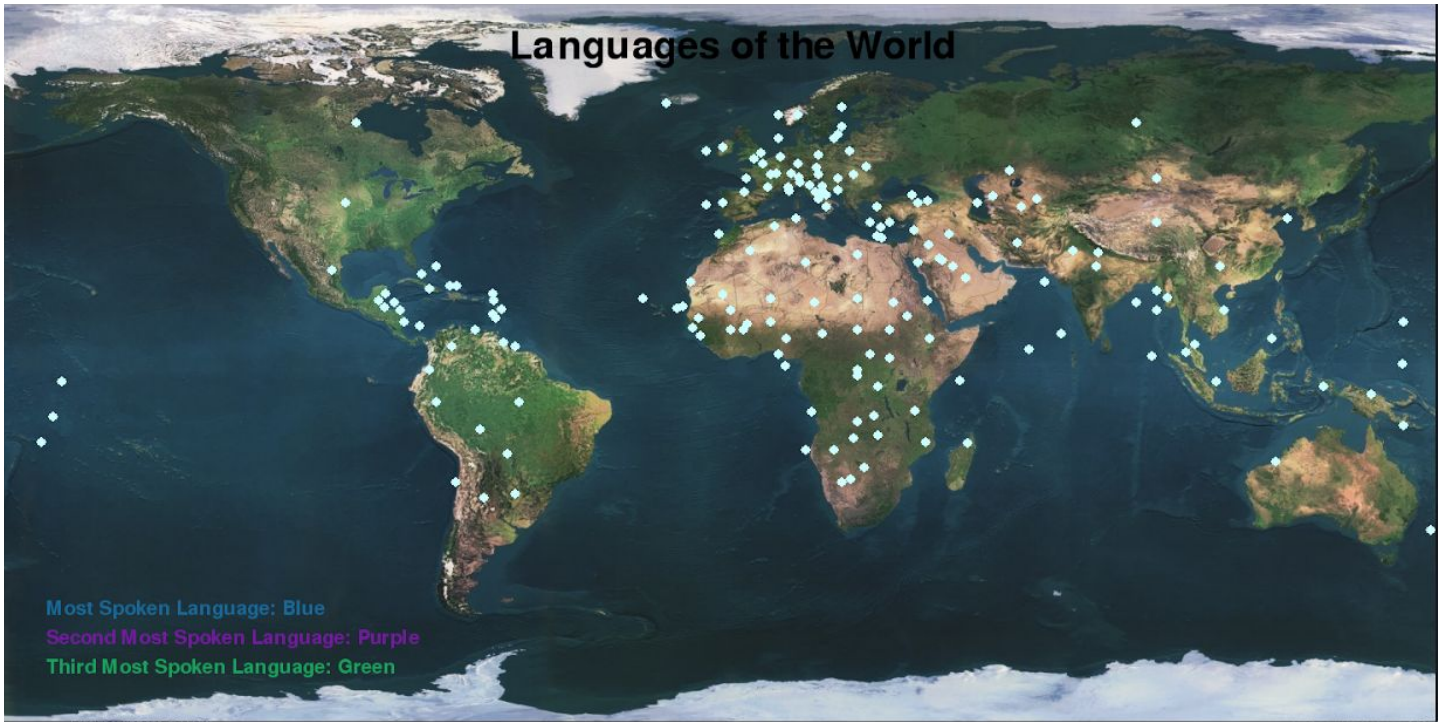
Mapping the Languages of the World

Project Overview

This data visualization is an interactive map that allows the user to learn more about the languages of the world. The user can click on the points at the center of the country and an information box appears with the country's name and the top languages spoken in that country. By using longitudinal and latitudinal data were were able to convert country locations onto a flat projection and create an interactive map.

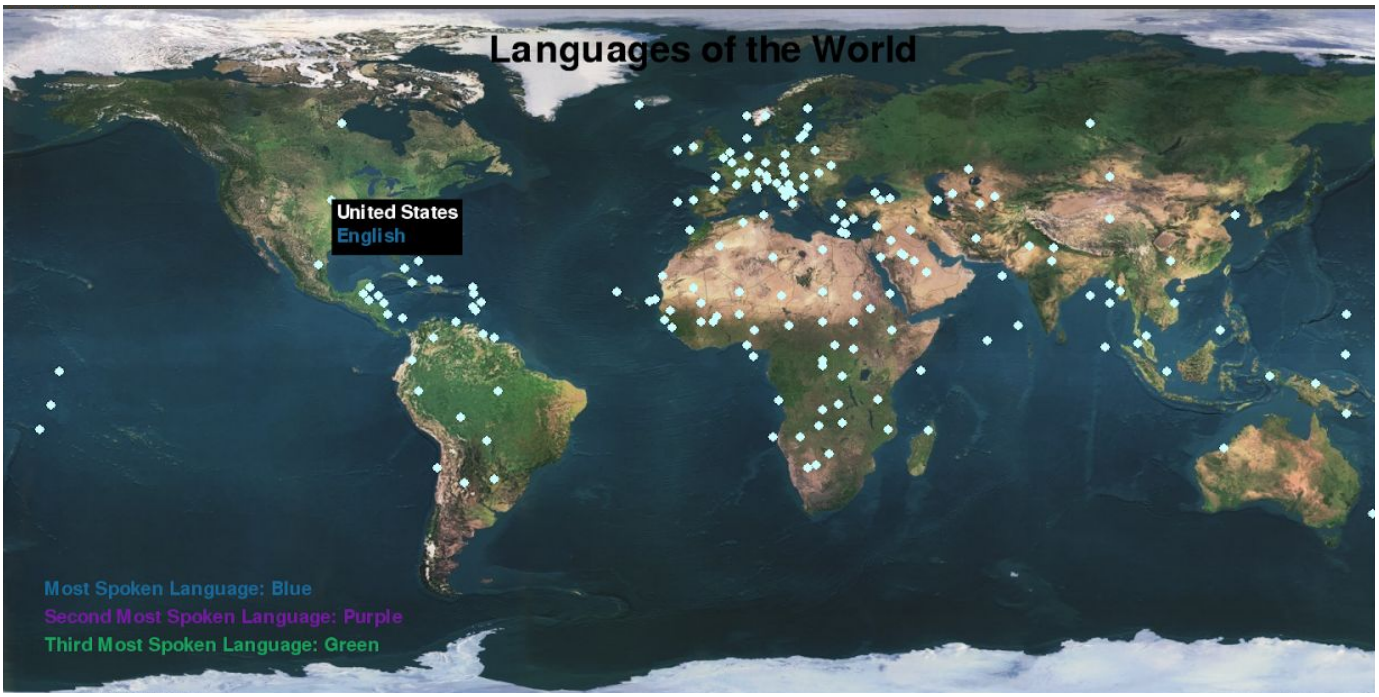
Results

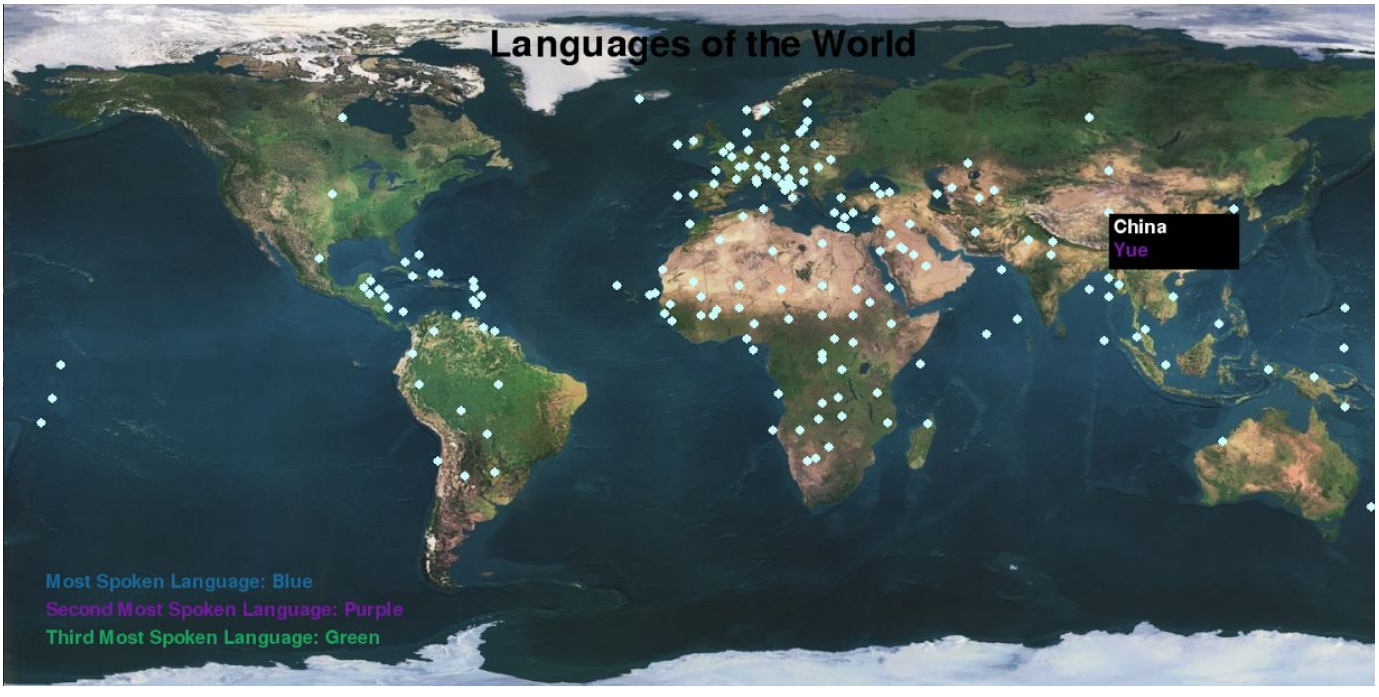
When we began this project, we thought about how it was difficult to know the names of the languages in each country. Some are intuitive, such as Finnish for Finland, but others like Dutch for Belgium are not. Moreover, most countries have more than one common language. We decided to make an interactive data visualization about these facts, resulting in the map that is pictured below.

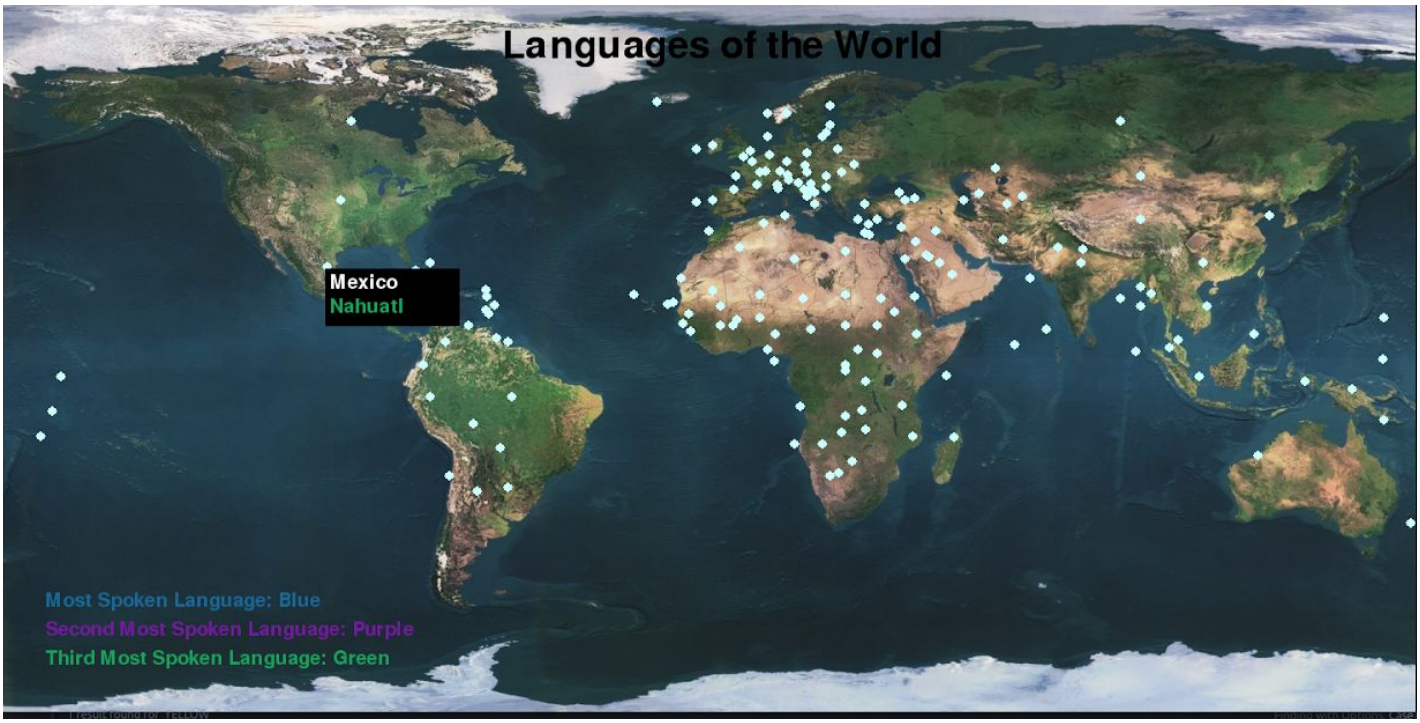


The way that map works is that first a user presses key 1, key 2, or key 3 and then clicks on a point on the map. Once clicked a text box will appear, indicating what the country on the map is, and if the key pressed was not 2 or 3, what the most spoken

language in that country is. If the key pressed was 2 or 3, then the second most spoken language and third most spoken language are shown.







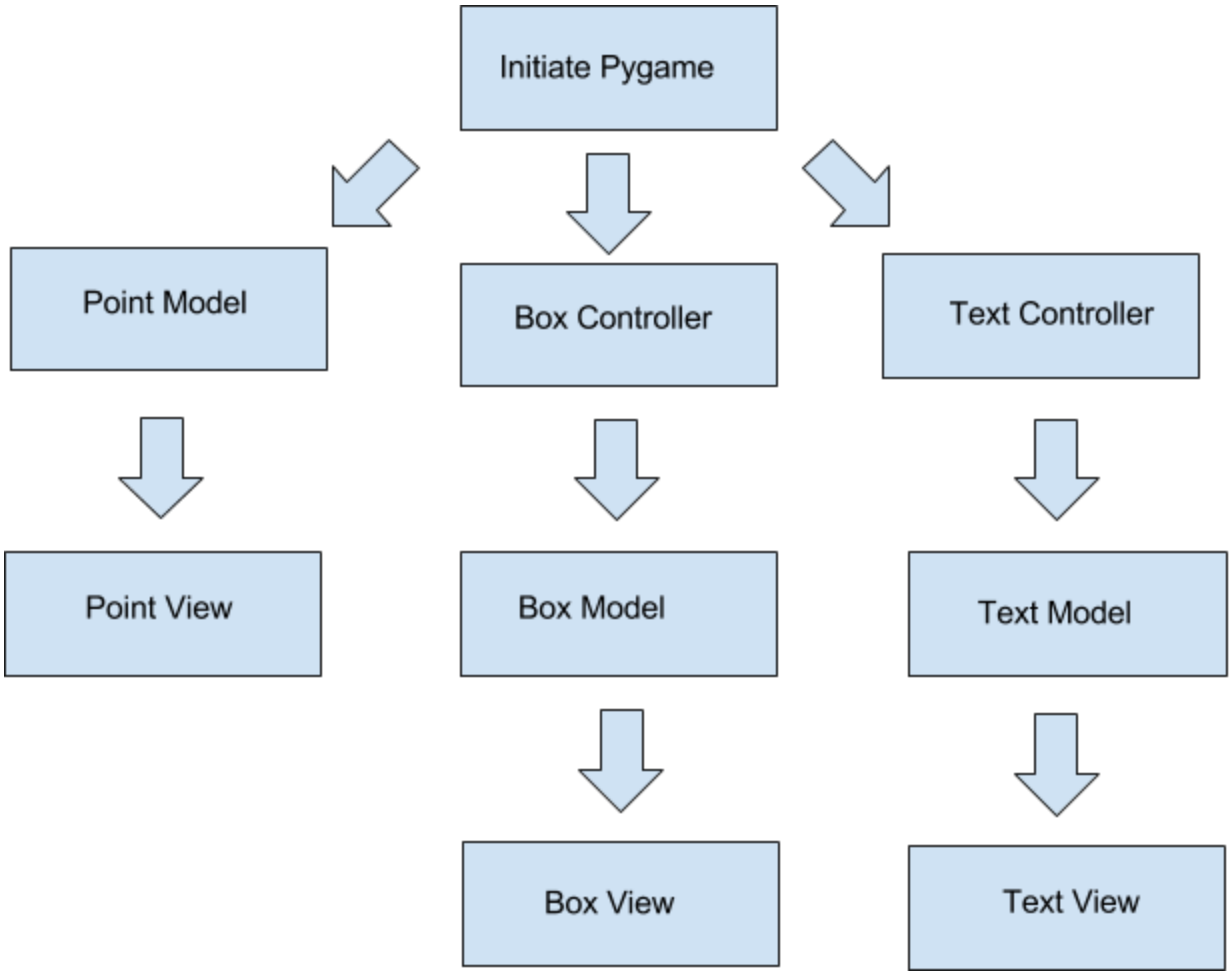
Implementation

In order to use data for our project we had to use the pandas library that allowed us to organize our data into a dataframe. However, there was some difficulty in doing so, because we were pulling our data from two different spreadsheets. After cleaning the data, we created dictionaries of the data that we ultimately put into the data frame.

When creating the interactive part of our code, we decided to split our code into a series of classes that would allow us to keep our data organized. Using pygame, we plotted our points on a screen using two classes. We had the points class, which stored information about each point, then we would pass each point through the points view class, which drew each point on the screen. We had a similar structure for multiple parts of our code, including the text box. For the text box, we had three classes. The InfoButton class stored information about the rectangle box, the InfoButtonView class drew the button, and the ButtonController class controlled when the button would appear.

Towards the beginning of the project when we were going through our logic, there were a lot of design decisions that we had to make. First we decided to use a data frame as opposed to having everything in a dictionaries, because a data frame made the data more manageable to handle. Additionally, when beginning our interactive piece of our project, we had to determine how to structure the code. Originally we had one class for

each object. For example, we had one class to store the point's information and draw the points. However, once realizing how complicated this made our code, we restructured it so that there was a class that created the object, a class that drew the object, and a class that controlled it. By making this change, we were able to reduce the amount of redundant code and make the objects compatible with other objects.



Reflection

From a process point, both partners were on the same page and dedicated to the project which allowed for the an enjoyable project experience. Some problems lied in our approach to the process. Of the 2 week project, we dedicated a great deal of time into learning external packages like pandas and geoplotlib. We poured about a week into understanding the external mapping visualization package called geoplotlib. We ultimately learned the package was not as flexible as we thought it was and abandoned the package for pygame. This was a large waste of time for this project, however now we are better at reading and comprehending other people's code. We were very

particular about version control and testing. Visualization methods and classes were initially tested on code that did not include the mapping data in order to isolate problems. Having understood classes and inheritance better prior to starting the project would have dramatically changed the structure of our code and would have allowed our code to be cleaner.

We planned the division of work to be Katya did the data cleaning and Samantha doing the visualization. We found that after the data was complete Katya also began working with visualization. While tackling big problems, like figuring out how to link the languages to the a moving textbox that follows the mouse click. We both separately worked on the problem experimenting with different approaches and then followed through with whoever's approach was better/ more efficient/ worked. We had a some difficulty with scheduling because both of us are very involved on campus in different activities. This caused us to have short meeting when we could and reconvene to update our progress. We also worked around this by working at night after project team meetings and such.