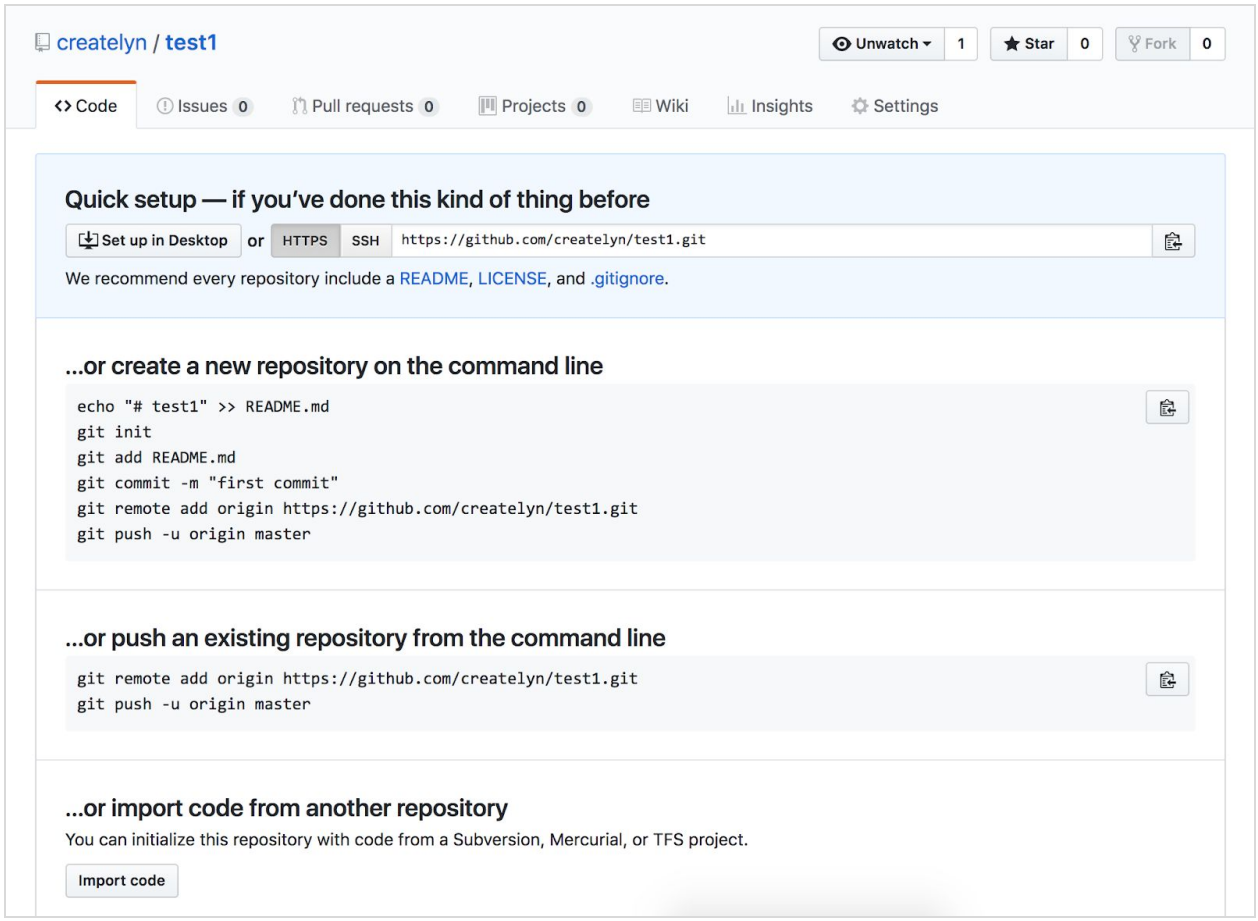
Advanced Web 2
Summer 2018

Assignment 3 Guidelines

For assignment 3, you are going to build off of your assignment 2 to create a marketing website for your favorite vacation destination. You will gain an understanding of adding your project to a Github repo, committing your changes, adding Sass to your project, and using Sass variables. Below is a list of directions for this assignment.

Directions

1. Copy your Assignment 2 from your AdvancedWeb2/Github
2. Call it **`assignment-3_lastname-firstname`**
3. Open your terminal
4. `cd` into your assignment 3 folder
5. Open up your assignment 3 in atom
6. Remove all contents in your `README.md` file
7. Add an `h1` using Markdown with the title “Assignment 3”
8. Using the command `git init` you will create an empty git repository
9. Using the command `git add .` you will add your files to staging to get ready to be committed to your repository
10. Using the command `git commit -m “initial commit”` to commit the files that you have added
11. Open Github
12. Create a new repository
    - Call it **`assignment-3_lastname-firstname`**
13. Don’t initialize this repository with a `README`
14. Don’t add a `.gitignore`
15. Click Create Repository, then you should see the follow screen:



16. You're going to push to an existing repository from the command line using the section that looks like the section below. Use the clipboard icon on the right or copy both lines. You can paste both lines into the terminal at the same time.

...or push an existing repository from the command line

```
git remote add origin https://github.com/createlyn/test1.git
git push -u origin master
```

17. Refresh your page, and you should see your code files.
18. Now you are going to add an .EditorConfig file to your assignment 3
19. Add the contents from the EditorConfig page in your week 3 module in Canvas
20. Add and Commit the file using the message "`added .EditorConfig`"

---------- class ended here ----------

21. Checkout your master branch.
22. **You're going to delete your `.gitignore` file (`git rm .gitignore`) that we originally added in class along with the .pdf (`git rm advancedWeb2_syllabus_summer2018-2.pdf`) I told you to add. We're going to circle back to that during week 4.**

23. Add and commit your changes using the message "`removed .gitignore and .pdf`"
24. Now checkout your `part-1` branch to add your sass file structure.
25. You're going to want to change the name of your current `main.css` to `week3.css` to reference since the new version of your `main.css` will be a compiled version of your `main.scss`.
26. Follow along with the recorded sass video to add sass to your assignment and create the file structure for your stylesheets (the css and sass folders). It should look like this:
    - **css**
        - main.css
    - **sass**
        - main.scss
        - **base**
            - _layout.scss
            - _reset.scss
            - _typography.scss
        - **variables**
            - _colors.scss
            - _typography.scss
27. Once your file structure is completed, add it and commit it with the message "`added sass file structure`"
28. Move the contents in your `normalize.min.css` to the `_reset.scss` file and delete your `normalize.min.css` file and `vendor` folder. Some of you included the reset that you learned from Advanced Web 1, and I'm ok if you use that in your `_reset.scss` file instead of your `normalize.min.css`, which I talked about in the video. Just make sure you delete the `normalize.min.css` and the vendor folder it's contained in.
29. Add and commit with the message "`added reset file`"
30. Import all of your `.scss` files into the `main.scss` with your `reset` at the top, then your variables, then your base, with comments identifying the contents below it. Your main.scss should look like this (include the comments):

```
// Reset
@import "base/reset";

// Variables
@import "variables/colors";
@import "variables/typography";

// Base
@import "base/layout";
@import "base/typography";
```

31. In your terminal use `sass --watch sass:css --style compressed` to watch the entire sass directory for changes, and tell it compile into the css directory, and compress the output
32. Based on the purpose of each file that I broke down in the recorded lecture, separate your `week2.css` styles into the file they belong in. Don't worry about nesting your styles right now.
    - Font styles will go into the typography folder
    - HTML5 structure tags into the `base/_layout.scss` file.
    - Make sure you delete your `week2.css` once you are done moving it's contents.
33. Add and commit your changes with the message "`added original css styles`"
34. Make sure you clean up your code based on what we talked about in class which can be referenced in the **Code Requirements Reference Sheet** at the end of this assignment.
35. Add and commit your changes using the message "`cleaned up code based on code requirements sheet`"
36. Add any colors that you used as variables into the `variables/_colors.scss`
37. Add and commit your changes with the message "`added color variables`"
38. Change all of the color values from their hex value to use their variable name
39. Add and commit your changes with the message "`replaced hex values as variables`"
40. Add the font you used as variable(s) into the `variables/_typography.scss`
    - Use the variable `$font-primary`
    - if you have two make the second `$font-secondary`
41. Add and commit your changes with the message "`added typography variables`"
42. Change the font value to use the appropriate typography variable.

43. Add and commit your changes with the message "`replaces font styles with variables`"
44. Push your changes to your repository (`git push origin part-1`)
45. Checkout your master branch
46. **DO NOT MERGE YOUR PART-1 BRANCH INTO YOUR MASTER BRANCH.** I referenced this in the video, but it will create a merge conflict!! We will go over this in class.
47. Update your `README.md` file to
    - add a brief description of what you did (i.e. added sass to your project)
    - Include a link to your repository using Markdown
    - Include the phrase as a blockquote "**I have read the Code Requirements Reference Sheet. My code follows the requirements**"
    - Include any comments/feedback you want to include on your project
48. Add and commit your changes with the message "`updated README.md for assignment 3`"
49. Push your master branch using `git push origin master`
50. Push your master branch to your repository and submit a link to your repo on Canvas.

Rubric

Criteria	**Points**
Code is in a Github Repository and includes all branches	15
Sass files are setup correctly	15
Styles are in the appropriate files	15
Minimum required commits based on the directions	30
Part-1 branch is merged into Master	5
.EditorConfig file is included	10
README.md file has required contents	10
Total	**100**
Deductions based on Code Requirement Reference Sheet	

Code Requirements Reference Sheet - Assignment 3

Unless otherwise noted, all code must:

- Rely exclusively on external CSS/Scss (no internal CSS, no inline CSS)
- Follow a clear style guide with consistent tabs/spaces, bracket placement, etc. – Use an Editor Config file to maintain consistency
    - Indent child elements (-1 point for each)
    - Use lowercase markup (-1 point for each)
    - Use efficient code. Remember: stay DRY! (Don't Repeat Yourself).
    - Utilize HTML5 tags meaningfully and accurately, i.e. only for content blocks that can stand alone, for any blocks of navigational links, on any and all page titles and subtitles, etc
        - Semantic sizing of h1-h6 - h1 is the largest, h6 is the smallest (-1 point for each)
    - Do not use HTML for solely presentational purposes. Do not use or <h3>, for example, to style content. These tags have their own specific meanings and should be used only when appropriate. (-1 point for each)
    - Empty lines between each `.scss/.css` declaration block (-2 points for each declaration block without an empty line between it)
    - Space between semi-colon of property and value (-1 point for each without a space)
        - `color: $white;` or `color: #FFFFFF;`
- Sass/css follows BEM naming conventions (-3 points for each styled universal selector without a class name)
    - Does not include typography tags
- File structure folders include (-5 points for each):
    - Folder for images called img (if images are included and referenced on .html file)
    - Folder for .css files called css
    - Folder for .scss files called sass

Any and all work you submit in the course of this class is expected to be your original work. The designs must be your own ideas, and the code must be your own handwritten code, **unless** (A) I have explicitly provided code for the given assignment or (B) I have specifically stated otherwise. This stipulation extends but is not all limited to the use of grid systems, templates and boilerplate products. If you have questions about this policy, ask for clarification.

Correct spelling, grammar and punctuation is expected on all work.