Sample Web Development Project

by Devmountain Tutorials

WHAT'S COVERED

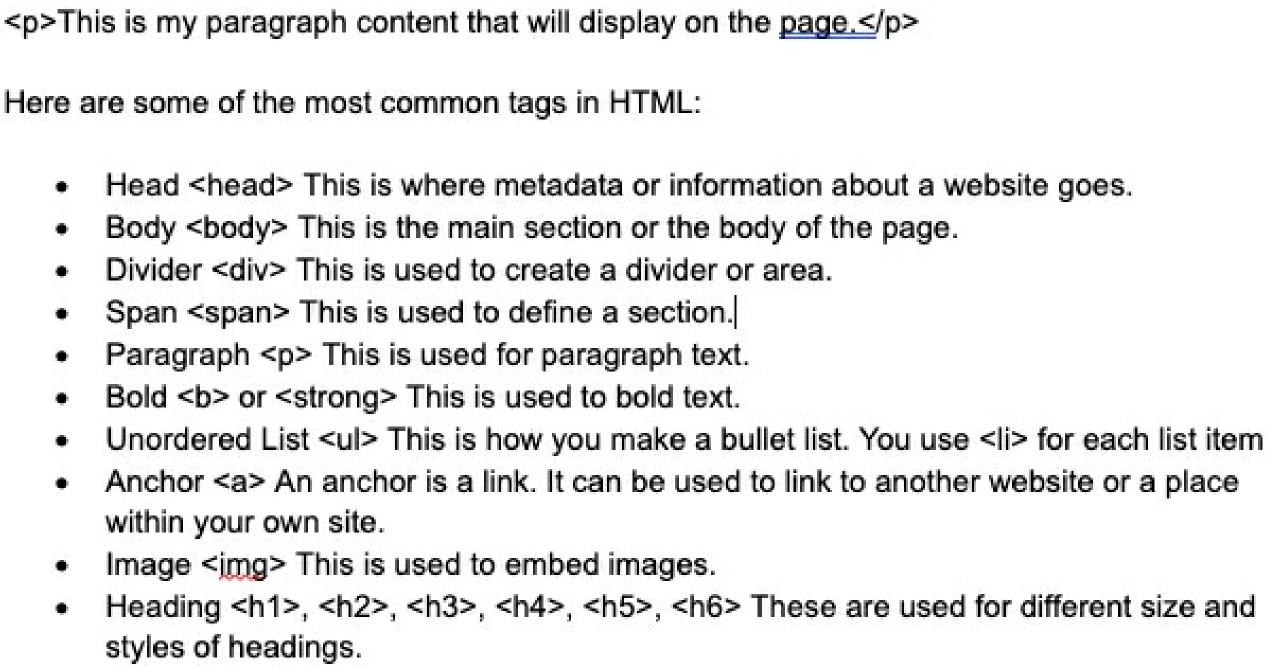
In this lesson, you will learn how to analyze the steps that the web developer takes to make sure that the website works effectively. Specifically, this lesson will cover:

1. Updating Webpage HTML

In the last section, I showed you some of the problems Web Developers encounter while building and maintaining a website. In this section, I'm going to show you more of the details that go into a web page. I'm coding a new version of the page that service providers use to sign up and manage their profiles. Camila mentioned that we are expanding services for other animals. The new version I'm creating will allow the providers to add additional details about the services they offer and the types of pets they can provide service to.

Before I start coding, I review the requirement wiki to see the new fields we are adding, and then Monique (Software Engineer) and I quickly sketch out how she will structure the data. I create a mock version of the web service that will connect to the website. This allows me to make my development changes without waiting for her part of the changes.

I make those changes to the service and then focus on the HTML changes. I look at the existing webpage to see if there are any components or modules that I can repurpose or reuse. The design Mori (UX) gave me is different from what we already have. The navigation of the site stays the same, but the profile section will just need some tweaks and then it can be repurposed. I told you a little bit about how HTML uses tags to structure the elements. There are specific tags to use when you're building different things.



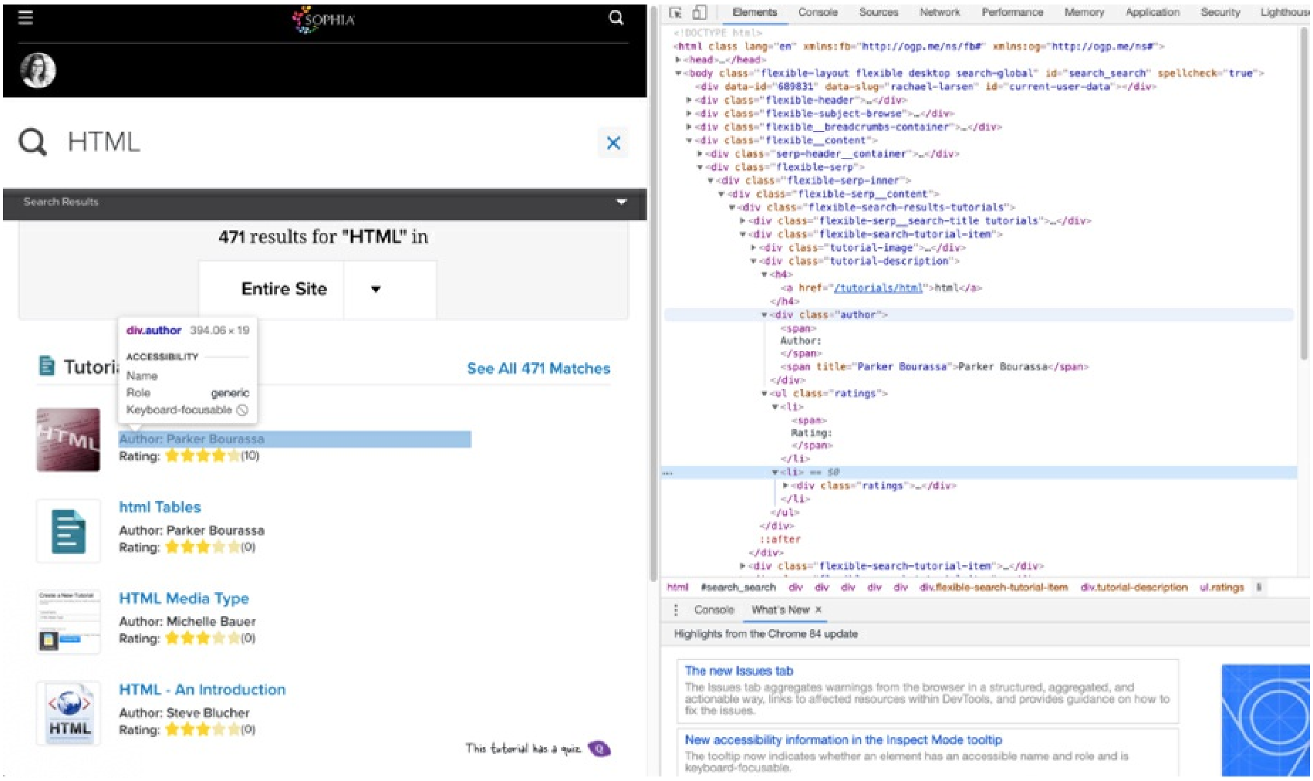
<p>This is my paragraph content that will display on the page.</p>

Here are some of the most common tags in HTML:

- Head <head> This is where metadata or information about a website goes.
- Body <body> This is the main section or the body of the page.
- Divider <div> This is used to create a divider or area.
- Span <span> This is used to define a section.
- Paragraph <p> This is used for paragraph text.
- Bold <b> or <strong> This is used to bold text.
- Unordered List <ul> This is how you make a bullet list. You use <li> for each list item
- Anchor <a> An anchor is a link. It can be used to link to another website or a place within your own site.
- Image <img> This is used to embed images.
- Heading <h1>, <h2>, <h3>, <h4>, <h5>, <h6> These are used for different size and styles of headings.

HTML Tags

Looking at a Sophia page with search results for HTML, how many of the HTML tags on the right do you recognize now? Kind of fun, huh? HTML tags are an easy way to structure documents and websites, but there are other benefits too. Screen readers and other device accessibility software use the HTML tags to assist users in navigating sites. Developers can use the alt attribute to provide alternative text for visual elements like images.



```
<!DOCTYPE html>
<html class lang="en" xmlns:fb="http://ogp.me/ns/fb#" xmlns:og="http://ogp.me/ns#">
▶<head>…</head>
▼<body class="flexible-layout flexible desktop search-global" id="search_search" spellcheck="true">
    <div data-id="689831" data-slug="rachael-larsen" id="current-user-data"></div>
  ▶<div class="flexible-header">…</div>
  ▶<div class="flexible-subject-browse">…</div>
  ▶<div class="flexible__breadcrumbs-container">…</div>
  ▼<div class="flexible__content">
    ▶<div class="serp-header__container">…</div>
    ▼<div class="flexible-serp">
      ▼<div class="flexible-serp-inner">
        ▼<div class="flexible-serp__content">
          ▼<div class="flexible-search-results-tutorials">
            ▶<div class="flexible-serp__search-title tutorials">…</div>
            ▼<div class="flexible-search-tutorial-item">
              ▶<div class="tutorial-image">…</div>
              ▼<div class="tutorial-description">
                ▼<h4>
                    <a href="/tutorials/html">html</a>
                  </h4>
                ▼<div class="author">
                    <span>
                    Author:
                    </span>
                    <span title="Parker Bourassa">Parker Bourassa</span>
                  </div>
                ▼<ul class="ratings">
                  ▼<li>
                      <span>
                      Rating:
                      </span>
                    </li>
...               ▼<li> == $0
                    ▶<div class="ratings">…</div>
                    </li>
                  </ul>
                </div>
                ::after
              </div>
            ▶<div class="flexible-search-tutorial-item">…</div>
```

Highlights from the Chrome 84 update

The new Issues tab

The Issues tab aggregates warnings from the browser in a structured, aggregated, and actionable way, links to affected resources within DevTools, and provides guidance on how to fix the issues.

New accessibility information in the Inspect Mode tooltip

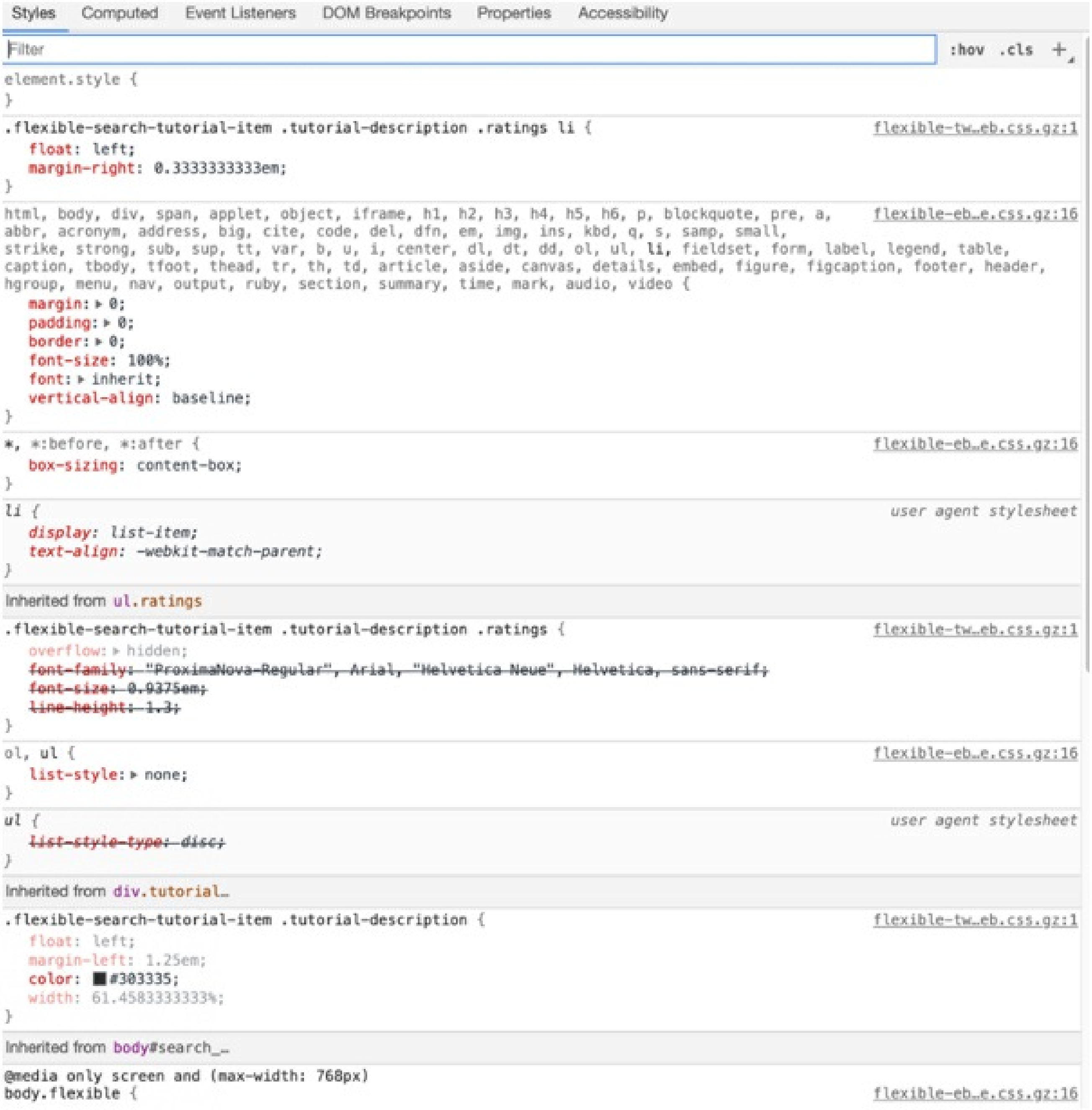
The tooltip now indicates whether an element has an accessible name and role and is keyboard-focusable.

HTML Code

2. CSS and Errors

Let’s return to my work on the Poodle Jumper website. Once the HTML elements are in place, I work on a few tweaks to the CSS. CSS is where we define the details about the font, spacing, styles, and colors for the page. The style of the new page is very similar, so I don’t have to do too many changes, but getting them to look just right takes attention to detail.

That’s the beauty of CSS, the styles cascade or flow from page to page once you’ve established the formatting. Not every web developer likes this part of building a website, but it is my favorite. The satisfaction I get when I see a beautifully executed web page is the best. Many web developers underestimate how much work it takes to get everything just right.



```
element.style {
}
.flexible-search-tutorial-item .tutorial-description .ratings li {      flexible-tw…eb.css.gz:1
   float: left;
   margin-right: 0.3333333333em;
}
html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre, a,      flexible-eb…e.css.gz:16
abbr, acronym, address, big, cite, code, del, dfn, em, img, ins, kbd, q, s, samp, small,
strike, strong, sub, sup, tt, var, b, u, i, center, dl, dt, dd, ol, ul, li, fieldset, form, label, legend, table,
caption, tbody, tfoot, thead, tr, th, td, article, aside, canvas, details, embed, figure, figcaption, footer, header,
hgroup, menu, nav, output, ruby, section, summary, time, mark, audio, video {
   margin: ▸ 0;
   padding: ▸ 0;
   border: ▸ 0;
   font-size: 100%;
   font: ▸ inherit;
   vertical-align: baseline;
}
*, *:before, *:after {      flexible-eb…e.css.gz:16
   box-sizing: content-box;
}
li {      user agent stylesheet
   display: list-item;
   text-align: -webkit-match-parent;
}
Inherited from ul.ratings
.flexible-search-tutorial-item .tutorial-description .ratings {      flexible-tw…eb.css.gz:1
   overflow: ▸ hidden;
   ~~font-family: "ProximaNova-Regular", Arial, "Helvetica Neue", Helvetica, sans-serif;~~
   ~~font-size: 0.9375em;~~
   ~~line-height: 1.3;~~
}
ol, ul {      flexible-eb…e.css.gz:16
   list-style: ▸ none;
}
ul {      user agent stylesheet
   ~~list-style-type: disc;~~
}
Inherited from div.tutorial…
.flexible-search-tutorial-item .tutorial-description {      flexible-tw…eb.css.gz:1
   float: left;
   margin-left: 1.25em;
   color: ■#303335;
   width: 61.4583333333%;
}
Inherited from body#search_…
@media only screen and (max-width: 768px)
body.flexible {      flexible-eb…e.css.gz:16
```

CSS Code

Every time you write new code, errors can happen. Errors can come from typos, syntax errors, or making an error in the logic. When a developer searches for issues and fixes them, it is called debugging. When I started developing websites, I dreaded the process of debugging. It can be difficult to know what the different error messages mean and how to fix them. I'll talk about debugging more in the next section.

I'm still waiting on the database changes from Monique (Software Engineer), she's been busy with another task, so I take the opportunity to show Mori (UX) and Jose (Product Manager) what I've I've built so far. They like the progress but had a suggestion on changing the order of the fields. When requirements change, it can be frustrating. I've learned it is part of the process, so I structure my work in small chunks to make it easier when things need to be changed. Once the database changes are complete, we'll be ready to wire it up and finish testing.

SUMMARY

This lesson explored a web development project in which the developer **updates a webpage's HTML** and CSS. The developer begins by reviewing the requirements and sketches out the structure of the data. Then, the developer repurposes some existing modules of the webpage and makes the necessary HTML changes. Afterward, the developer works on the CSS to define the font, spacing, styles, and colors for the page. Additionally, the lesson discussed how **CSS errors** can occur when writing new code, and how debugging is necessary to resolve those errors. Finally, the lesson emphasized the importance of structuring work in small chunks to make it easier to handle when requirements change.

Source: This tutorial was authored by DEVMOUNTAIN and Sophia Learning. Please see ourTerms of Use.