d3.transition

We learn about d3.transition, and talk about when we would use it instead of CSS transitions. We then update our histogram to animate using d3.transition.

Contents
d3.transition
Final code for this lesson

CSS transitions have our back for simple tasks, but for more complex animations we'll need to use d3.transition() from the d3.transition module. When would we want to use d3.transition() instead of CSS transitions?

- When we want to ensure that multiple animations line up
- When we want to change attributes when the animation ends (for example starting another animation)
- When the property we want to animate isn't a CSS property (remember when we tried to animate text fill but forgot that fill is not a CSS property, but d3.transition can animate non-CSS property changes)
- When we want to chain multiple animations and removing elements with animations
- When we might interrupt halfway through a transition
- When we want a custom animation (for example, we could write a custom interpolator for changing text that adds new words to a given string)

Let's get our hands dirty by re-implementing the CSS transitions for our histogram.

Let's again start by animating any changes to our bars. Instead of adding a transition property to our stylize.csv file, we'll start in the chart.js file where we set our .barRects attributes.

As a reminder, what we can

we're creating a d3 selection object that contains all .rects elements. Let's log that to the console as a reminder of what a selection object looks like:

```
d3.select("svg").select("g").select("g").select("g
```