HTML5 Semantic Elements Lab

- Due No Due Date
- Points 1
- Submitting a website url

(https://github.com/learn-co-curriculum/phase-1-html5-containers-lab) (https://github.com/learn-co-curriculum/phase-1-html5-containers-lab/issues/new)

Learning Goals

- Introduce common semantic tags in HTML
- Explore their use by applying them to existing content

Introduction

In HTML5, there are many new tags that help us describe *what kind of content* exists within a specific tag. These are referred to as *semantic* elements. In this lesson, we will be introducing some of the most useful tags.

Non-Semantic Elements

Before we dive into semantic elements, lets see some examples of *non-semantic* elements. Two of the most commonly used HTML tags are `span` and `div` . Neither tag has automatic styling. The only difference between them: content wrapped with the `span` tag will display without line breaks, whereas content wrapped with `div` *will*:

```
<span>This content will share the same line...</span><span>...as this content</span>

<div>
  This message will appear on a new line
</div>
```

These tags certainly have their uses, and developers can sometimes favor them *because* of the lack of styling. However, they don't give any indication as to what type of content they're wrapping. They are just *dividers* of the content.

Semantic Elements

Many semantic elements also lack automatic styling, and act very similar to the `div` tag. What they provide, instead, is an explanation of what they wrap.

`<header>` and `<footer>` Tags

The first two semantic tags to discuss are the `header` and `footer` tags. The purpose of these may seem obvious to those who have used document editors like Microsoft Word. The `header` tag is used to wrap all content we would want to contain within the top, (header), portion of a page. The `footer` is for everything at the foot, (bottom), of a page:

```
<header>
  <!-- Headers often contain company logos -->
</header>

<!-- All the main content of a web page goes in between -->

<footer>
  <!-- Footers often contain resources, privacy policy links, and copyright information -->
</footer>
```

Commonly, a website with many different pages will have the same header and footer content on each page...the only content that changes is what is in between.

`<nav>` Tags

Typically, inside or just below the header section of a page are navigation links to help users access different parts of a website. For this block of links, we can use the `nav` tag. Wrapping `nav` around links helps describe those links as the page navigation itself:

```
<nav>
  <a href="about.html">About</a>
  <a href="contact.html">Contact</a>
</nav>
```

A reader glancing over an HTML page can quickly see what these links are meant for. The `nav` tag is not meant for all links, just those typically used for site navigation.

`<main>` Tag

The `main` tag specifies the *main* content of a web page. This would typically be everything in between the `header` and `footer` areas, and may contain many nested tags:

```
<header></header>
<nav></nav>

<main>
  <!-- All the main content of a web page goes here -->
</main>

<footer></footer>
```

With these few tags, common content within a web page can be separated in a way that is easy to understand.

`<section>` Tag

Within the `main` tag, we can continue to breakdown content into specific, meaningful sections. One way we can do this is to use the...well... `section` tag.

```
<section>
  <p>Lorem ipsum dolor sit amet...</p>
  <p>Lorem ipsum dolor sit amet...</p>
</section>
```

The `section` tag can be used to define specific portions of a web page. A page may have multiple boxes of content within a larger container like `main` . For each box, we can use a `section` tag to separate the content.

`<article>` and `<aside>` Tags

The `section` tag is more informative than the `div` tag, but it still may not be as specific as we need. For particular parts of a web page, we have semantic options like `article` and `aside` . The `article` tag is for containing written content such as a news story or blog post. The `aside` tag is for containing content that may be related to other content, but is better kept separated.

```
<article>
  <h1>First Human Digitizes Brain</h1>
  <p>In 2018, Chrome Boi became the first human to digitize their brain. They now live in the Internet.</p>
</article>

<aside>
  <h4>Once human, now digital</h4>
  <p>A quick visit to https://en.wikipedia.org/wiki/Draft:Chrome_Boi will show you the ascended individual</p>
</aside>
```

`<figure>` and `<figcaption>` Tags

Along with `section` , `article` , and `aside` , we also have some tags specific for containing image and media content. The `<figure>` tag wraps self-contained media content. For instance, a blog post could have an accompanying image to support the content.

The `figure` tag also comes with a companion for providing captions, the `figcaption` tag. Since `figure` is used for media, the `figcaption` tag can be used to add an additional message about that media or its source.

```
<section>
```

```
    <article>
      Lorem ipsum dolor sit amet...
    </article>

    <figure>
      <img src="images/intro-pic.jpg"  alt="An exceptional living room." title="Welcome to Exceptional Living Rooms">
      <figcaption>"An Exceptional Living Room" by Leonardo DaVinci, photograph</figcaption>
    </figure>

  </section>
```

Here, we've wrapped an image in the `<figure>` tag, and included a `<figcaption>` providing the title and creator of the image.

Practice with Semantic Elements

Let's practice what we've discussed. In `index.html` , we have a web page with some example content for a real estate agency. However, most of the HTML tags within the `body` are non-semantic `div` and `span` tags.

Your task is to read through the provided comments and add in the appropriate semantic tags. Run `npm test` to test your work and use the provided error messaging to work through the tests.

Note: there are a *few* semantic tags in `index.html` not explicitly discussed in this readme. Use the comments to figure out what tag you will have to use.

Make sure that for every `div` and `span` you replace, that you also replace the corresponding *closing* tag!

You can view `index.html` in the browser by running `httpserver` or opening the file in a separate browser tab. It is worth noting, though, that the layout of the page won't change as you add semantic tags. We are not changing the styling or structure, but the description of the content contained on the page.

Conclusion

Using semantic tags serves multiple functions. They provide a greater *readability* for yourself or anyone else who may edit an HTML document in the future. They also make it easier to *style* your pages. As a bonus, they help search engines identify and categorize content on websites.

When using Cascading Style Sheets, we can easily set up styling for *just* the specific semantic elements. These tags are more natural to write and faster to understand than `div` and `span` tags.

There are more semantic tags to explore, some of which you've already used! Tags such as `form` and `table` are semantic as well, as they describe the contents within.