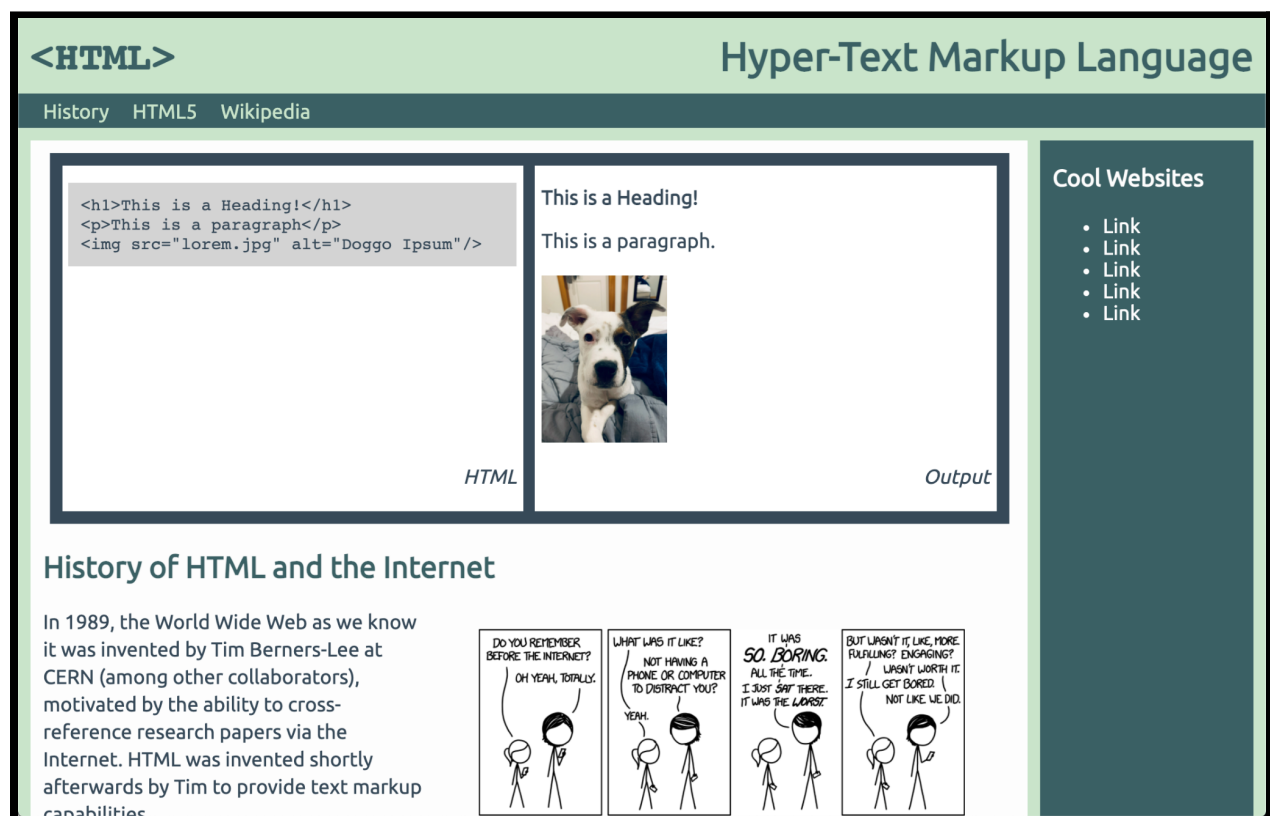


Homework Set 1 - HTML of... HTML!

Due: Sunday, April 18th 2AM PST (morning)

Overview

In this assignment, you will finish implementing a webpage that introduces a brief history of HTML and the internet using various types of HTML tags and responsive layout methods using CSS.



Learning Objectives

- Practice following webpage specifications given a PDF and/or other visual and text-based artifacts as a design basis.
- Write your webpage using HTML and CSS:
 - Choose appropriate HTML tags with an understanding of the semantic meaning of each tag and the context they are used in a webpage.
 - Use CSS properties to style a webpage to match a given specification.
 - Use ids and classes appropriately.

- Use appropriate layout methods to cleanly organize the page in a responsive manner, without over-using HTML nesting of tags or unnecessary flex containers.
- Produce quality readable and maintainable code that conforms to class standards:
 - Separate content and structure (written in HTML) from style (written in CSS).
 - Be able to test that your webpages are valid and conform to HTML and CSS standards.
 - Reduce redundancy in your CSS by using context- and/or pseudo-selectors in your code.
 - Demonstrate understanding of code quality expectations modeled in lectures and in the course's Code Quality Guide.

Starter Files and Final Deliverables

In the hw1-starter.zip file on Canvas you will find the following starter files:

File	Description
<code>index.html</code>	The partially complete HTML file for the “History of HTML” page
<code>styles.css</code>	The starting stylesheet for you to finish.
<code>lorem.jpg</code>	A provided image for the HTML example at the top of the page.

Structure of Starter HTML

The given `index.html` already contains some of the structure for your content, including a header, a main area, and a footer. **Note:** For full credit, you are not to change any of the HTML that is clearly specified as unchangeable in the HTML comments.

Within the `main` area of the page, there is a primary `section` element with three children (some of which may have nested elements):

- `<article id="examples">` - this element contains a side-by-side view of HTML code and corresponding output. **Note:** For full credit, you are not to change any of the HTML within this tag.
- `<section id="history">` - this section provides content with a brief history of HTML and the Internet. You need to finish marking up the content with appropriate tags within, including the two images which should be represented as figures with captions for the source of the image. The provided HTML includes absolute links to the images and the URLs for the anchor tags as shown in the expected output.
- `<section id="html5">` - this section provides a brief summary of HTML5 including a side-by-side comparison of document structure in HTML vs. HTML5. You are not to

change any HTML in the `#html-comparison` section; you will need to use CSS selectors and flex layout styles appropriately to match the expected output.

- There is also an `aside` element within the `main` which represents the right sidebar of the page. You are to finish marking up this element to create a bulleted list of 5 links to interesting websites of your choosing. The link text displayed on the page should not be the full URL, but the link sources should be.

External Requirements

This section will describe the external output of your webpage. For full credit, your solution should adhere to *both* the visual references (screenshots) and text specifications, and you may not add styles that are unspecified (e.g. setting a font size that is never mentioned). If you think the specification is missing a requirement, make sure to review both text-based and visual guides and also confirm that the requirement isn't covered implicitly by another requirement.

Sample Screenshots

Although you do not need to match the output pixel-for-pixel, we find it is helpful for students to know on what system each screenshot came. Example screenshots are included on [Canvas](#).

Provided CSS

We have provided some starting CSS for you to get started with flex layout. Any flex containers should be specified using selectors in the first provided ruleset, and any of these that are columns should be specified in the second. If you would like, you may use a class (e.g. `flex-container` or `column`) for these rulesets and add it to the appropriate HTML tags.

```
/* Provided starter: styles for all flex containers */
<selectors> {
  display: flex;
  justify-content: space-between;
}

/* Provided starter: styles for flex containers that are in column layout */
<selectors> {
  flex-direction: column;
}
```

Text Description of Requirements

For full credit your webpage must have the following:

Page Body

- The page body should have 0 margin on all sides (you will need to override the default margins to achieve this).
- The body should have flex column-oriented layout to display the `top header`, `main`, and `footer` vertically.

Thematic Page Styles (Colors and Fonts)

- The background color of the page should be `#c9e4ca` (the lighter green) and text color should be `#364958` (dark green) unless otherwise specified.
- All paragraphs on the page should have a line height of 1.14 (similar to a text document editor, this refers to the ratio of text and spacing between lines).
- The page should use the Ubuntu Google Font (select **both** Regular 400 and Bold 700 to support bold text on the page), imported with a `link` tag in the HTML, and fall back to Verdana, followed by sans-serif as the system default (in other words, 3 font families should be specified in the correct order in the CSS property).
- The navigation element within the top header, the `#examples` article, and the sidebar should both have a background color of `#3b6064`. This same color should be the text color of the other text in the top header as well as all other headings on the page, links within the main sections (not the sidebar) and the text in the footer.
- All direct children of the main element should have a background color of `#fdfdfd` (there are no elements on the page with any default `ffffff` white color). The text within the side bar should have this same color for the font color.
- Any HTML code, the `&` (ARING_SYMBOL) and the (COPYRIGHT_SYMBOL) referenced on the page should use [HTML entities](#) appropriately to render as shown in the expected output. Such HTML code should also be wrapped in the appropriate *inline* HTML tag and should be bolded.
- There are various links in the provided HTML which you will need to markup appropriately. The full URLs should not be visible in the rendered page, and **no links should be underlined**. Refer to the screenshots for expected appearance. For the image captions, Only the XKCD and w3.org text should be linked, following the unlinked Source: text.
- Visited links in the main section (excluding the navigation bar and sidebar) should use the same color as the background color of the body (`#c9e4ca`) for their font color (hint: use the appropriate pseudo-selector). For the links in the navigation bar and sidebar, their color should be the same whether they are visited or unvisited.
- All headings within the main sections should be second-level headings. The sidebar should have a third-level heading.

- All figure captions should be italicized.

Top header and Bottom footer

- Refer to screenshot at the top of this spec for the header and footer appearance (from Chrome on Mac)
- The navigation element at the top of the page should be marked up with the appropriate HTML elements *within* the top header.
- The list of links in the navigation bar should have 5px of padding on all sides and 0 margin on all sides, as shown on the expected output. Each list item should have 15px of margin on the left and no bullets/other list features should be visible.
- Both the `h1` element in the top header and the `footer` element should have a height of 60px and left/right padding of 10px. The `footer` (only) should have a bottom padding of 10px. Note that the navigation bar should be in the top header, but does not account for the specified 60px height.
- All `h1` elements should have 0 margin on all sides.
- The `<HTML>` at the top header should be positioned in the top left corner and the `</HTML>` in the `footer` should be positioned in the bottom right corner; both should be bold with 24pt font and wrapped in appropriate HTML tags to represent inline code. The “Hyper-Text Markup Language” and the copyright information should be positioned in the opposite corners of their respective elements.
- Text within the top header (not including the navigation bar) and the `footer` should be vertically aligned in the middle as shown in the expected output. Hint: Use flex layout with `justify-content: space-between` and `align-items: center;` on appropriate elements to achieve this layout.
- There should be 10px of padding on the bottom of the `footer`.


Sidebar

- The sidebar on the right should have 10px of margin on the left, 10px of padding on the left/right and a min width of 200px.

main

- The main element should be a flex container with its two direct children laid out with `justify-content: space-between;` in the appropriate cross axis.

#examples

<pre><h1>This is a Heading!</h1> <p>This is a paragraph.</p> <img src="lorem.jpg" alt="Doggo Ipsum" /</pre> <p><i>HTML</i></p>	<p>This is a Heading!</p> <p>This is a paragraph.</p>  <p><i>Output</i></p>
---	---

The HTML for this section is mostly provided for you, but you are to use CSS selectors and layout styles appropriate to meet the expected output (without adding unspecified tags to this section or classes/ids). The only part of this section you are allowed (and required) to change is to modify the contents of the left `div` to meet the expected output:

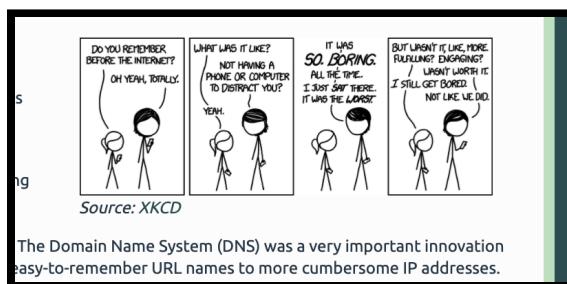
```
<!-- TODO: Fix the HTML below to match the left box of the screenshot -->
<div>
  <h1>This is a Heading!</h1>
  <p>This is a paragraph.</p>
  
</div>
<!-- End TODO -->
```

- The preformatted text on the left should have a background color of `#d3d3d3` and 10px of padding on all sides. You should also add the property `overflow-x: scroll;` to this element so that on smaller page widths, a horizontal scrollbar allows a user to scroll to read the text instead of it overlapping where it shouldn't.
- The `h1` element ("This is a Heading") in the example output should be unchanged in the HTML, but CSS should be used to make it 16pt instead of the browser's default font size for `h1` elements (the other `h1` on the page should still use the default font size). This element should also have 1em of margin on the top.
- The image displayed of Lorem the dog should have a width of 100px.
- The `#examples` article should be a flex container with the appropriate `space-between` property value. It should have 5px of margin on all sides and 10px of padding.
- Both of the `div`s directly inside of the `article` should be flex column containers, with "HTML" and "Output" at the bottom of the respective container. They should be 45% the width of their parent element with 2% padding on each left/right side. They should have the same background color as the rest of the main section (`#fdfdfd`).

Captioned Images

There are two images in the main `section` elements (excluding the one of Lorem the dog in the HTML example). These should be represented as figures with figure captions. As mentioned in lecture, the `float` property is often misused in CSS and we generally prefer flex layout for easier, more responsive layout. However, these are examples where float *is* appropriate in order to float the figures to the side of the text (the figures containing the images and captions should be floated to the right). **These two figures are the only two page elements that should have any `float` layout.**

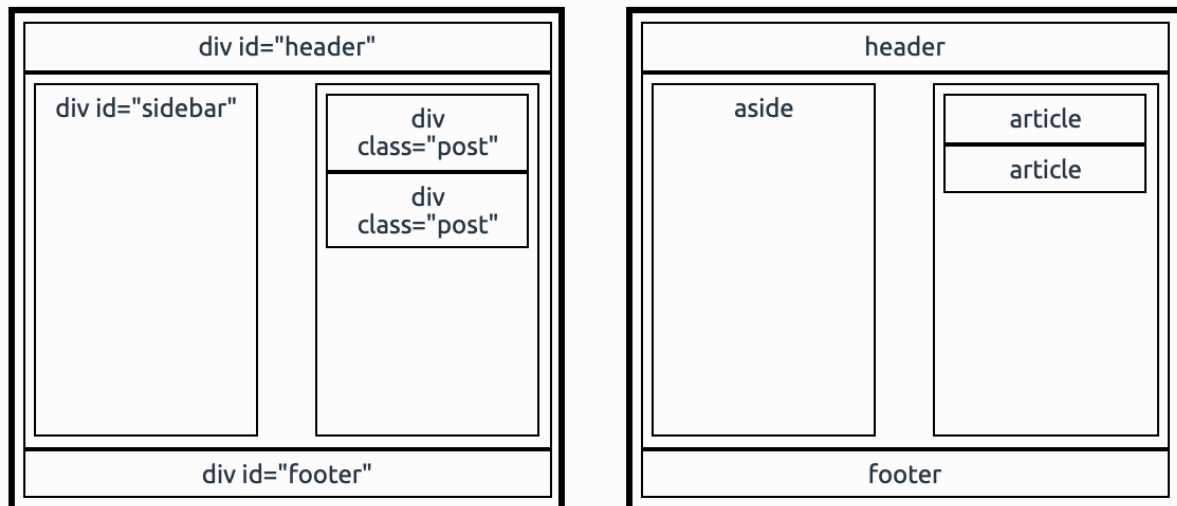
The XKCD figure should have a width of 50% with a max width of 400px. The W3C figure should have a width of 120px with 0 margin on the top and 20px of margin on all other sides. The images within both figures should span 100% of their parent container.



#html-comparison

HTML5

In 2014, HTML5 was released by W3C with various important new features, notably special elements for semantic tags. Semantic tags have made for more standardized content organization of web documents, improved accessibility features, and making it easier and more efficient for search engines to crawl and rank websites.



HTML5 has also added support for more media formats on the web with tags like `<audio>` and `<video>`, and has also added support for various form input tags and attributes for different types of user input and client-side validation.

This part of the assignment will involve the most consideration of flex layout and CSS selectors to achieve the expected output. The HTML within `#html-comparison` is provided for you, including two direct `div` children: `#html-structure` and `#html5-structure`. These correspond to the left and right page layout examples in this section. Both `div`s have nested `div`s within, which you'll need to style and layout appropriately (again, `div`s are appropriate here since they are used for visual layout and not for semantic purposes, though it's a bit meta in this case :)). We have also provided a `"main-structure"` class to help make it easier to layout these elements in CSS.

- All text in the `#html-comparison` element should be centered.
- **All** `div`s within the `#html-comparison` element should have 5px of padding on all sides. All `div`s within the two children of `#html-comparison` should have a 2px solid black border, but these two direct children should have a 4px solid black border.
- Each of the two children of `#html-comparison` should be flex columns, take up 45% width of their parent and have a height of 300px.
- The two `div`s with the `main-structure` class should take up the remaining space of their parent column-oriented containers (hint: use the flex child property `flex-grow` to achieve this). These `div`s should also themselves be flex containers (but row-oriented instead of column-oriented).

Bottom Quote

This HTML is given to you since appropriate ways to structure `blockquote`s with authors can be a bit ambiguous. But you should add CSS to:

- Give a 2px solid black bottom border of the `blockquote` element
- As mentioned above, all figure captions, including this one, should be italicized.
- Align the author text to the right within the `figure`

Internal Correctness Requirements

For full credit, your page must not only match the external output requirements listed above, you must also demonstrate that you understand what it means to write code meeting the web standards expected by this class. This includes the following "Internal Correctness" requirements:

- Your `index.html` file must link to `styles.css` in the head of the page; **no CSS should be written in the HTML.**
- Your HTML and CSS should demonstrate good code quality as demonstrated in class and detailed in the CS 101 Code Quality Guidelines. Remember to review your CP1 feedback for any relevant things to address in HW1. For Module 1, common code quality requirements students miss include:
 - Using consistent indentation, proper naming conventions, curly brace locations, etc. Remember that IDs and classes should be in all-lowercase conventions and multiple words are optionally separated by "-".
 - Keep lines fewer than 100 characters for readability (links are an exception to this rule).
 - Express all stylistic information on the page using CSS defined in `styles.css`, not in HTML. Do not use presentational tags such as `b` or `font`.
 - Do not use any deprecated HTML tags listed on [this page](#).
 - Prefer organizational tags and CSS selectors instead of using [too many classes or IDs](#) in your HTML.
 - Do not include unused, duplicate, or overridden CSS rules or rulesets. Use shared CSS selectors to factor out redundancy (see Code Quality Guide for more examples). For example, the content block shares many (if not all) styles in common, so you should not specify those styles twice in your CSS file. Similarly, use context selectors to avoid needing to apply classes and ids to large numbers of elements. Make sure to double-check that you didn't leave any unused styles in before submitting!
- Use HTML5 organizational tags such as `article`, `section`, and `aside` to divide sections appropriately with semantic meaning. We have provided most of the semantic tags and `divs` for you; in the examples with `divs`, they are used for styling/layout purposes of the

HTML vs. output and HTML vs. HTML5 examples, but are contained with semantically-appropriate tags.

- Use `h1–h6` heading tags as appropriate. Do not skip heading levels in your page to get smaller headings (use CSS instead if appropriate).
- Your HTML and CSS files must be well-formed and successfully pass W3C HTML5 and W3C CSS3 validators with no errors (warnings are ok).

Documentation

Place a comment header in each file with your name, section, a brief description of the assignment, and the files contents. We have already provided some starter comments for you to fill out in the provided code. Examples:

HTML File:

```
<!--  
    Name: Lorem Hovik  
    CS 101 Spring 2021  
    Date: April 1st, 2021  
    This is the index.html page for my portfolio of web development work.  
    It includes links to side projects I have done during CS 101,  
    including an AboutMe page, a blog template, and a cryptogram generator.  
-->
```

CSS File:

```
/*  
 * Name: Lorem Hovik  
 * CS 101 Spring 2021  
 * Date: April 1st, 2021  
 * This is the styles.css page for my portfolio of web development work.  
 * It is used by all pages in my portfolio to give the site a consistent  
 * look and feel.  
 */
```

Debugging Tools

We strongly recommend that you use the Chrome Inspector on this assignment. This will help you see the rendered styles of each element! You will also want to become familiar with it as early as possible, as it will become particularly useful for debugging JavaScript in future assignments!

Grading

This assignment will be out of 15 points and will be distributed as follows:

- External Correctness (9 pts)
- Internal Correctness (5 pts) - The internal requirements listed in this specification are met.
- Documentation (1 pts)