CS 171

Homework 5

This homework is due on *Monday, February 29th 2016 at 11:59 pm EST*.

This homework requires that you have read the article “Enter, Update, Exit” by Christian Behrens: https://medium.com/@c_behrens/enter-update-exit-6cafc6014c36#.3mzg93bya

Optional reading: Mike Bostock’s ‘Thinking with Joins’, and chapters 9-10 in *D3 - Interactive Data Visualization for the Web*.

1) FIFA World Cup™ - All Time Statistics

> The FIFA World Cup™ is the biggest single-event sporting competition in the world and is played by the senior men’s national teams from the 209 Member Associations of FIFA. The competition has been played every four years since the inaugural tournament in 1930, except in 1942 and 1946 when it was not held because of the Second World War.
>
> The 20 FIFA World Cup tournaments have been won by eight different national teams.
>
> *FIFA (http://www.fifa.com/fifa-tournaments/statistics-and-records/worldcup)*

Template

To help you get started with this homework assignment we have prepared a template that you can use. It is based on the front-end framework *Bootstrap* and it includes the JS libraries: *D3*, *D3-tip* (tooltips) and *jQuery*. Furthermore, a CSV file (`fifa-world-cup.csv`) is stored in the folder *“data”*. Of course, you can also start with an empty project and just copy the dataset.

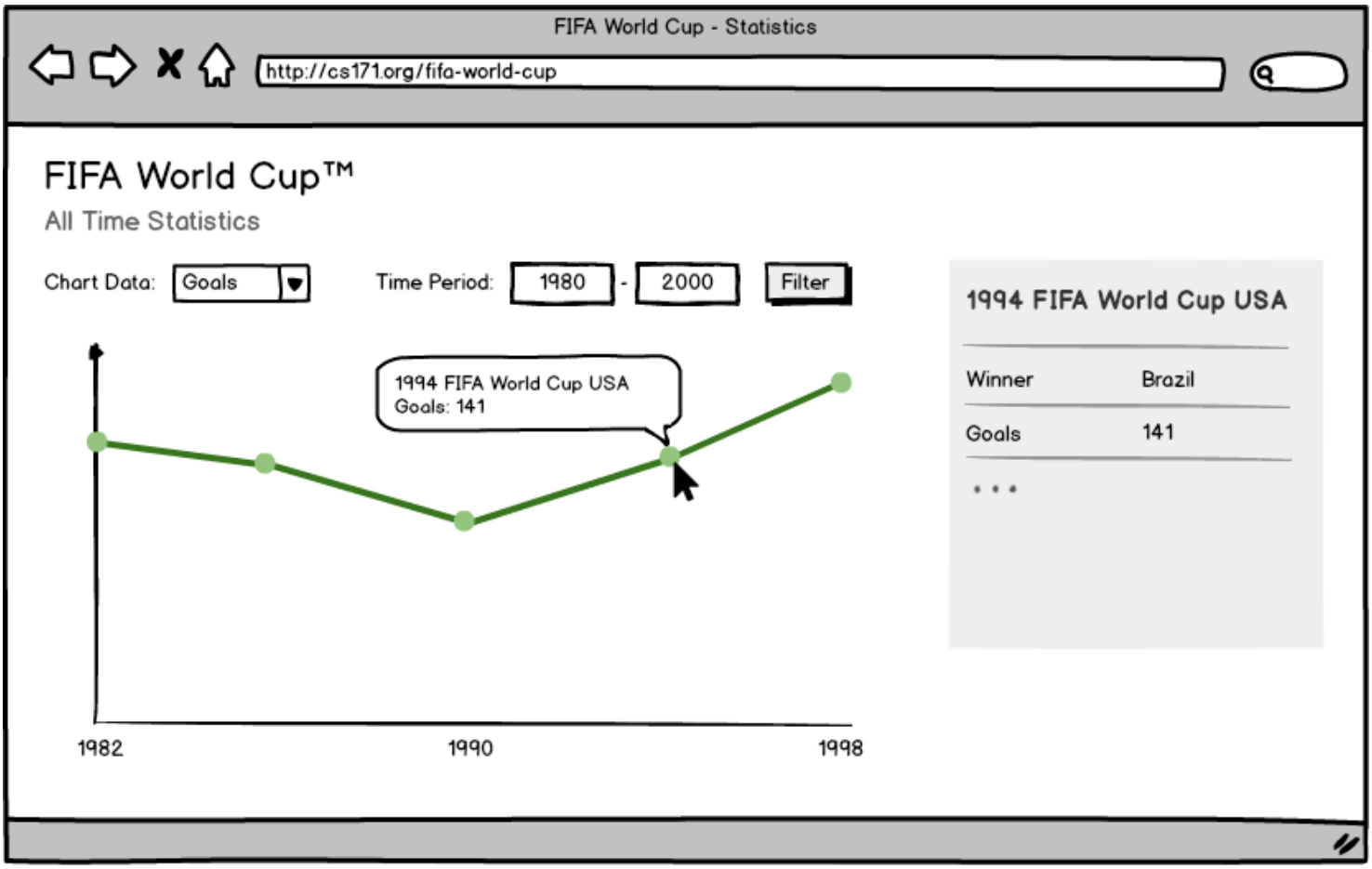
http://www.cs171.org/2016/assets/scripts/hw5/template_data.zip

Overview

A D3 line chart will be the core component of your visualization. The x-axis represents the time period and the

y-axis will show the attribute of the current user selection: **goals**, **average goals**, **matches**, **teams** or **average attendance.** Additional input fields allow the user to limit the time frame that is shown. If you hover over an event in the line chart, the most important attributes are displayed in a tooltip. If you click on an event, detailed information is visible in a separate container to the right of the chart.

You can use the following mockup as a guide:



Implementation

1. **Download the resources**

   Please download the template and the CSV file as a ZIP file:
   http://www.cs171.org/2016/assets/scripts/hw5/template_data.zip

2. **Familiarize yourself with the provided framework and the FIFA World Cup data**

   main.js

   - The script creates a new SVG drawing area inside the container `#chart-area`
   - We have separated the loading of the CSV file and the update sequence. The function

`loadData()` reads the data and saves it in the variable `data`. The function `updateVisualization()` should include all the dynamic chart elements and should be called every time something changes. Beyond that, feel free to customize the structure according to your preferences.

- We have used the D3 time formatting function to parse string values (year the world cup was held) to *date objects*. You can read more about D3 time formatting here: https://github.com/mbostock/d3/wiki/Time-Formatting.

  If you want to convert it back - to a year value with four digits - you can use: *formatDate(d.YEAR)*

```
var formatDate = d3.time.format("%Y");

var newDate = formatDate.parse("2020");
newDate  // Returns: Wed Jan 01 2020 00:00:00 GMT-0500 (EST)
var year = formatDate(newDate);
year     // Returns: "2020"
```

3. **Create scales and axis functions**

   - Create the inital scales and axes. Make sure to pick appropriate scales, especially for the x axis!
   - Keep in mind that you should update the input domains each time the data changes.
   - We recommend that you first implement the y-axis for showing the data attribute *"goals"*. After you have implemented the initial line chart, you will then connect your visualization with a select box to create a dynamic y-axis (in Activity II, part 6). This will allow the user to interactively change the attribute that is being visualized on the y axis.

4. **Use the `d3.svg.line` function to map the data to an SVG path and draw the line chart**

   - Read more about D3's path generator: https://github.com/mbostock/d3/wiki/SVG-Shapes#path-data-generators
   - Create a path by using the d3.svg.line function.
   - Try different values for the ***interpolate*** property (*linear*, *monotone*, …) and choose a setting that fits best.
   - Make sure that you can update the path with new data afterwards
   - Debugging hint: Make sure you only draw a single path!

   Important: Usually, in D3 you map your data elements to the same number of visual elements (e.g., circles). Here, you will map all your data elements to a single path. This is reflected in a different way to create and update your path, different to the standard enter/update/exit methods. Also, you might want to look at the difference between data and datum in D3.

5. **Draw the axes**

- Append the axis components to the drawing area only once
- The axis functions must be called every time the data changes

6. **Emphasize the data points on the D3 line graph**

   - Append an SVG circle on each x/y intersection *(20 tournaments = 20 circles)*
   - The points are meant to highlight the individual events and will be used as an anchor point for the tooltips later
   - Use the D3 update pattern: enter, update, exit

7. **Implement a dynamic y-axis**

   The user should be able to switch between different data attributes for the y-axis:

   - Goals
   - Average Goals
   - Matches
   - Teams
   - Average Attendance

   Create an HTML select box and react to changes in your JS code (update scale → redraw visual elements). The default value should be the *number of goals*. The x-axis shows the time frame and always stays the same.

   Make sure that your code stays well-structured and understandable, especially after these types of modifications.

8. **Include smooth, animated transitions whenever something in the visualization changes**

   Instead of jumping from one state to another, perform a smooth transition when the user selects a different mode (y-axis value). The duration of the transition should be *800ms*.

   - Add transitions to the path, the overlayed circles, and the axes

9. **Implement tooltips**

   - The provided template contains the D3 tooltip library ***d3-tip*** (CSS and JS): https://github.com/Caged/d3-tip
   - You can use the libray or create your own tooltips
   - Display the tooltip whenever the user hovers over a circle in the line chart (circle = specific world cup instance)
   - The tooltip should include the ***edition*** (year, country) and the respective ***y-axis value***, which depends on the users selection

10. **Implement a time period filter**

    Enable the user to limit the visible FIFA World Cups to a specific time frame.

    - Create two HTML input fields (lower and upper bound) and a button to update the visualization. Optionally, if you feel confident in JavaScript and were able to finish the last steps quickly, you can also implement a range slider.
    - The default values should cover the whole time period of the data (20 FIFA World Cups).
    - After the user modifies the time frame you have to filter the dataset, update the scales/axes and redraw the SVG elements (`updateVisualization()`).
    - You don't need to worry about error handling here. We expect that the user enters a valid time frame.
    - Adjust the transition of the circles and the path in the line chart: Think about how you can best visually represent the filtering action in the transition. The transition should be intuitive and not break the mental model the user has of the data. Use the animation below as guidance.

11. **Show the details for a single FIFA World Cup edition**

    If the user clicks on an SVG circle on the line graph, the invidual statistics for this specific event should be displayed in a separate container beside the chart.

    - Extend your HTML structure to display the additional information
    - Create a ***click listener*** for the SVG circles. For this you have to make sure that the tooltip you previously created does not overlap with the circle. You might have to add an offset to the tooltip to accomplish that.
    - We have prepared an empty function `showEdition(d)` . You can call it with the data of the selected event and implement the logic there. Re-factoring code into separate functions is a crucial factor for maintaining the readability and extensibility of your programs. Your D3 update sequence remains compact and you can reuse the functions.
    - Use *JS functions*, *D3* or *jQuery* to update the HTML content with all the available information for a

FIFA World Cup edition: *title, winner, goals, average goals, matches, teams, average attendance*.

12. **Create a proper style for your webpage**

    Maintain good spacing between UI components, overall layout, font size, color scheme, etc. Also, make sure that transitions are intuitive and clear.

2) Interaction Design

1. **Chose a visualization**

   Chose one of your designs from last week. If you don't think they are sufficient for this task, please provide a new one. You can also use a different dataset, but only if you can provide a link to the data source (no artificial data).

2. **Create an interaction storyboard**

   Design an interaction storyboard (branching storyboard) that uses at least three interaction strategies for data and view specification or/and view manipulation. Imagine yourself in a situation where you have to present your interaction concepts to a client. Please use pen and paper.

3. **Answer the following questions/tasks for EACH interaction:**

   - Which questions can you answer by using your interactions?
   - Use the *Heer & Shneiderman* taxonomy to classify every interaction.

4. **Create a PDF with ALL your results**

   - Scan your branching interaction storyboards.

3) Submit Homework on Vocareum

Submission instructions:

1. **Upload your D3 homework under 'work/hw/implementation'.** (Depending on your project structure you might have to create new directories in Vocareum. You can upload multiple files at once into the same directory or directly upload your zipped directory tree.)

2. **Upload your design part as a PDF file under 'work/hw/design'.** (Make sure to keep the overall size of your submission under 5MB! Sketches don't have to be in the highest resolution, but should still be readable.)

3. **Upload the completed lab (activity I, II, and III) under ‘work/lab’**

4. **For DCE students, upload your lecture activities under ‘work/lecture’**

For your final submission you will have to:

- Click the ‘submit’ button
- Double-check your ‘latest submission’, check that all the visualizations are working as expected. You can run them directly in Vocareum and look at them in a new window
- Upload a teaser image of your homework (under ‘Action’/‘upload gallery thumbnail’. This teaser will show up in the classes gallery and should motivate other students to look at your solution in more detail.