Homework 1: Calculator Front End

Due date: January 19, 2015 at 11:59pm

This homework is the first in a series of three homeworks in which you will build calculator applications. For this first assignment, you will only build the front end user interface for the calculator. In the second and third assignments you will make the calculator work. For the second homework, you will implement the calculator functions using JavaScript running client-side in the web browser. For the third homework, you will implement the calculator functions by sending web request to a Django application running server-side.

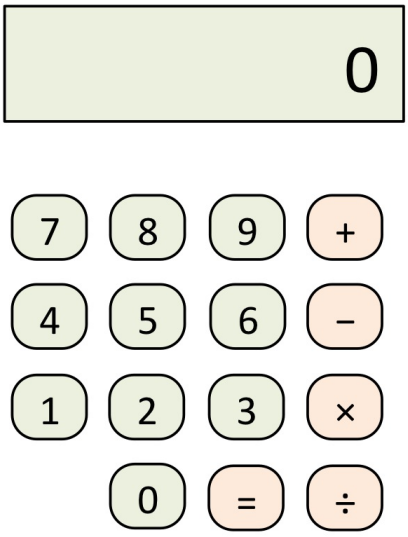
For this assignment, you will use HTML and CSS to create the front end to be used in both the client-side and server-side calculators.

The learning goals for this assignment are to:

- Use Git and GitHub to correctly submit the homework.
- Gain experience with a modern version control system (Git) and practice documented incremental development with Git commit (and commit messages).
- Create a front end for a small-scale application with correct usage of HTML and CSS.

Specification

This section describes the front end of the calculator that you will implement.



- The calculator will contain 15 clickable buttons: a button for each digit 0 to 9, and **+**, **-**, ×, ÷, and **=**.[1,2]
- Buttons must have rounded corners.
- Digit buttons must by a different color than other buttons.
- Your calculator should display a single integer value that the user may not directly edit. For this homework, the value is zero.
- When the user hovers over a button, its color must change.

Note: The buttons will not do anything (yet) when you click on them.

[1] The HTML character entities for these math operations are `&plus;` `&minus;` `&times;` and `&divide;`.
[2] The HTML character entity for the equals sign is `&equals;`.

Requirements

Your submission must also follow these requirements:

- You must have some design and general theme used throughout your page using CSS and HTML.
- You must submit one HTML file called `calculator.html`.
- The CSS must be separated out into (one or more) static file(s).
- You may include images (separated out into one or more static files).
- You must write the CSS and HTML yourself. You may not include packages from other sources (such as Bootstrap, for example). You may not use JavaScript for this homework assignment.
- Cite all external resources used and any additional notes you would like to convey to your grader in the `README.md` file.

Grading criteria

This section contains some specific criteria we will follow while grading your homework. However, keep in mind that *for substantial credit your solution must clearly demonstrate the learning goals for this assignment, which are described above in the introduction.*

Committing your work [20pt]

Your use of Git and your full commit history is an evaluated aspect of your work in this course. As you work you should demonstrate good use of Git, our version control system. By "good" we mean that you should commit small, incremental changes and use meaningful commit messages.

Incremental changes mean that each commit should focus on a single issue, feature, or addition to your solution. For example, if you implement five new features then it is poor to commit all your changes in a single large commit. Instead you should commit five times, with each each commit containing all the changes for a single feature.

Commit messages should consist of a brief one-line summary and an optional, more detailed explanation separated by an empty line. For example, one commit message might be:

```
Add persistent navigation bar to mockup

As per Marvin's paranoid feedback regarding easy navigation, the
sidebar links have been moved to a persistent top bar.
```

The first line of the commit message is similar to an email subject, and the optional explanation is like the body of an email message. For more guidance on how to write good commit messages, see http://tbaggery.com/2008/04/19/a-note-about-git-commit-messages.html).

Specification fulfillment [40pt]

Your submission must follow all specifications and requirements introduced in the previous sections.

Coverage of technologies [40pt]

While we are not strictly grading for style, we are grading for correct usage of the introduced technologies.

- You must run your solution through the W3 HTML validator (http://validator.w3.org/) and CSS validator (http://jigsaw.w3.org/css-validator/) and have no errors.
- It is extremely bad practice to use inline CSS throughout a front end. This is why we require that you put your CSS into a separate file.
- You must use semantically correct tags. For example, use the `<button>` tag for buttons, not the `<div>` or the `<image>` tag.

Additional thoughts

- What HTML elements should you use, knowing that this front end will eventually be used for server-side calculation? Do the types of elements you use change when the front end is used for client-side (JavaScript) calculation?

Turning in your work

Your submission should be turned in via Git and should consist of an HTML file called calculator.html and associated assets (CSS and images). You may want to organize your files logically. Here is an example directory structure (some directories/files omitted) of a submission.

```
[YOUR-ANDREW-ID]/hw1/
  |-- calculator.html
  |-- css/
    |-- calculator.css
  |-- images/
    |-- [images, etc.]
  |-- README.md
```