Assignment Name: Simple CSS

Due Date: See Canvas

Goals:

You have been given an HTML file which has all of the required content to recreate the provided screenshot using only CSS. **Do not change the body of the html file,** just the head to link the CSS stylesheets and favicons. As always, use best practices, e.g. using an (unmodified) default style sheet, only lowercase letters, syntax and accessibility validation, etc.

Objectives:

1. Demonstrate ability to write simple CSS - particularly the box model.
2. Use hexadecimal values for all colors (please use full 6 digits for autograder purposes)
3. Demonstrate understanding of cascade properties in CSS
4. Demonstrate the ability to use accessibility checkers on your code.
5. Demonstrate ability to host files properly

Resources:

https://github.com/UMSIComplexWebDesign/HW2-2022

You are provided with a repo that contains the necessary html file, the default stylesheet, and screenshots of the final results. You will need to create a css file that will transform the original page to one that resembles our example using the outlined CSS styles below and link it to the html file.

Requirements:

01. All files and folders present, HTML file and CSS stylesheet in proper format, no styles in HTML. **DO NOT MAKE ANY CHANGES TO THE HTML BODY**
02. Always name your stylesheet style.css and keep it in the css folder
03. Always name the default stylesheet html5reset.css and keep it in the css folder (you should use the same file from homework 1 but I provided a copy this time just in case)
04. Validate all of the code - do this throughout the entire process.

---

1. Link the stylesheets correctly.

2. Link the favicon correctly.

3. Style the body with the following:
    a. Default text is **Verdana** font with **Times New Roman** as a fallback option**
    b. The background color is black (use hexadecimal values)
    c. The foreground is white (use hexadecimal values)
    d. There is a 25px margin on the top and sides. None on the bottom.

4. The level 1 heading is 48px, centered, and has padding of 15px

5. The level 2 heading is 32px, with grey (#808080) text-shadow property that is offset 2px horizontally and 4px vertically

6. Paragraphs use 24px font and have 15px padding on the bottom and left only

7. Images match the following styles:
    a. All images are 350px wide (do not set a height)
    b. All images are centered
    c. All images should have a white 10px dashed border

8. All divs must:
    a. be at least 100px tall (but will automatically be taller if needed)
    b. have 20px padding on all sides
    c. have a 3px solid white border on all sides.

9. The intro div should have a 15px border-radius (to curve the edges) and a 15px margin on the top.

10. The red div has the following styles:
    a. Background color is hexadecimal value for red
    b. Text is aligned on the right side
    c. A left margin of 50px;
    d. A border of 3px on the bottom and 10px on the left;

11. The blue div has the following styles
    a. Background color is hex value for blue
    b. Text is aligned on the left side
    c. A right margin of 50px;
    d. A border of 10px on the right and 3px on the bottom;

12. The yellow div has the following styles
    a. Background color is hexadecimal value for yellow
    b. Text is aligned on the right side
    c. A left margin of 50px;

    d. A border of 3px on the bottom and 10px on the left;

    (Let's fix the color contrast here – White on yellow is hard to read!)
    e. The heading and paragraph text is black (use hexadecimal values)
    f. The heading uses a #c7e2c7 text-shadow that is offset 2px horizontally and 4px vertically

13. The green div has the following styles
    a. The background color is hexadecimal value for green (Use #00EB00, #008500, or #00D800) **updated
    b. Text is aligned on the left side
    c. A right margin of 50px;
    d. A border of 10px on the right and 3px on the bottom;

14. Any span elements should be underlined, **have all uppercase**, and not have any text-shadow.

15. The footer should use a gradient background that starts at #000000, goes to #0000ff at 25, #0066ff at 70, and ends back at #000000.

16. Paragraphs within the footer have centered 20px text and 10px of padding on the top, right, and bottom only. (No padding on the left.)

Example

See the Github repo with three sample screenshots. One is from a screen size of 1200px, another is from 650px wide screen, and the other is an example where I have changed the html file that replaced the blue class with an red class and the yellow class with the intro class.

Tips!

1. Write your CSS from broadest to narrowest. Look at the requirements, which ones apply to the whole page? Which ones only apply to certain sections? Use this to help define your CSS selectors.
2. If you get an error, use Inspect Element to check the current value. Make sure that a different rule isn't overriding a rule you wrote.
3. Don't forget that you can validate CSS code: https://jigsaw.w3.org/css-validator/ is one.
4. Order is important for many CSS parameters, even your fonts! Make sure your desired font and the fallback are in the right order.
5. Images can be a little tricky, what CSS parameter can we use to make sure it's always on its own line?
6. Automatically centering elements seems hard but you really only need 1 line of code! Do you remember what it is?

7. CSS means cascading style sheets, use the cascade to your advantage. Sometimes you may only need 1 style for a selector.

Deeper Questions…

1. What error would be generated if you made the paragraph text smaller?
2. What visual differences are there if the html5reset file is not used?
3. Are any of your rules repeated in multiple places in the css? Could you do more to optimize your code?

Submission Requirements:

1. Upload your code to Github in a repo called homework2-2022. Make sure to link it to Github Pages so we can see your deployed code.

2. Submit link to the autograder in Canvas. We want to see the deployed page, not the code so the URL should be **http://*<your-user-name>.github.io/homework2-2022***

Any changes to code after the due date can result in a late penalty at any time during the semester. If you want to modify your code for personal use - make a new copy.

Autograder

1. If you are losing points on the autograder, use Inspect Element to carefully check that the element has the styling that you were expecting.
2. If the autograder is giving you a 0
   a. Check the URL you submitted, does it go to the proper page?
   b. Validate your CSS to make sure you don't have any syntax errors.