Homework #3: Global Developments

In this homework, you'll apply knowledge about D3 animations and joins to create a multi-line chart (i.e., a line chart with multiple lines) that interactively updates as countries and attributes for the dataset are changed. This assignment i[illegible] will include the following aspects:

- Download the Global D[illegible] (`global_development.csv`) from the CORGIS website for use in your visualizatio[illegible]es of interest that you want to visualize.
- Display a line chart on [illegible]hows a user-selected set of countries (one country per line) over the dataset's time[illegible].
- Using HTML controls, t[illegible] (1) which countries are shown in the line chart, and (2) which global development attribute is being visualized. When countries are added/removed, or when the attribute is changed, you will use D3 joins and transitions to animate the line chart from the "previous state" to the updated one.
- Additionally, the user should be able to invoke a "playback" option, which will re-draw the line chart, drawing the lines from left-to-right using animations.

We don't give you any starter code for this assignment: you'll have to create everything from scratch, including downloading the the dataset and making the `index.html` page. You have the freedom to stylize your chart (and webpage) as desired, though points will be deducted for sloppy design.

> ❗ This homework asks you to perform very specific animations and interactions. Be sure to carefully read the steps below!

As a reference, the GapMinder site shows a multi-line chart with some of the functionality that you'll be implementing: https://www.gapminder.org/tools/#$chart-type=linechart. Specifically:

- The line chart shows several countries across a time range (one country per line). You can select a development indicator you want to visualize on the y-axis. When you do this, the chart will update to plot the new attribute's data. (However, the GapMinder site simply re-draws the chart with the updated data. You will animate the lines from the old-to-new positions.)
- You can also add and remove countries from the chart. (In the Gapminder site, you can only add/remove one country at a time. You will add/remove a group of countries based on their geographic region, such as all of the North American countries. The Gapminder page also requires you to click an `Apply` button to update the line chart; you will update the page as soon as the user selects/deselects the countries.)
- When you hover over a country's line, all of the other lines on the chart are de-emphasized by making them semi-transparent. You'll implement a similar highlighting on your page.
- Clicking the play button animates the lines across the x-axis's time range. You'll implement a similar "drawing animation" on your page.

> ❗ As always, **the sharing and copying code with other students is considering cheating, and passing off (part of) a codebase from someone else as your own is plagiarism.**

Step 1: Create your dataset and initial webpage

Download the Global Development dataset from the CORGIS website and pick ten attributes that you would like to visualize.

There's a `data` folder in this repository where you should place the downloaded CSV file. The data folder contains a `country_regions.csv` file which you will use to map the countries to their geographic regions (Europe & Central Asia, Latin America & Caribbean, etc.).

> ❗ If a country is in the CORGIS dataset, but NOT in the country regions file, you can either ignore it (i.e., not visualize it in your webpage) or you can manually add it to the `country_regions.csv`.

Create an `index.html` pag[illegible]n the root directory of this assignment repository. Link to the D3 library (remember, y[illegible] and if desired, create external CSS or JS files to put your code. You may also use Boo[illegible]ugh this is not required.

The exact design of your we[illegible]out it should include the following.

- Simlar to previous homeworks, at the top of the page, title your page and add your name and email.
- You'll need an SVG to hold your chart. The size should be between 900 x 600 and 2048 x 1080.
- You'll also need a control panel to hold the HTML elements for manipulating your visualization. The control panel should have the following controls (with text labels):
    - **Global Indicator:**: A `select` dropdown listing your ten selected attributes. The currently selected attribute will be what shown on the line chart.
    - **Regions**: You should have a way to select one or more regions (the `World bank region` column in the `country_regions.csv` file). There are seven regions here. The countries for the currently selected region(s) wil be what is shown in the chart. You should also add controls or widgets that let you do "Select All" and "Deselect All" actions which will select (or deselect) all seven regions. It is up to you exactly how you want to implement this Regions functionality, though my suggestion would be something like organizing the set of checkboxes inside a `div` or a `select` dropdown (here's an example of the latter).
    - **Opacity**: When showing a lot of countries in your chart, overlapping lines might cause clutter. One way to get around this is by making the lines semi-transparent. You can use an `<input type='range'>` slider for this. It's not necessary to make the opacity go all the way down to 0% (i.e., fully invisible); I'd recommend a value between 20-40%.
    - **Play**: A control that lets the user animate the lines across the chart (like as shown on the Gapminder site). One way to do this is with an `<input type='button'>` element. When pressed, the chart will begin animating through the years. Alternatively, instead of a "Play" button, you could add an icon or image looks like a play button.

My suggestion is to place these elements in a control panel area, either to the side of the chart or above or below it, so they can be organized in a way that makes sense. Controls should use consistent styling/theme. Feel free to use Bootstrap or custom CSS to do this.

Step 2: Import (and wrangle?) your dataset

Import the two CSV files into your page: the CORGIS `global_development.csv` file and the `countries_regions.csv` file. You may do any data wrangling you feel is necessary, either before or after importing your csv files. The point of this step is to get the dataset to a point where, based on the user interacting with the HTML controls, the line chart can update appropriately.

> 🔍 The dataset goes from 1980-2013, though some countries may be missing data for certain years. If a country attribute starts after 1980, or stops prior to 2013, you can stop drawing the line at that point. If a country has a "gap" year (e.g., say there's data for 1990 and 1992, but not 1991), you can either (1) directly connect the line between the existing years (i.e., connect the line from 1990 to 1992), (2) make the line go to a value of 0 for the gap year, or (3) you can stop drawing the line for the missing year, and pick it [illegible] year (i.e., the line would stop drawing at 1990, and re-start at 1992).

Step 3: Line ch[illegible]ngs

The default encodings for y[illegible] be the following:

- The x-axis will be bottom aligned, and the y-axis will be left-aligned. The x-axis should go from 1990-2013. The y-axis will go from 0 to the max value of the currently selected global development attribute, based on the current set of countries from the selected regions. Use linear scales scale for these axes, and be sure to label each axis (the y-axis label should update as you change the attribute!).
- For each selected region, draw the countries from that region in the line chart. Each country should be drawn using a line. The thickness of the lines is up to you, but pick a reasonable size (not too skinny, not too thick). The lines for each region should have their own distinct color hue (e.g., all of the Latin America & Caribbean countries are color A, while Europe & Central Asia countries are color B, etc.).
- At the end of each country's line, draw a circle (similar to what is shown in the GapMinder page). The size should be a little bigger than the thickness of the country line, and you may optionally give this circle a border. To the right of the circle, add a text label that is the name of the country, colored using that region's color hue.
- The opacity of the lines will be based on the range slider in the control panel. The "max" value should be "no transparency;" it's up to you how transparent you want to make the minimum value.

When your page initially loads, it's up to you whether you want the chart to be blank (i.e., no regions selected), or pre-populated with data.

Step 4: Line chart interactions

Based on the user interacting with the control panel, your chart should perform the following actions:

- **Adding a new region**: If the user adds a new region by clicking/checking it on the control panel, you will add that region's countries as new lines on the chart. In doing this, you might also need to adjust your y-axis (in the case that one of the new countries has the new max value for the y-axis). Follow this sequence of actions when adding a region:
  - First, if the y-axis needs to change, do so using an axis transition. At the same time your axis is animating, you should also call an animate/transition on the existing lines on the line chart, so that they animate alongside the axis to their new positions (based on the update y-axis scale). To do this, you can invoke transitions both on the axis and on the existing lines, and make the transitions have the same duration. The default transition duration is 250 ms, but this is very fast, so set it a little slower so it's easier to follow (I recommend between 500-1000 ms).
  - Next, once the y-axis and existing lines have completed their animations to their new positions, add the lines (with circles and labels) for the new countries to the chart. When adding these, make them "fade in" using an animated transition that begins at 0% opacity (i.e., fully transparent lines/circles/labels) and goes to the current opacity setting based on the control panel's range

slider. Keep in mind, you'll want to delay this animation until after the axis transition completes. Similar to the above step, you can also pick the duration that you want this fade in to take (between 500-1000 ms).

- **Removing a region:** This is the opposite of the above interaction. Here, the user can deselect a region from the control panel, and you will remove its countries from the line chart (and, if necessary, re-scale the y-axis). The steps f[illegible] the above, only in reverse:
    - First, make the co[illegible]g removed "fade out," by transitioning their opacity to 0% (use the same du[illegible]n removing the lines/circles/labels from the chart.
    - Once the lines ha[illegible] out, if necessary, transition the y-axis to its new scale and transition the rem[illegible]e the animating y-axis. Use the same animation duration for this as you di[illegible]
- **Changing the y-attribute:** If the y-attribute is changed, you'll need to transition the y-scale's domain based on the new attribue. Compute the new max for the current set of countries in the chart, and transition both the y-axis and the lines based on this scale. (This transition should look like the axis transitions you might do in the "add a new region" and the "remove a region" steps.)

- **Change the opacity of all countries:** If the user modifies the opacity range slider in the control panel, adjust the opacity of all country lines (and circles and labels) that are shown in the line chart accordingly. New countries that are added should also be shown at this opacity.

- **Hover on a country line:** When a user hovers on a country line, circle, or text label, you should highlight that country and de-emphasize the other countries in the chart. (The Gapminder site above does a similar action: try hovering on a line and see how the other countries are made more transparent.) Make the hovered country's line/circle/text fully solid, and make the other countries a bit more transparent than they currently are. One consideration here is the opacity range slider: you'll want to make the countries more transparent than what the slider is set to, while still being at least somewhat visible (i.e., not all the way to 0%).

- **Playback button pressed:** The final interaction is when the user presses the Play button/image in the control panel. When this happens, you should clear and re-draw the line chart in an animated manner, like as shown in the GapMinder example above. (Here is a D3 example you might use as a reference, though this only animates the lines; you'll also need to animated the cirlces and labels: Animated path multi-series line chart .) Set the transition duration to be a little bit slower than D3's default of 250 ms (I recommend between 500-1500 ms, but this is up to you).

Step 5: Extra credit opportunities

There are three ways you can potentially receive extra credit for this assignment:

- Your webpage and visualization is designed/styled in an especially attractive way that the grader likes. Note that this E.C. opportunity is not applicable for a grade appeal. (+1)
- Instead of only showing the country text labels to the right of the line, the user can instead toggle to show flag icons instead. To do this, I'd suggest adding an HTML control in your control panel that lets you toggle between showing text labels and showing flag icons. You might consider using the Free Country Flags in SVG dataset for this, or you can find your own set of flag images. (+1)
- Let the user change the beginning and ending timesteps for the line chart (i.e., instead of your chart always showing the x-range from 1980 to 2013, let the user pick the start and end points). To do this, you'll probably want to add HTML controls in your control panel that let you specify the beginning and ending timesteps. If the user updates one of these, you should use animated transitions as appropriate

to modify the x-axis, the y-axis, the country lines, and the country text labels to fit the new beginning and end years. How you schedule and invoke these transitions is up to you, but they should look nice. (+2)

Grading

This assignment is worth te

- Step 1 is worth 1 point
- Step 2 is worth 1 point
- Step 3 is worth 3 point
- Step 4 is worth 5 point
- Step 5 is worth up to 4 extra credit points