Project 4
Interactive Programming
Stephanie Northway and Caleb Kissel

Overview:

Our project uses the Bokeh library to visualize data gathered by Google regarding copyright takedowns across their services. Specifically, we provide a visualization of Fox's attempts to remove content, and provide a slider that can show both successful and unsuccessful attempts

Results:

Our project performs as expected, modifying circle size to indicate amount of takedown requests filed in a given month. It responds promptly to the slider and is effective at displaying this particular dataset.

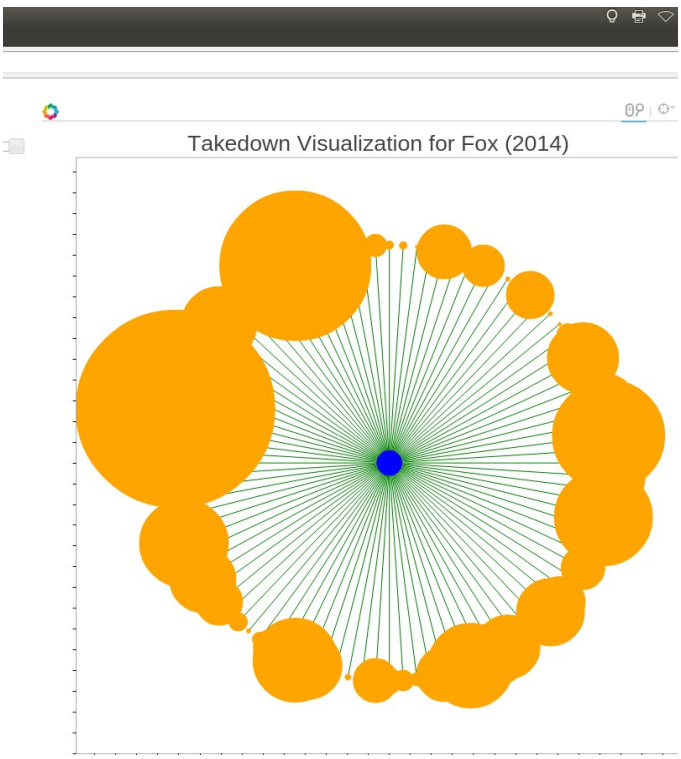
The code itself is relatively general; color is selected by the type of circle, and size is given on a month-by-month basis pulled directly from a .csv. Though tuning may be required to present data well, in general this code would work on any involved party, requester or target. However, the data set is too large to handle with the pandas library; the .csvs are over 2 gb in size. Further complicating matters, they are unordered and multiple entries reference each other, meaning that they cannot be safely truncated. This limits us to examining just one party at a time.

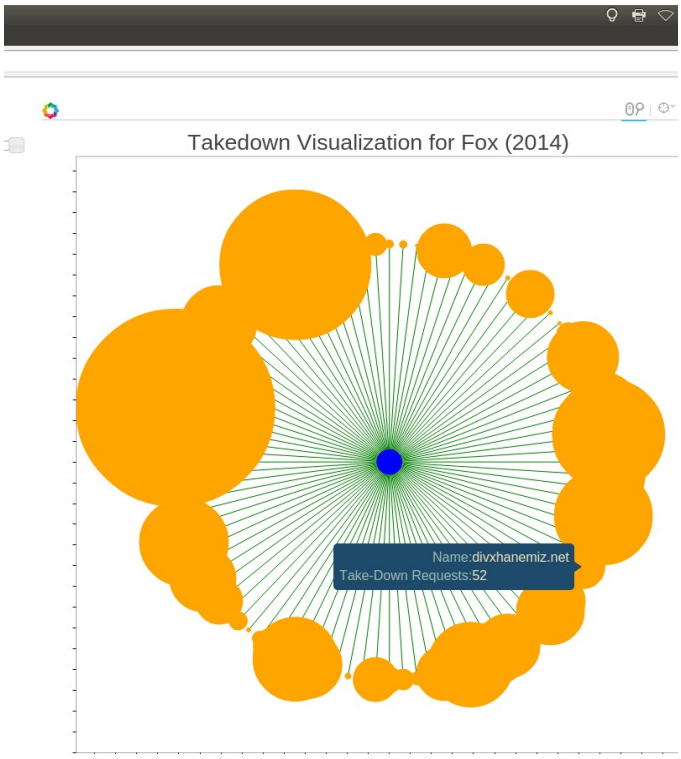
Implementation:

Our visualizer takes the form of a javascript applet and companion data parser. First, the parser takes in a name and searches out relevant information among the .csvs. Then, it creates objects of the class 'entity'. Each one has a name, a category (requester or target), and a list of how many requests were exchanged between that entity and the main central entity. These entity lists are themselves passed into an object called Display_Entity, which contains two items; one, the main entity, and two, a list containing each entity that filed against them (or was filed against).

This object is sent over to the applet creator, which calls another function to create a ColumnDataSource, a type of dictionary used by bokeh. This is the file that will be used to draw, so the actual dataset used for ColumnDataSource is given by the slider position. The applet then draws circles and lines appropriately to represent the data.

Originally, we wrote the plotting function without any objects to represent entities. This function just took lists of circle sizes and mapped them to positions. It looked much the same as our current implementation, but was inflexible. For example, we decided we wanted tooltip information of various kinds to appear over the circles when hovered upon. This required that we tie our information together under the umbrella of a class, which resulted in our current implementation. It took a lot of time, but is much more robust in the end, and makes extension to add more effects (changeable lines, etc) possible.





Reflection:

In retrospect, our dataset was both very large and very interconnected. Either on their own would have been a non-issue, but both complications together meant that we were unable to work with the whole dataset or truncate it effectively. We ended up having to work with hand-truncated data, which is not the optimal situation.

Nonetheless, our code itself can handle nearly any dataset; it just so happens that they only way we can pass it complete data is to select chunks of the .csv manually. This robustness is due to a strong object-oriented approach on our part, the result of significant re-factoring. As a team, we were able to move quickly to address problems, and communicated well. We worked well overall, although we divided the project very neatly in half, and have learned less than we each hoped about the half that belonged to the other. Before beginning, it would have been useful to be more familiar with the interactivity options of bokeh, as we only discovered how challenging this requirement was as we reached the end of the project.