(/)

CSS, basic

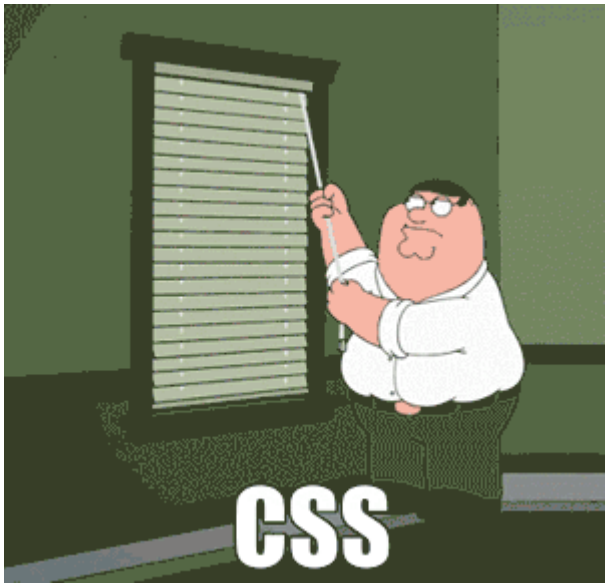
CSS

Novice

By: Guillaume Salva, CTO at Holberton School

Weight: 1

Manual QA review must be done (request it when you are done with the project)



Resources

Read or watch:

- What is CSS (/rltoken/KrIY2QooifMLy-cCKQN3xQ)
- Getting started with CSS (/rltoken/8WgVhQI-7elvmAVGktAgqQ)
- CSS Selectors (/rltoken/F8nlCHIHHf6O8eHXkLVppQ)
- Basic CSS (/rltoken/YKJ5F9KT8We_dRENkHhDNw)
- Learn to Code HTML & CSS (/rltoken/6760_a7C02TMIuSdpvPQnw)
- Specifics on CSS Specificity (/rltoken/Gdjuy58dJk6d9ozmZu1EIQ)
- CSS SpeciFishity (/rltoken/_zfWl9yfTDzST1aYXMMSOA)

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone (/rltoken/lPDHMarsO0MoHML8ETUerg), **without the help of Google**:

General

- What is CSS
- How to add style to an element
- What is a class
- What is a selector
- How to compute CSS Specificity Value
- What are Box properties in CSS
- How does the browser load a webpage

Requirements

General

- All your files should end with a new line
- A `README.md` file, at the root of the folder of the project is mandatory
- You are **not allowed** to install, import or use external libraries. This website must be build with only HTML/CSS/JavaScript. No NodeJS, React, VueJS, Bootstrap, etc.
- Your code should be W3C compliant and validate with W3C-Validator (/rltoken/2FiUndXaGuK3y3XTDerXtw)

Tasks

0. Some early styling

mandatory

Copy into this folder `index.html` and `tweets.html` that you created in the previous project (/rltoken/q8CeCpYeGdwG5JP5vN9Y5w):

- Create an empty `styles.css` .
- Create the file `base.css` and set the content of this file (/rltoken/nCsaEFm3GqaoXavk2hQ-kw)

In each of your HTML files, add these 2 lines within the `<head>` tag (do not confuse with the `<header>` tag!):

```
<link href="base.css" rel="stylesheet">
<link href="styles.css" rel="stylesheet">
```

If you refresh your webpages in your browser, they should now look a bit better!

We just told your webpages to use the CSS rules described in those two files, and to apply them to your HTML code.

Repo:

- GitHub repository: `alx_html_css`
- Directory: `css_basic`
- File: `base.css, styles.css, index.html, tweets.html`

Help

0/2 pts

1. Positioning

mandatory

Even though each element of your webpages now has a different style, you may notice they still are stacked on top of each other, and that's probably not what you want. CSS brings a few different approaches to positioning that one may use depending on cases.

For this project, we're going to use CSS Flexbox, a recent set of CSS properties that work with recent browsers.

Our goal is to get a layout that looks like this:

<table>
<tr><td colspan="2">Content of the <code><header></code> tag</td></tr>
<tr><td>Content of the <code><article></code> tag</td><td>Content of the <code><aside></code> tag</td></tr>
<tr><td colspan="2">Content of the <code><footer></code> tag</td></tr>
</table>

As a reminder, there are also two container tags playing critical roles here:

- `<main>` contains the area that includes `<article>` and `<aside>`.
- `<body>` contains all of them together.

Here are steps to get this layout, which you will have to write as CSS code in the `styles.css` file:

- Both container tags, `<body>` and `<main>` must be told that they are containers to flexible boxes: you need to apply the `display: flex` property to both of them.
- However, `<body>` contains a column of three boxes (`<header>`, `<main>` and `<footer>`), therefore you must apply the flex-direction: column property to `<body>`; whereas `<main>` contains a row of 2 boxes (`<article>` and `<aside>`), so you must apply the `flex-direction: row` property to `<main>`.
- Ensure the `<main>` tag keeps an automatic height and width, by applying the `flex: auto` property to it.

(/)

- To wrap up the layout, you want to be sure that your content (article) takes ⅔ of the width of the page, and your aside takes ⅓; you can assign to them the number of boxes they should fill in. This is done by applying the property `flex: 2` to `<article>` (using up 2 boxes), and `flex: 1` to `<aside>` (using up 1 box).
- Finally, you want to be sure that the user can scroll within your `<article>` and your `<aside>`. You can do this by applying the `overflow-y: auto` CSS property to both of them.

If you've done all of those properly, and your HTML structure is correct, you should get exactly the layout we were trying to get.

Do note that the exact rendering you're getting may be slightly different depending on your browser (namely, about whether header and footer are fixed or not when the user scrolls), but should always match the layout presented above.

Repo:

- GitHub repository: `alx_html_css`
- Directory: `css_basic`
- File: `styles.css`

Help **0/6** pts

2. Responsive web design

mandatory

You may notice that the website keeps this layout when you make your browser window smaller, or visit it from a smartphone, and it's unpleasant to use that way for your users. No worries, we covered this for you: just add the attribute `class="works_on_smartphone"` on the `<body>` tag in your `index.html` file, and the layout you just created will degrade nicely as you resize the window!

But you'll notice, if you visit your website on a smartphone, that it will still have that layout, and just seem "zoomed out". That's because you need to adjust what is called the "viewport" of the browser when rendering the page. Hint: this gets done by adding a certain tag to your HTML code.

Repo:

- GitHub repository: `alx_html_css`
- Directory: `css_basic`
- File: `index.html, styles.css`

Help **0/2** pts

3. Some more styling

mandatory

Your website is unique, and if you want it not to look like everyone else's, you may want to take some time to style everything of it as you wish!

What you're allowed to do:

- You may add any non-positioning-related CSS rules to `styles.css` (like colors, backgrounds, borders, ...)
- You may do whatever you want with the HTML content and the CSS that applies inside the `<article>` tag.
- You may add a logo to the top-left of your page. To keep it simple, rather than use an image, feel free to use a unicode character, from this table (/rltoken/Ry39Otio_47NE3N4BWRebg) for instance. One way to make it work: add a first item in the list in your `<header>`, before all other list items, and just put the HTML-code for the character in it (it starts with "&" and ends with ";"). To make it look like a logo, if you want the character to be bigger, you can add the `class="logo"` attribute on the
- tag, we added the CSS rule for you.

What you're not allowed to do:

- Do not change the layout strategy, done with CSS Flexbox, as described above. Feel free to experiment with positioning within the `<article>` tag if you wish; but please refrain to do so anywhere else.

Repo:

- GitHub repository: `alx_html_css`
- Directory: `css_basic`
- File: `index.html, styles.css`

Help

0/8 pts

Score

Please review **all the tasks** before you start the peer review.

Ready for my review

Skip this project

Previous project (/projects/2062)

(/)