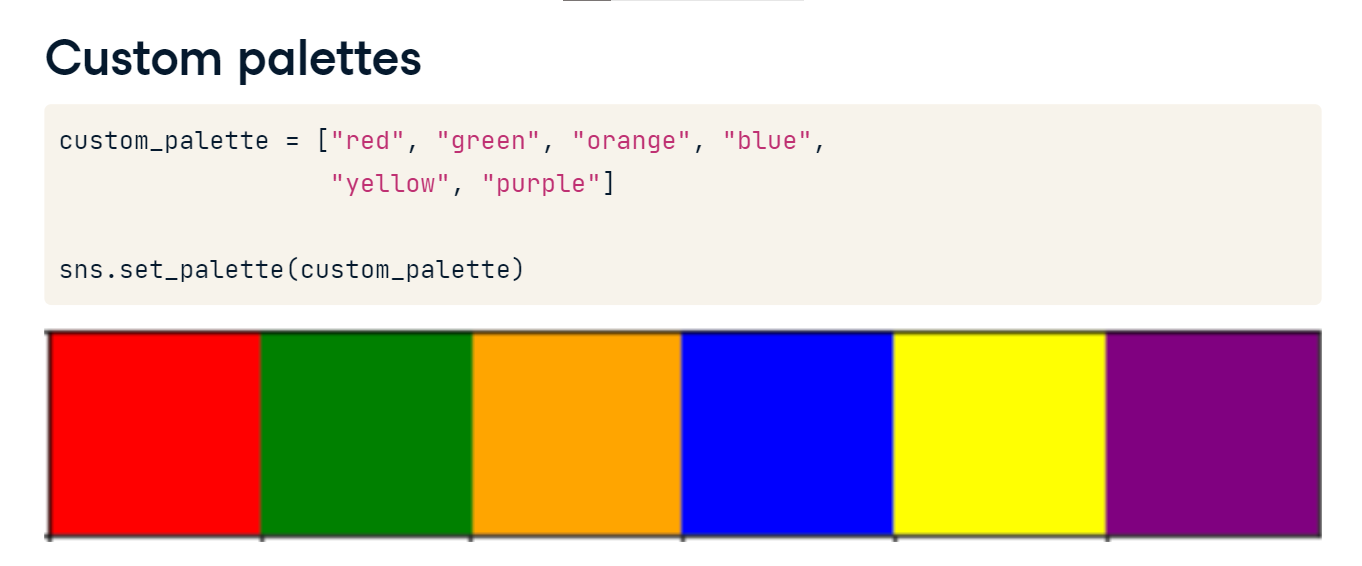
Another group of palettes are called sequential palettes. These are a single color (or two colors blended) moving from light to dark values.



**Sequential palette example**

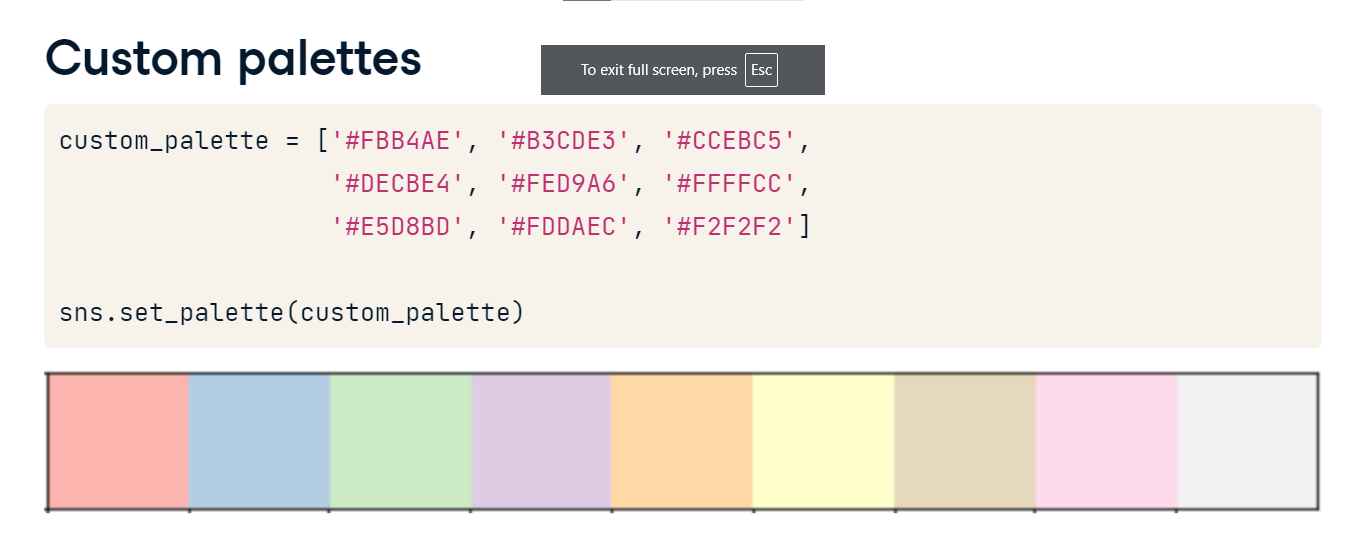
Sequential palettes are great for emphasizing a variable on a continuous scale. One example is this plot depicting the relationship between a car's horsepower and its miles per gallon, where points grow larger and darker when the car has more cylinders.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. <https://seaborn.pydata.org/>



**Custom palettes**

You can also create your own custom palettes by passing in a list of color names...or a list of hex color codes.



**Changing the scale**

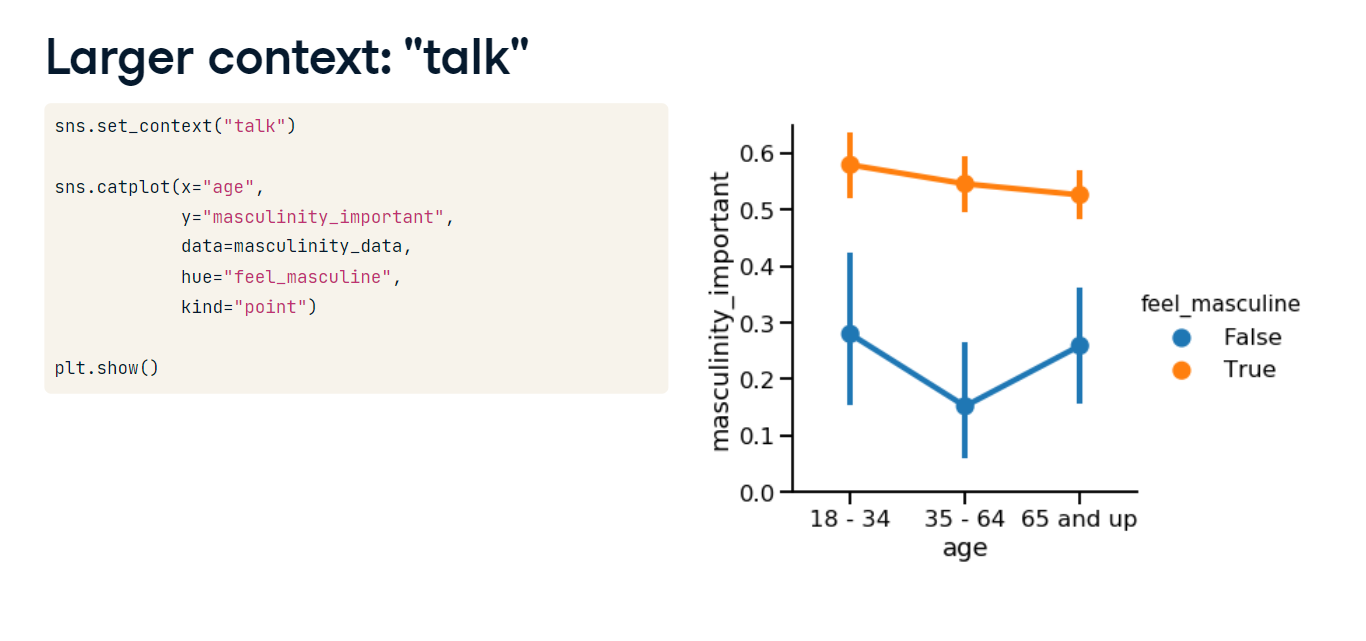
Finally, you can change the scale of your plot by using the "set context" function. The scale options from smallest to largest are "paper", "notebook", "talk", and "poster".

**Default context: "paper"**

The default context is "paper".

**Larger context: "talk"**

You'll want to choose a larger scale like "talk" for posters or presentations where the audience is further away from the plot.

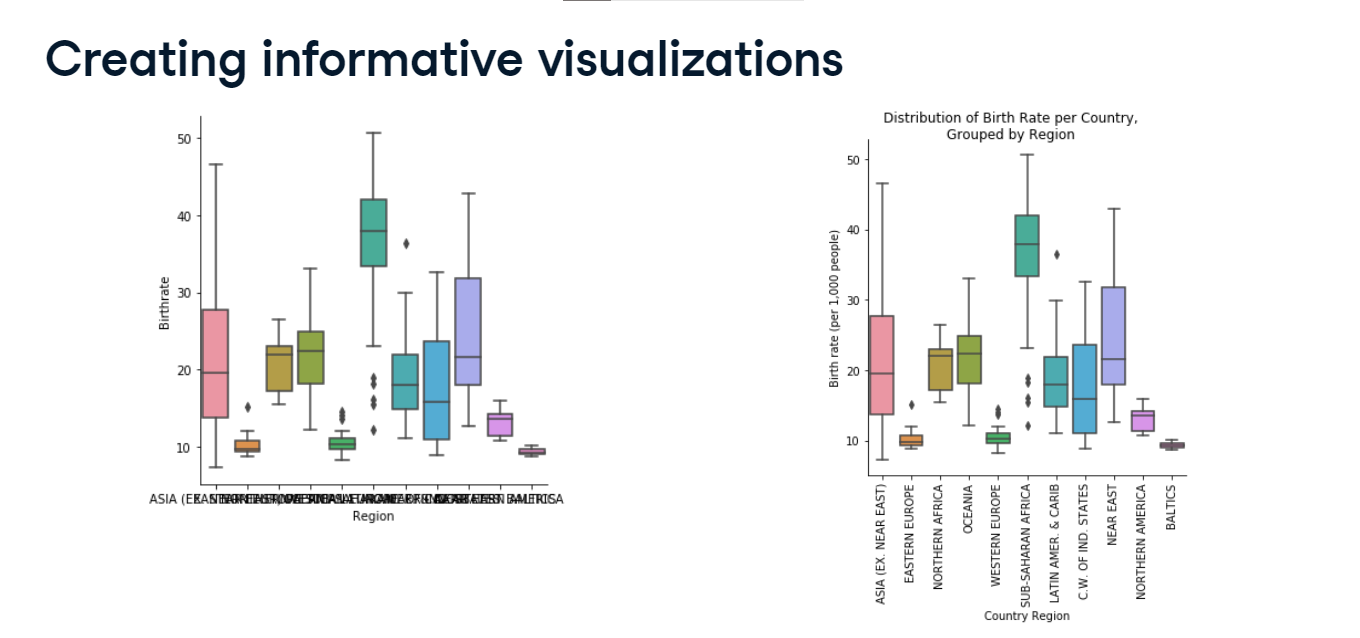


**Adding titles and labels: Part 1**

Welcome! In the next two lessons, we'll go over one of the most important parts of any data visualization: plot titles and axis labels.

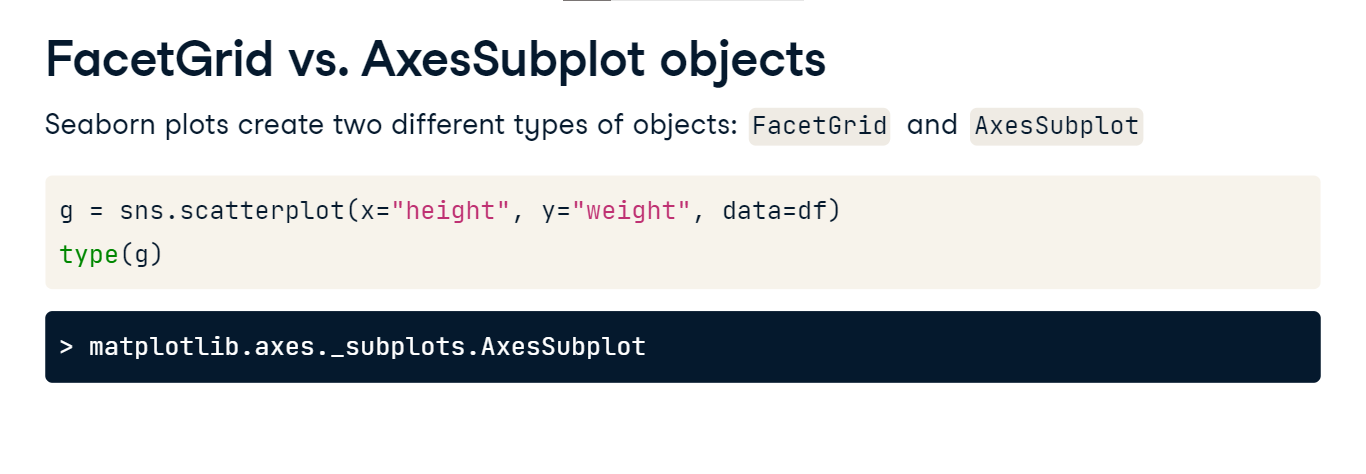
**Creating informative visualizations**

We create data visualizations to communicate information, and we can't do that effectively without a clear title and informative axis labels. To see this, let's compare two versions of the same visualization. On the left, we see box plots showing the distribution of birth rates for countries in each of 11 regions. On the right, we see the same visualization with three key modifications to make it easier to understand. A title is added, which immediately orients the audience to what they're looking at. The axis labels are more informative, making it clearer that birth rate is measured per one thousand people and birth rates are measured per country in each region. Finally, the x-axis tick labels are rotated to make it clear what each region is called. Let's learn how to make these changes.



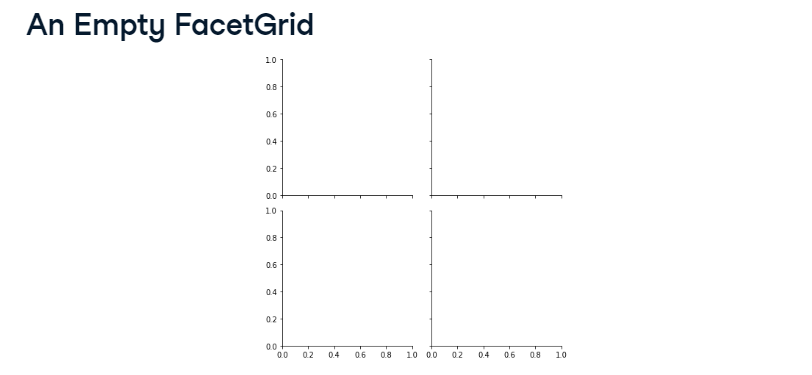
**FacetGrid vs. AxesSubplot objects**

Before we go into the details of adding a title, we need to understand an underlying mechanism in Seaborn. Seaborn's plot functions create two different types of objects: FacetGrids and AxesSubplots. To figure out which type of object you're working with, first assign the plot output to a variable. In the documentation, the variable is often named "g", so we'll do that here as well. Write "type" "g" to return the object type. This scatter plot is an AxesSubplot.



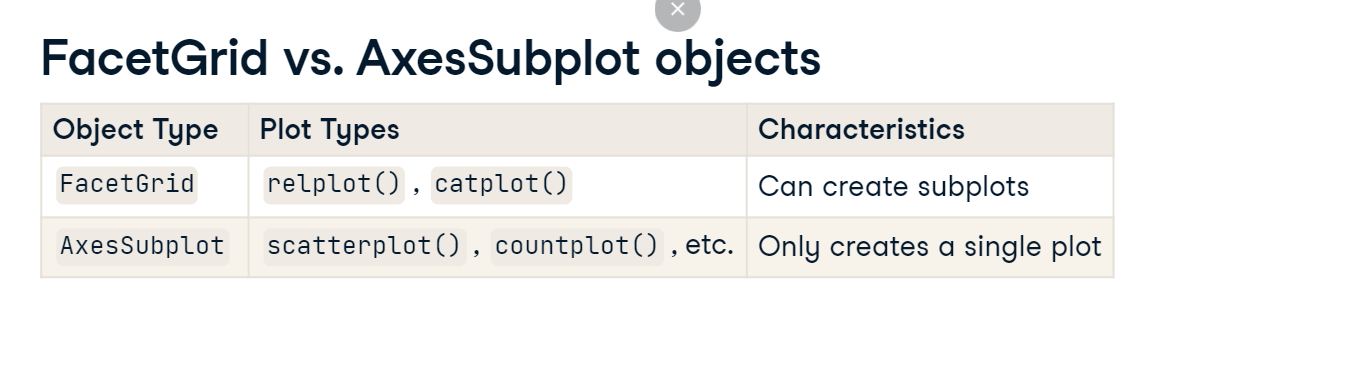
**An Empty FacetGrid**

A FacetGrid consists of one or more AxesSubplots, which is how it supports subplots.



**FacetGrid vs. AxesSubplot objects**

Recall that "relplot()" and "catplot()" both support making subplots. This means that they are creating FacetGrid objects. In contrast, single-type plot functions like "scatterplot()" and "countplot()" return a single AxesSubplot object.

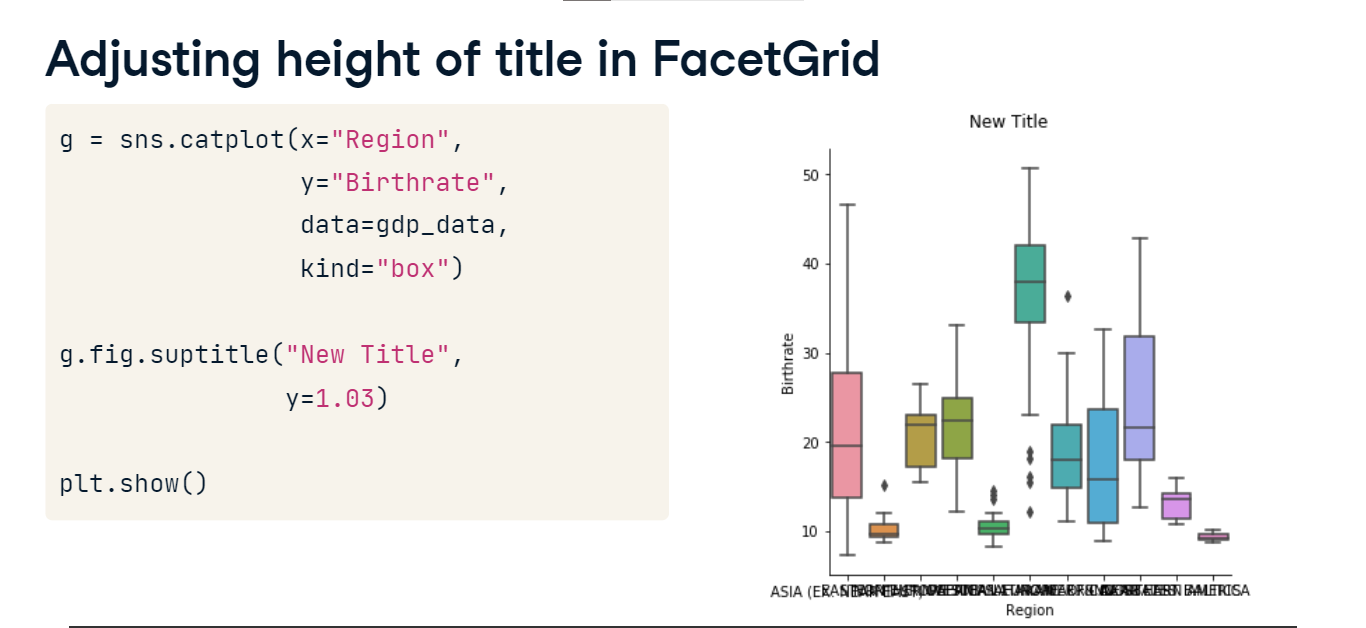


**Adding a title to FacetGrid**

Let's return to our messy plot from the beginning. Recall that "catplot()" enables subplots, so it returns a FacetGrid object. To add a title to a FacetGrid object, first assign the plot to the variable "g". After you assign the plot to "g", you can set the title using "g dot fig dot suptitle". This tells Seaborn you want to set a title for the figure as a whole.

**Adjusting height of title in FacetGrid**

Note that by default, the figure title might be a little low. To adjust the height of the title, you can use the "y" parameter. The default value is 1, so setting it to 1 point 03 will make it a little higher than the default.



**Adding titles and labels: Part 2**

Hello! In this lesson, we'll continue learning how to customize plot titles and axis labels.

**Adding a title to AxesSubplot**

In the last lesson, we learned how to add a title to a FacetGrid object using "g dot fig dot suptitle". To add a title to an AxesSubplot object like that from the "box plot" function, assign the plot to a variable and use “g dot set\_title”. You can also use the “y” parameter here to adjust the height of the title.



**Titles for subplots**

Now let's look at what happens if the figure has subplots. Let's say we've divided countries into two groups - group one and group two - and we've set "col" equal to "Group" to create a subplot for each group.

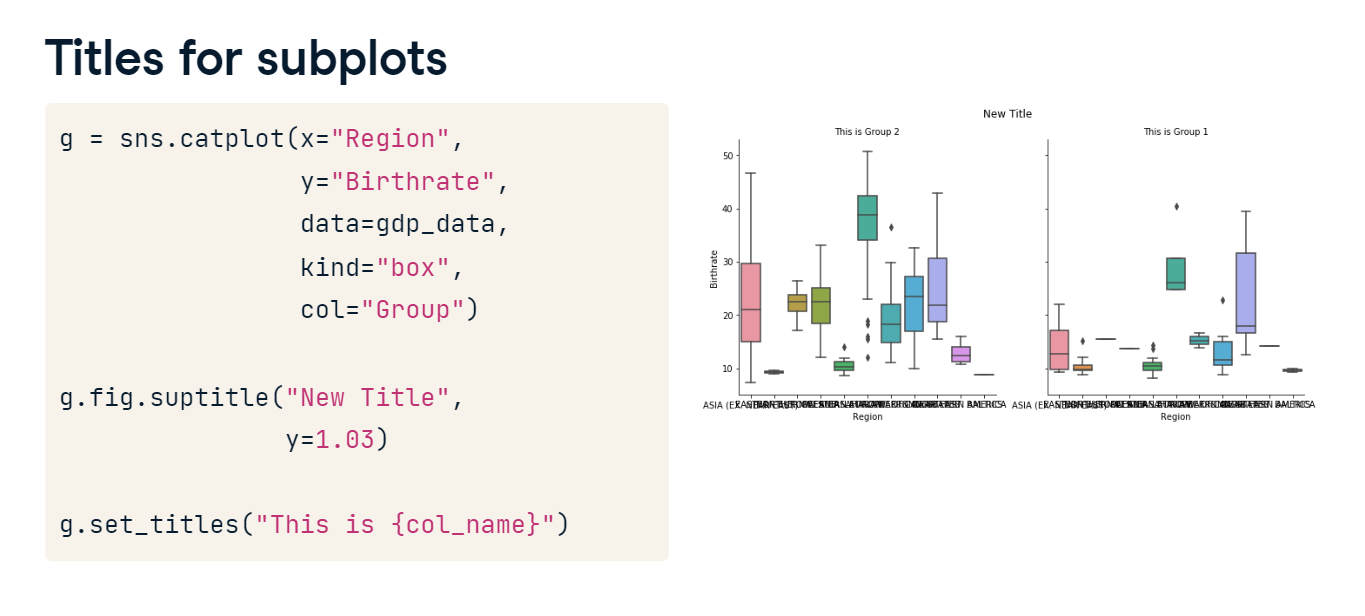
**Titles for subplots**

Since g is a FacetGrid object, using "g dot fig dot suptitle" will add a title to the figure as a whole.



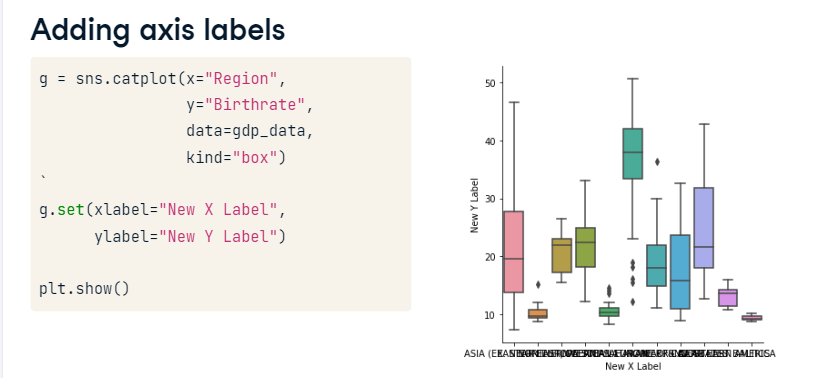
**Titles for subplots**

To alter the subplot titles, use "g dot set\_titles" to set the titles for each AxesSubplot. If you want to use the variable name in the title, you can use "col name" in braces to reference the column value. Here, we've created subplot titles that display as "this is group 2" and "this is group 1".



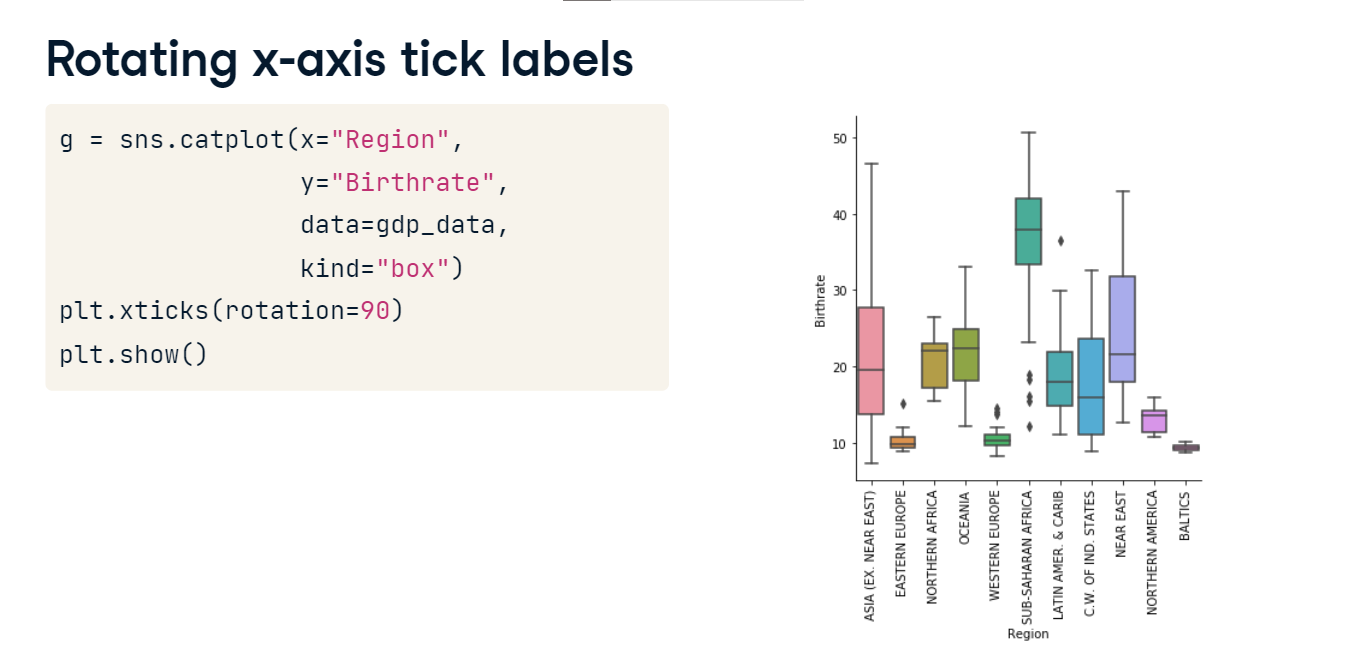
**Adding axis labels**

To add axis labels, assign the plot to a variable and then call the "set" function. Set the parameters "x label" and "y label" to set the desired x-axis and y-axis labels, respectively. This works with both FacetGrid and AxesSubplot objects.



**Rotating x-axis tick labels**

Sometimes, like in the example we've seen in this lesson, your tick labels may overlap, making it hard to interpret the plot. One way to address this is by rotating the tick labels. To do this, we don't call a function on the plot object itself. Instead, after we create the plot, we call the matplotlib function "plt dot xticks" and set "rotation" equal to 90 degrees. This works with both FacetGrid and AxesSubplot objects.



**Putting it all together**

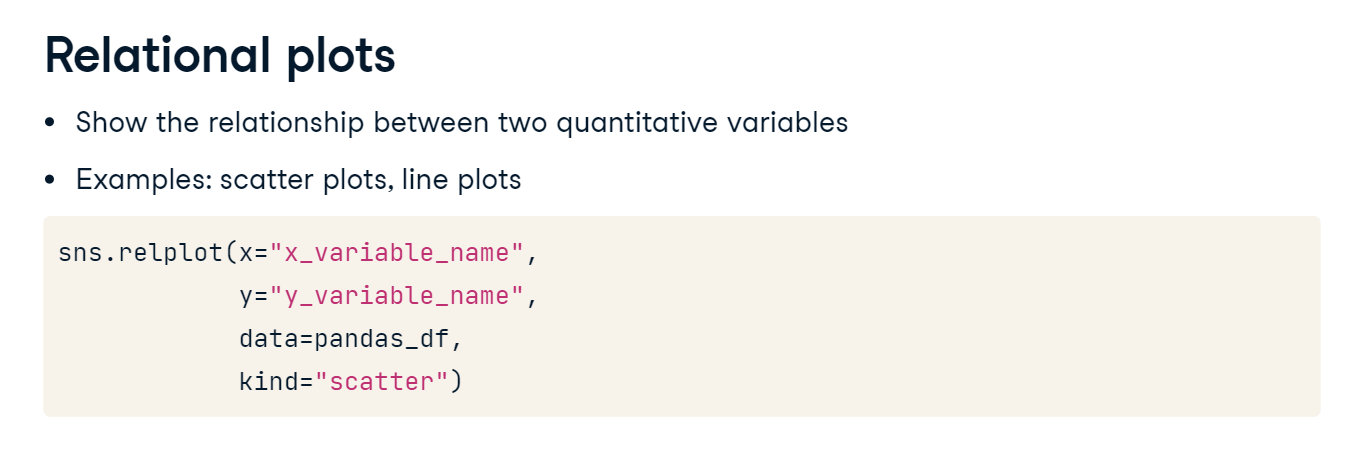
In this course, we've learned a great deal about how to create effective data visualizations in Seaborn. In this lesson, we'll review what we've learned and connect the pieces together to form a cohesive picture of how to use Seaborn for future projects.

**Getting started**

The first thing to recall is simply how to import Seaborn and its related library, Matplotlib. To do this, write "import seaborn as sns" and "import matplotlib dot pyplot as plt". Recall also that at the end of your data visualization code, you'll call "plt dot show" to show the visualization.

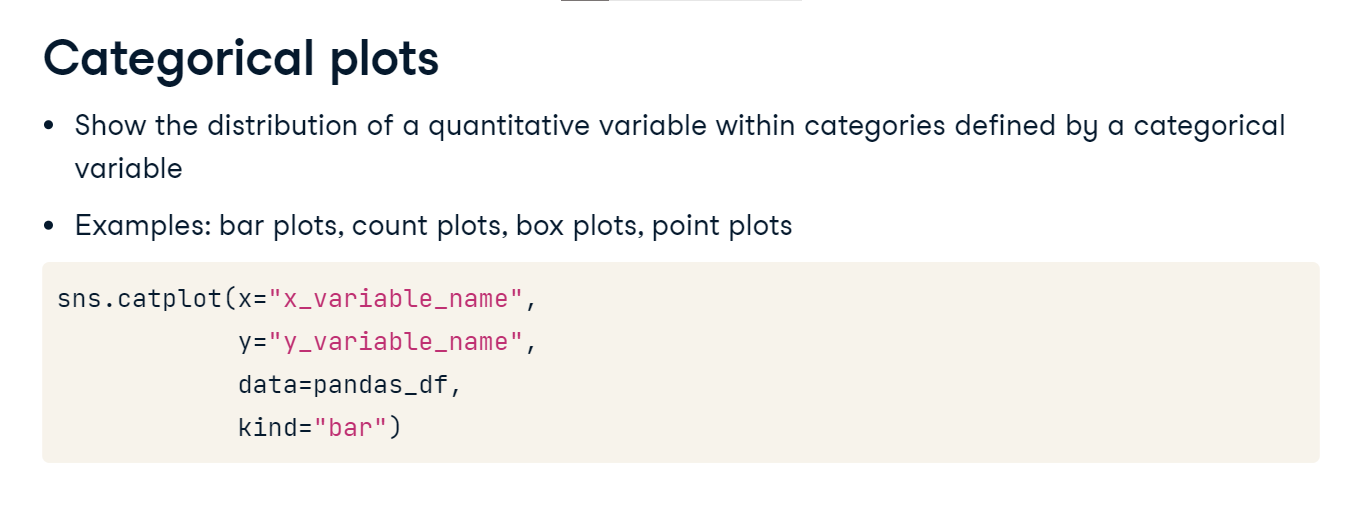
**Relational plots**

After you've imported the appropriate libraries, the next thing to do is to choose what type of plot you want to create. Relational plots are plots that show the relationship between two quantitative variables. Examples of relational plots that we've seen in this course are scatter plots and line plots. You can create a relational plot using "relplot()" and providing it with the x-axis variable name, y-axis variable name, the pandas tidy DataFrame, and the type of plot (either scatter or line).



**Categorical plots**

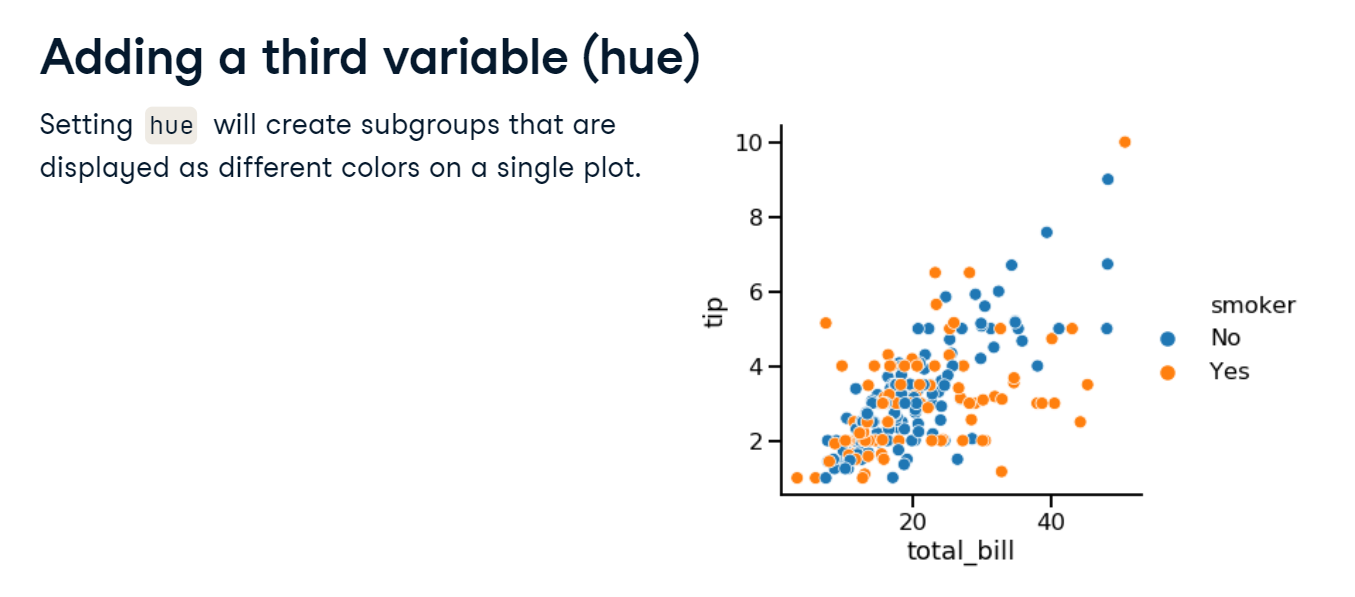
Categorical plots are another type of plot. These describe the distribution of a quantitative variable within categories given by a categorical variable. Examples of categorical plots we've seen are bar plots, count plots, box plots, and point plots. You can create a categorical plot using "catplot()" and providing it with the x-axis variable name, y-axis variable name (if applicable), the pandas tidy DataFrame, and the type of plot (either bar, count, box, or point).



**Adding a third variable (hue)**

If we want to add a third dimension to our plots, we can do this in one of two ways. Setting the "hue" parameter to a variable name will create a single plot but will show subgroups that are different colors based on that variable's values.

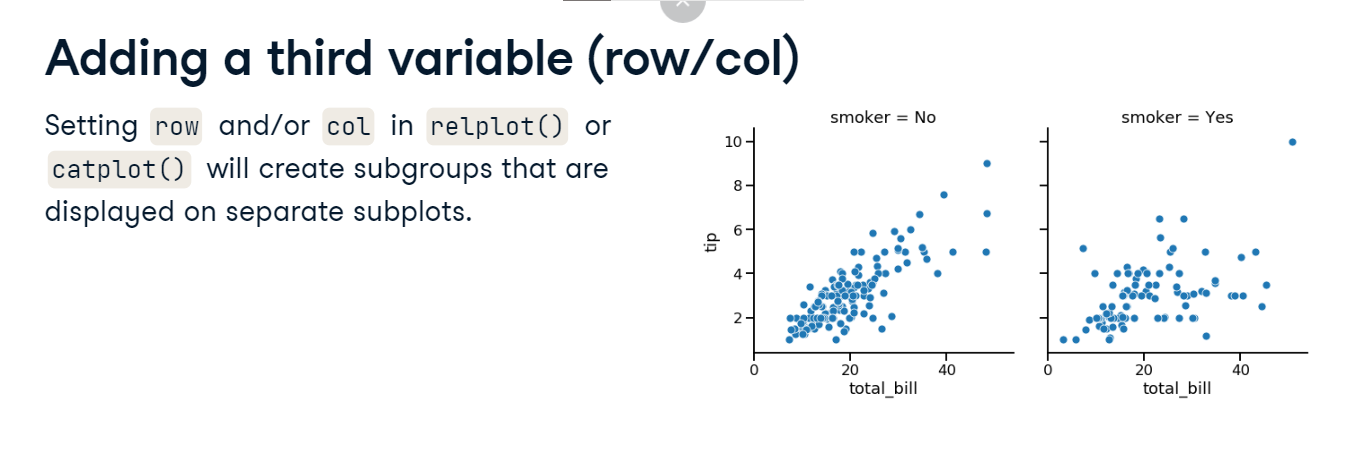
1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. <https://seaborn.pydata.org/>



**Adding a third variable (row/col)**

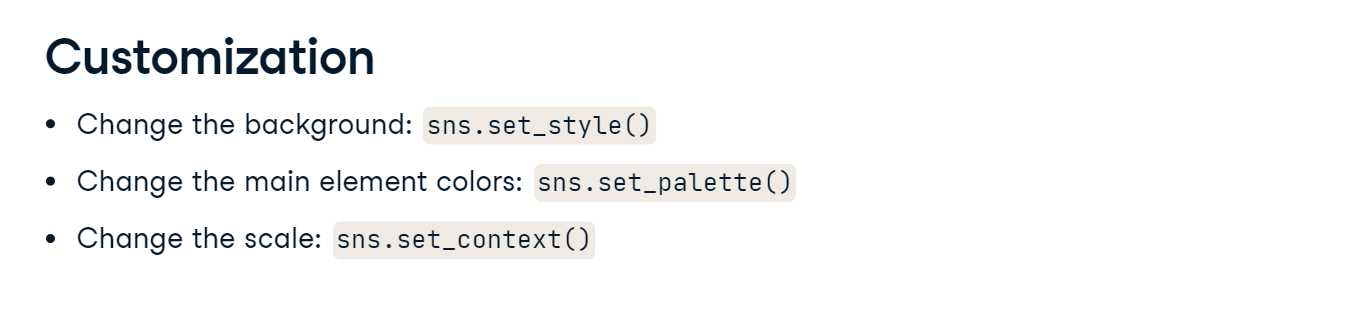
Alternatively, you can use "relplot()" and "catplot()"’s "col" and "row" parameters to graph each subgroup on a separate subplot in the figure.

1. 1 Waskom, M. L. (2021). seaborn: statistical data visualization. <https://seaborn.pydata.org/>



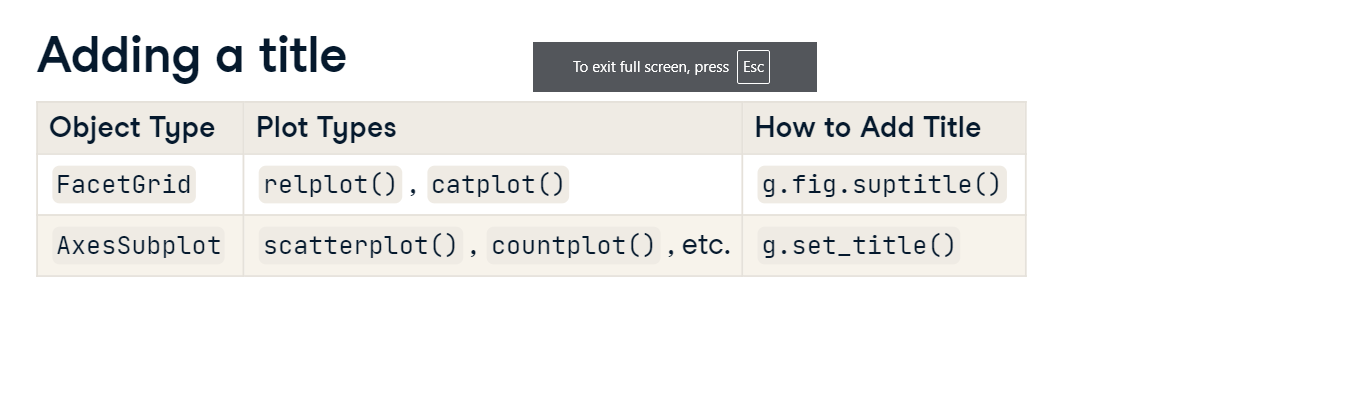
**Customization**

Once you have the basic plot created, you might want to customize the plot's appearance to improve its readability. You can change the background of the plot using "set\_style", the color of the main elements using "set\_palette", and the scale of the plot using "set\_context".



**Adding a title**

Finally, every plot should be given an informative title and axis labels. Recall the two types of plot objects - FacetGrids and AxesSubplots - and the way to add a title to each of them.



**Final touches**

Also recall how to use the "set" function with the "xlabel" and "ylabel" parameters to provide custom x- and y-axis labels, and how to use "plt.xticks" with the "rotation" parameter to rotate the x-tick labels.

