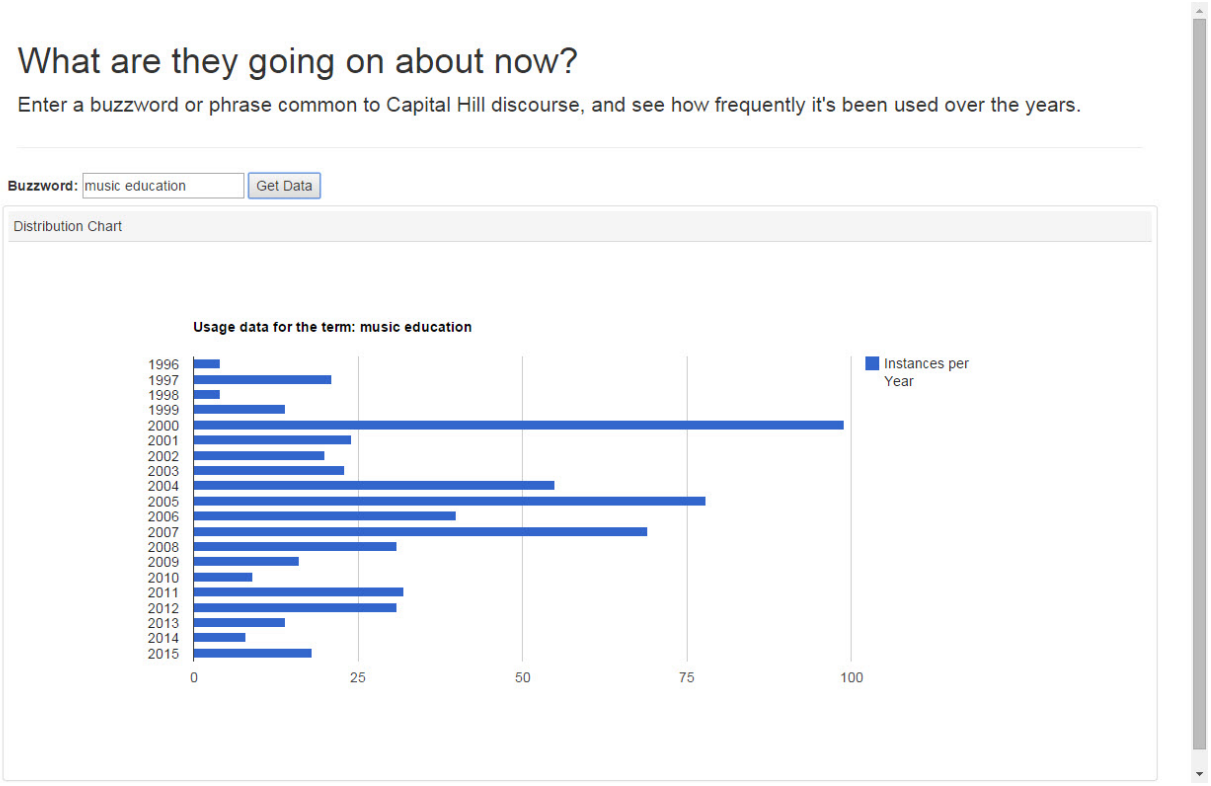
RIVERSIDE JS WORKSHOP

Language-Graph Coding Sprint Challenge
June, 2015
RiversideJS.net

THE CHALLENGE

Working in 3-person teams you will build a simple application that creates a frequency-distribution graph for any given word used in Congress over some period of time. To make this work, you'll need three things: a simple form to input a word or phrase, a data source to provide word-usage data, and a mechanism for graphing the returned data on the page.

Here's a usage scenario. A user enters a term or phrase in the form, let's say *music education*, and hits the submit button. The application retrieves data from an API and then generates a graph showing how many times the term, *music education* in this case, was used in Congress each of the last 20 years. Here is a screen shot of what it might look like:



YOUR RESOURCES

The resources you'll need to complete this challenge can be found in two places:

- Language usage data can be gotten from the Capitol Words API provided by the Sunlight Foundation. You will need a key to access the API which you sign up for here.
- Graphing can be done using Google's Visualization API. For those who have never used it, Google Visualization is a powerful, easy-to-use library for creating all kinds of charts and graphs. For this sprint a simple bar chart is sufficient to complete the challenge. There is also a Quickstart guide that may be useful.

There is no restriction on using other libraries or frameworks that help your team complete the challenge. Here are script tags for jQuery and Bootstrap if you choose to use either:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js"></script>
```

DETERMINING THE WINNER

At the time this was written a suitable winning prize had not yet been secured. However, the winning team will be determined by popular vote, subsequently crowned in laurel wreaths and hoisted high on the shoulders of their colleagues -- figuratively speaking of course. Criteria for voting includes

- State of completion: is it done?
- Functionality: does it do what it's supposed to?
- Completion time: in the event of identical solutions, who finished first?
- Bells and Whistles: are there additional features which make it extra cool?

TIPS

Time is the killer here. This challenge could be completed in less than 60 minutes by an individual, if s/he is either wickedly smart or has previous experience using Google Visualization. Most won't be able to, though. (It took me 90 minutes.) Teamwork is the key to success. Divide tasks wisely. Good luck!