12.2.2 Practicing JavaScript Methods

Roza feels more comfortable with using JavaScript methods to manipulate data, but would like to consolidate her skills in building an actual plot. Instead of jumping into the belly button dataset, she would like to practice creating a simpler graph in Plotly that requires some behind-the-scenes JavaScript action. She will create a bar chart of five cities whose populations have seen the greatest increase in the period 2016–2017.

We will help her from beginning to end. Are you ready?

The first thing we need to do is to obtain the dataset, which can be downloaded using the link below.

Download the dataset (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/module_12/data-12-2-2-resources.zip)

Now let's set up our `index.html` to use Plotly. Create `index.html` and paste the following code into it.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Fast-Growing Cities 2016-2017</title>
  <script src="https://cdn.plot.ly/plotly-latest.min.js"></script>
</head>
<body>
  <div id="bar-plot"></div>
  <script src="data.js"></script>
  <script src="plot.js"></script>
</body>
</html>
```

Note the following:

- The `<title>` tag is appropriate to the chart Roza is creating: "Fast-Growing Cities 2016–2017."
- The first `<script>` tag links to the Plotly CDN.
- The `<div>` tag, where the Plotly chart will be displayed, is titled "bar-plot."
- The second `<script>` tag links to the data file, `data.js.`
- The third `<script>` tag links to the script file for the chart, i.e., the JavaScript code.

IMPORTANT

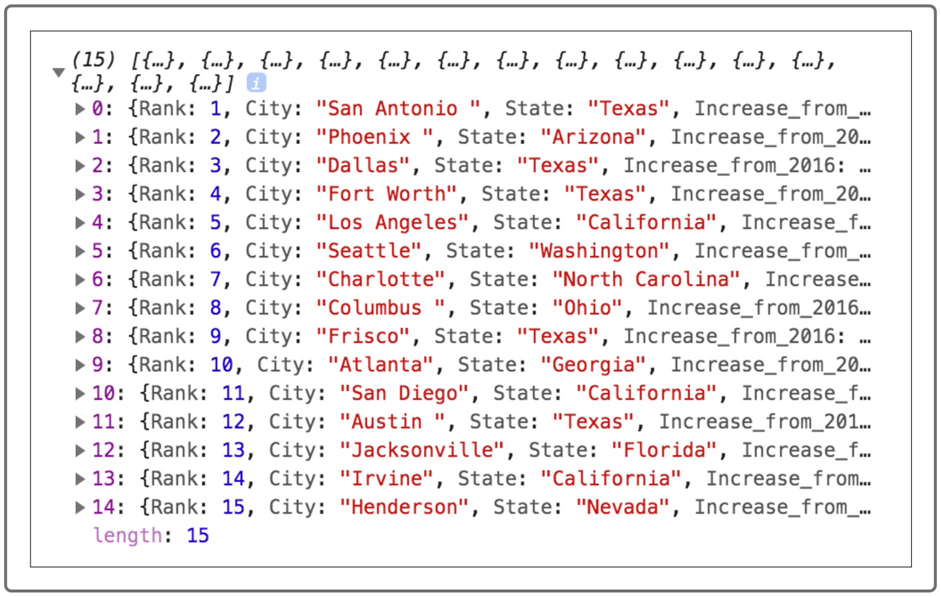
> Always make sure that your files are linked correctly and in the right order. Since Roza needs the Plotly CDN to use Plotly, the CDN `<script>` tag should come first. Since she will also need the data to load before using Plotly, the data file should be linked next. Finally, the script file is linked.

Now take a moment to examine the dataset, contained in `data.js`, of which you have seen a sample.

```
var cityGrowths = [
  {
    "Rank": 1,
    "City": "San Antonio ",
    "State": "Texas",
    "Increase_from_2016": "39208",
    "population": "1511946"
  },
```

The variable `cityGrowths` is assigned to an array of objects, each of which contains a city and some associated facts, such as its population. This array, although contained in a separate file, can be accessed from `plot.js` so create a blank file named `plot.js` now. To verify that the data is correctly read in, Roza should call `console.log(cityGrowths);` in `plot.js`.

Here's the result in the browser console:



```
▼(15) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…},
 {…}, {…}, {…}]
  ▶0: {Rank: 1, City: "San Antonio ", State: "Texas", Increase_from_…
  ▶1: {Rank: 2, City: "Phoenix ", State: "Arizona", Increase_from_20…
  ▶2: {Rank: 3, City: "Dallas", State: "Texas", Increase_from_2016: …
  ▶3: {Rank: 4, City: "Fort Worth", State: "Texas", Increase_from_20…
  ▶4: {Rank: 5, City: "Los Angeles", State: "California", Increase_f…
  ▶5: {Rank: 6, City: "Seattle", State: "Washington", Increase_from_…
  ▶6: {Rank: 7, City: "Charlotte", State: "North Carolina", Increase…
  ▶7: {Rank: 8, City: "Columbus ", State: "Ohio", Increase_from_2016…
  ▶8: {Rank: 9, City: "Frisco", State: "Texas", Increase_from_2016: …
  ▶9: {Rank: 10, City: "Atlanta", State: "Georgia", Increase_from_20…
  ▶10: {Rank: 11, City: "San Diego", State: "California", Increase_f…
  ▶11: {Rank: 12, City: "Austin ", State: "Texas", Increase_from_201…
  ▶12: {Rank: 13, City: "Jacksonville", State: "Florida", Increase_f…
  ▶13: {Rank: 14, City: "Irvine", State: "California", Increase_from…
  ▶14: {Rank: 15, City: "Henderson", State: "Nevada", Increase_from_…
   length: 15
```

Now take a moment to plan our next steps. We will need to:

1. Sort the cities in descending order of population growth.
2. Select only the top five cities in terms of growth.
3. Create separate arrays for the city names and their population growths.
4. Use Plotly to create a bar chart with these arrays.

Sort and Select the Cities

The first step is to sort the cities by population growth. For this she will use the `sort()` method, which in turn will call an anonymous function to sort objects by the `Increase_from_2016` property.

The next step is to select only the top five cities by population growth. We'll use `slice()` to perform this task.

Create Arrays of City Names and Growth Figures

Now we need to create two arrays: an array of city names, and an array of corresponding population growths. We'll use the `map()` method to extract these properties. These arrays will be the `x` and `y` axis data of the Plotly chart.

Suppose that Roza wants to use `map()` to create a separate array of the top five city names, as well as the top five growth figures. How would we help her do this?

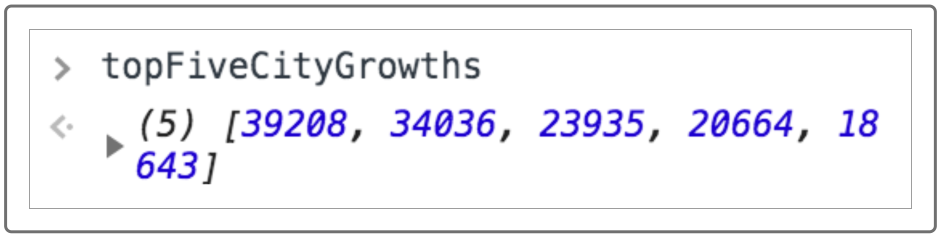
Add the following JavaScript code from `plot.js`.

```
var topFiveCityNames = cityGrowths.map(city => city.City);
var topFiveCityGrowths = cityGrowths.map(city => parseInt(city.Increase_from
```

Did you notice the `parseInt()`method used above? You might have also noticed that all values in the dataset are strings. For example, in this object, the numerical values are formatted as strings, as indicated by the quotation marks, rather than integers.

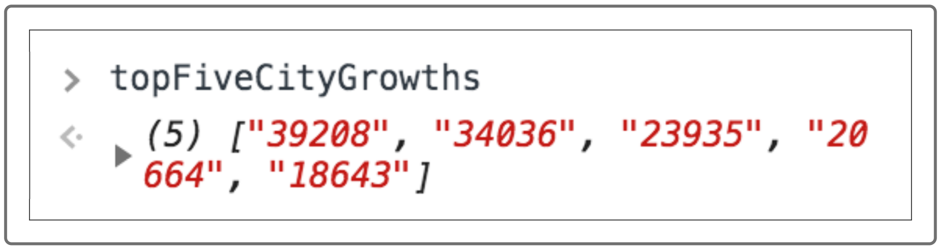
```
{
    "Rank": 2,
    "City": "Phoenix",
    "State": "Arizona",
    "Increase_from_2016": "34036",
    "population": "1626078"
  },
```

That is why `parseInt(city.Increase_from_2016)`converts strings into integers. While JavaScript is flexible enough to interpret numbers enclosed in quotation marks as numbers, it's good practice to explicitly transform, or cast, strings into numbers. Below is the array that is returned when `parseInt()` is called:



```
> topFiveCityGrowths
<· ▶ (5) [39208, 34036, 23935, 20664, 18
  643]
```

And this is the array that is returned when `parseInt()` is not called:



```
> topFiveCityGrowths
<· ▸ (5) ["39208", "34036", "23935", "20664", "18643"]
```

Create a Bar Chart with the Arrays

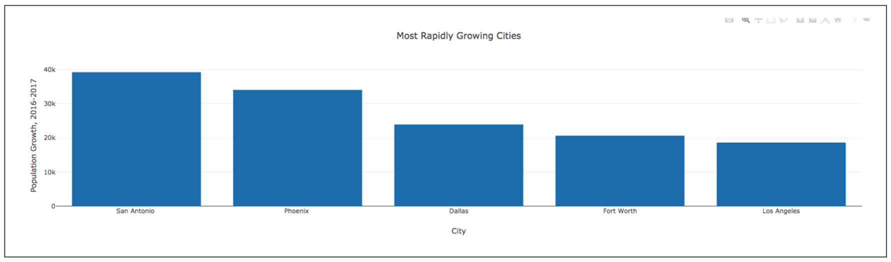
The final task is to render these arrays in Plotly. Add the following code to `plot.js`.

```
var trace = {
  x: topFiveCityNames,
  y: topFiveCityGrowths,
  type: "bar"
};
var data = [trace];
var layout = {
  title: "Most Rapidly Growing Cities",
  xaxis: {title: "City" },
  yaxis: {title: "Population Growth, 2016-2017"}
};
Plotly.newPlot("bar-plot", data, layout);
```

Here's what's happening in this code:

- The `trace` specifies the type of graph as a bar chart as well as defines the `x-` and `y`-axis data.
- The variable `data` encloses `trace` in an array to meet Plotly's format requirement.
- The variable `layout` is assigned to an object that specifies the chart's title and axis labels.
- Finally, the graph is rendered with `Plotly.newPlot()`.

This is the resulting chart:



Well done!

SKILL DRILL

Use the same dataset to create a bar chart of the seven largest cities by population.