LAB INTRODUCTION

In this Lab you will apply all the techniques you have learned in class so far and build your first "Event Driven" website using only HTML, CSS and JavaScript. This will be a single web page that can be rendered in the browser directly (by just simply opening the file in a browser) or by placing the contents of the file in your apache2 web servers documentroot folder and loading the page on localhost.

Submission Process

To complete this project you will write and **submit 1 file**: one HTML document with CSS and JavaScript. Submissions will be on canvas under Lab 3.

Late submissions will be penalized by 5% per day (or part thereof), up to 20 days. After 20 days, the assignment will be given a 0%.

Background:

As you have seen in class so far there are many interesting things you can do with JavaScript and event driven programming. In this lab we will focus on a library that helps you achieve this goal a little more efficiently. The name of this library is jQuery and it facilitates AJAX/AJAJ (AsynchronousJavaScript And XML/JSON).

Requirements

To complete this project you will write and submit **one HTML file** named lab3.html. This file may contain embedded CSS code, and will certainly contain embedded JavaScript and/or jQuery code. This file must be an ASCII file (i.e. a plain text document with a .html extension). You may create it with any editor but you must ensure that it is a text file. In other words, it should not contain anything except ASCII characters (including HTML tags/CSS rules/JavaScript and content).

YOU MAY USE A CODE EDITOR! This editor can be used to generate the HTML markup of your document. Any CSS or JavaScript must be subsequently added to this file by hand (i.e. manually editing the HTML file).

This lab will be a web page containing a quiz implemented upon an interactive Google Map. The behavior of this program must match the specification below. ***Where unspecified, all behavior and layout choices are those of the programmer.*** If clarifications are needed, please post questions on **Slack** in general.

(To assist you, an example HTML file that loads a Google Map has been uploaded to the class google drive in labs folder as mapStuff.html. Note the use of the dblclick event handler in this code, namely how the event's default behavior [zoom] is turned off by preventDefault. *You may wish to use a similar handler for other map controls in your own lab.*)

To help you generate google maps api keys please see the following link:

DO NOT USE YOUR CSUN EMAIL TO MAKE AN ACCOUNT

https://developers.google.com/maps/get-started

Please note that you can still use the map api without a token but it will work a little funky, this is ok so long as all the code is present.

Required behavior:

- When the page first loads, a Google Map is loaded showing the CSUN campus
    - This map can not be scrolled, zoomed, nor may pins be dropped
    - The only acceptable user input is a double-click event **(which does NOT zoom!)**

    - **You need not show the entire campus,** but the map should show most of the campus (or at least the sites of interest)...
- A quiz should then begin immediately after page load
    - Some HTML outside of the map should appear asking the user, for instance, which of the buildings is Jacaranda
    - If the user double-clicks inside the bounds of Jacaranda hall, in this example, the user gets this question right, and a **green** rectangular outline should be drawn around Jacaranda on the map.
    - If the user double-clicks outside the bounds of Jacaranda hall, in this example, the user gets the question wrong, and a **red** rectangular outline should be drawn around Jacaranda on the map
    - Concoct a quiz with 5 such questions, asking the user to identify parking lots or buildings on the CSUN campus
        - The questions of the quiz should appear dynamically (i.e. in response to user click events)
            - The questions need not be dynamic themselves. In other words, you may ask the same quiz questions every time, and even have them "hard-coded"
        - If a user double-clicks in the correct region, it is outlined in green
        - If a user double-clicks incorrectly, the correct answer region is outlined in red
        - Some text should appear beneath the current question to indicate whether the user got the question right or wrong
        - Every new question should appear in place of the prior one. The right/wrong-answer text should appear only after the user has selected an answer with a double-click (and should otherwise not be visible).
        - If the building or lot in question does not have a perfect rectangle shape, the corresponding red/green outline should **approximately** surround the structure. Use your discretion.
    - After the user has answered all 5 questions, an animated div containing text should fly onto the screen (using any type of animation you enjoy). This div presents the results (example: "You scored 4 out of a possible 5 correct!")
    - You may program the animation using any JavaScript/jQuery method you prefer
    - **Avoid CSS animations or other non-JavaScript solutions** for this part

The stylesheet and JavaScript/jQuery should be embedded in the <head> section of the HTML document.

The JavaScript code must use functions where appropriate. You may use anonymous functions for event handling where convenient.

The JavaScript code should have occasional explanatory comments where appropriate (on the order of once or twice per function).

You may use jQuery* for this project if you desire.** You are responsible for locating and learning the Google Map jQuery API on your own if you wish to do this. ***Use your discretion.

Note: you may also mix jQuery and JavaScript code! If you choose to do this, it is advisable to use JavaScript code to interface with the Google Map API, and jQuery to handle user events and create fun effects.

Grade Breakdown:

10%: Submission instructions followed to the letter (1 html and 1 css file submitted, named as stated above, with no contents except plain-text HTML and CSS respectively)

10%: JavaScript has proper formatting, proper use of functions, and comments as appropriate

20%: Web page renders properly in Mozilla Firefox as well as Google Chrome

30%: Web page contains all required content, visually organized and styled according to spec

30%: JavaScript code abides by spec

Cheating: This project (like all projects in this class) is an individual project. You can discuss this project with other students. You can explain what needs to be done and give suggestions on how to do it. You cannot share source code. If two projects are submitted which show significant similarity in source code then both students will receive an F in the course. Note a person who gives his code to another student also fails the class (you are facilitating the dishonest actions of another).

Source code that is copied from websites without citation will also count as cheating, and the same consequences apply.