Exploration — HTML Tags

Introduction

In this module, we are going to look at a small subset of tags in detail to understand how they are used. Then we will look at a larger collection of tags and talk about what they are used for in more general terms.

Block-level vs. In-line Elements

HTML elements have historically being characterized as being either block-level elements or inline elements.

- **Block-level** elements break up the flow on the content and are typically displayed by browsers with a newline both before and after the element.
- **Inline** elements do not break the flow of the content, such as a phrase within a sentence.

Example

`<p>` is block-level element, while `<span>` and `<em>` are inline elements.

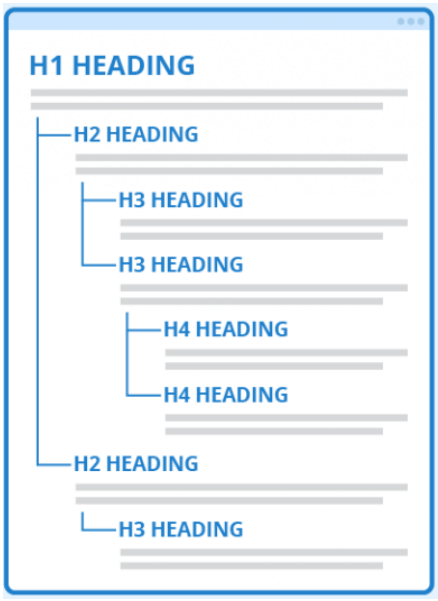
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Inline vs. Block Elements</title>
  </head>
  <body>
    <p>Span is an <span>inline element</span> and <em>doesn't</em>
break the flow of the content.</p>
```

Code Editor

This is where you write code. Code describes how programs work. The editor helps you write code, by coloring certain types, and giving you suggestions when you type. Click on one of the examples to see how code looks.

Next

Headlines h1 to h6



Text within a headline tag is represented as a headline (or heading; not to be confused with <head> or <header> (which we'll learn about later). The default style of a headline tag includes a bolder, larger font. They are used **heirarchically** in sections and articles of web pages to announce new or subsequent content. The opening and closing tags are always required. `<h1>` is used first, as highest level heading. Headngs run from `<h1>` through `<h6>` with `<h6>` being the lowest level heading that you might find deep in an article. In the document you are reading right now the title at the top of the page is an `h1` and the heading at the top of this section is an `<h2>`. If there are subsections they are `<h3>` headings. Headings are block-level elements.

If an <h1> headline is used at the top of the document, such as in a <section> element, then its <article>s will typically have <h2> level headlines with sub-topic headings <h3> and lower nested below it, as per the diagram at the right.

When to Use Headlines

If you would potentially find something listed in the table of contents, it is a great candidate for a heading. Headings should be used to add organization to a document. So any section or sub section that needs a title can use a heading for it. Some tags are designed with their own headlines, such as <figcaption> for figures/images and <caption> for tables, and <legend> for fieldsets in forms.

When Not to Use Headlines

To ensure our web pages are accessible to screen readers (for the visually impaired) and search engine algorithms, the use of headlines <h1> to <h6> must be used hirarchically to denote new or sub-level content. They are not used for any other purpose. They help structure a page, for clarity, rather than to add style to other elements. Even though they bold and perhaps larger, doesn't mean we can use them to add emphasis in a paragraph, for example.

Section, Article and Div

These elements along with a few others are used to section off content. Visually these do very little on their own. They will usually break their content into a block so that a line break comes before and after. Beyond that, how they look depends on the styles we apply to them using CSS (which we'll learn about in Module 4). Although they often all do the same thing visually, it is important to understand why they are used.

<section>

The **section element (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/section)** is used to make a thematic grouping of content. It is a group of content that is all related but does not quite stand on its own. Usually the first child of a `section` will be a headline <h1> that describes what is in that section. If there is not a good way to classify the content in the section using a heading, you may want to consider using a `article` instead. A `section` should only be used if all the content is related. **Example:** A newspaper has these sections with multiple articles in them: *Local News, Arts, Sports, Real Estate, International News*, etc.

<article>

The **article element (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/article)** is typically used inside a section and denotes a single specific topic with a second-level headline <h2>. **Example:** The *Arts* section of the newspaper includes three articles: *New Arts Complex Breaks Ground, Arts in Agriculture Exhibit at LaSells,* and *Corvallis Art Walk Thursday.*

The **div element (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/div)**, e.g., `<div>special circumstances</div>`, is a placeholder for dynamic content (such as we'll see in the

React framework we work with in Module 5). It can also be used to divide content when no other existing element makes sense. It has not style or dimensions.

Creating Links through Anchors

We create links from one page to another using **the anchor element (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/a)** , `<a>`. The content between the opening and closing `<a>` tags should describe the link when someone clicks to navigate to it. The `href` attribute is used to specify the URL where the link will take the user when it is clicked.

Examples

The following anchor element links to the URL `https://developer.mozilla.org/en-US/`. The text `MDN's Developer Website` describes the link. When someone clicks on the text, they will be taken to the page `https://developer.mozilla.org/en-US/`.

```
<a href="https://developer.mozilla.org/en-US/">MDN's Developer Website</a>
```

> **Note:** The above URL is an example of an **absolute URL** which is a complete URL to a resource including the protocol and the domain name. We can also create links using **relative URLs**. We will study the absolute and relative URLs in the next exploration.

Hyperlinks within a document

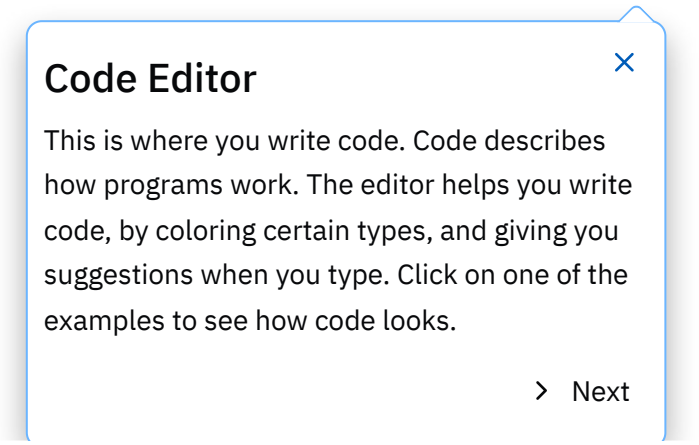
It is possible to use the **anchor** element <a href="" > to link to a specific location within a document, rather than just the top of the document. To do this:

1. Add an `id` attribute to the element we want to link to, and
2. In the URL, add the value of the `id` attribute at the end, preceded by a hash symbol `#`.

Example

In the following HTML document, we have defined three internal links, identified by the `id` values `precursors`, `browsers`, and `governance`.

```
1  <!doctype html>
2 ▼<html lang="en-US">
3
4 ▼<head>
5      <meta charset="utf-8">
6      <title>History of the Web</title>
   </head>
7
8  <body>
```



Let us say we save this document is a file named `link-example.html` on our local file system, so that the URL of the document is as follows `file:///C:/Users/nauman/m3/link-example.html`. If we open this file in a browser, the document will be displayed starting at the top of the document. However, if we enter the following URL in the browser `file:///C:/Users/nauman/m3/link-example.html#governance`, then the browser will scroll down with the element with the ID value `governance` displayed in the browser.

Images

The use of images (beyond standard icon libraries) won't be required in this course, but it is important to understand the basics, if you want to embellish the frontend design of your work.

The `<img>` tag is used to display photography and illustrations. `<img src="kitten.png" alt="A white kitten is curled up next to a pillow">`. The `src` attribute is required and indicates the URL where the image is located. The `alt` attribute is an accessibility feature that specifies an alternative detailed description of the imagery in the photo/illustration, which is read by screen-readers to aid the visually impaired. Alternaitve text is required for websites that must adhere to American Disability Act (ADA). The <img> tag self-closes with > or /> and displays inline rather than block; it *does not* automatically make a new line. Other options for photography and

illustration are detailed in **A Guide to the Responsive Images Syntax in HTML. (https://css-tricks.com/a-guide-to-the-responsive-images-syntax-in-html/)**

So that image files load quickly, it is important to optimize them. Optimized images meet these specifications:

Descriptive file name
To improve search results, file names should include who, what, when, where as much as possible.

Small file size
Keep file size as small as possible for the fastest load time. Serve high resolution images only to high resolution devices.

Exact dimensions
Crop and reduce the size of images to fit the dimensions of the space in your web page. This might mean several sizes are needed.

Correct file format
Online photos are usually .JPG.
Flat-color, line-art images like logos and clipart are usually .GIF and sometimes 8-bit .PNG.
Graphics with true transparency need 24-bit .PNG.

Reduced resolution
Monitors render between 72 or 300+ pixels per inch (ppi); the older default is 72ppi. Much higher resolutions are available now, so providing multiple image sizes for each is a standard.

Color Mode
RGB for .PNG, .JPG, .SVG, and .WebP, and Indexed for .GIF.

, <b>,

This family of elements is for adding meaning or style to specific text within a paragraph. This is different from the previous elements which generally served to group text.

The `strong` element marks text that is more important, e.g., `<strong>This is important</strong>`. The tag is bold by default and it provides voice inflection when read by a screen reader (for the visually impaired). It requires an opening and closing tag. It is *not* used in place of headlines.

<b>

The `<b>` element makes text stylistically different from other text. `Look at me, I am <b>different</b>`. This could be used to highlight keywords in a paragraph or conform to style guidelines set by some journal. It does not give additional meaning or provide voice inflection, so it not commonly used anymore.

The `<em>` element adds emphasis to a word to improve its meaning in a sentence. By default, it is italicized and provides voice inflection for screen readers. Italics are additionally used for article and artwork titles, however, adding an *anchor* to the title instead, will underline it and allow linking to its parent website. When no website exists to link to, then using works well. The <i> element *used to be used* for italics, but it is now used for denoting *icons*.

Lists and Tables

Lists and Tables are more complex than the previous elements we have learned about so far. They use more than one element nested within their parent element.

Lists

Lists come in three flavors: ordered, unordered, and definition list.

 for unordered lists

An **unordered list (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/ul)** is created using the tag `ul`. It can have one or more line items `<li>` nested between its opening and closing tags and can *only* have `<li>`s as children, like this:

```
<ul>
  <li>first item</li>
  <li>second item</li>
  <li>third item</li>
</ul>
```

The above markup for the unordered list displays like this:

- first item
- second item
- third item

 for ordered lists

Ordered lists (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/ol) are automatically numbered or otherwise sequenced, unlike unordered lists. For this, we use the tag `ol` instead of `ul`. Here is the same list, now ordered, i.e., with `ol` instead of `ul`:

1. first item
2. second item
3. third item

<dl> for definition lists

Definition lists (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/dl), using tag `<dl>`, are similar but have pairs of terms <dt> and definitions/descriptions <dd> and are nested like this:

```
<dl>
  <dt>MDN</dt>
  <dd>Developer guides for all things web.</dd>
  <dt>CSS Tricks</dt>
  <dd>Frontend web developer guides.</dd>
  <dt>World Wide Web Consortium</dt>
  <dd>Documents web specifications.</dd>
  <dd>Coordinates interoperability standards.</dd>
  <dd>Advicates for accessibility, internationalization, web security, and privacy.</dd>
</dl>
```

This is rendered as:

MDN
Developer guides for all things web.
CSS Tricks
Frontend web developer guides.
World Wide Web Consortium
Documents web specifications.
Coordinates interoperability standards.
Advicates for accessibility, internationalization, web security, and privacy.

Tables

One of the most complex sets of nested elements, which we will use a lot in this course, is **the `<table>` element** (https://developer.mozilla.org/en-US/docs/Web/HTML/Element/table).

Here is an sample table from the W3C standard that we examine in detail:

```
<table>
  <caption>Positive and Negative Characteristics</caption>
  <thead>
    <tr>
      <th>Characteristic</th>
      <th>Negative</th>
      <th>Positive</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Mood</td>
      <td>Sad</td>
      <td>Happy</td>
    </tr>
    <tr>
      <td>Grade</td>
      <td>Failing</td>
      <td>Passing</td>
    </tr>
  </tbody>
</table>
```

And here is what that table looks like when it is rendered in your browser:

Positive and Negative Characteristics

Characteristic	Negative	Positive
Mood	Sad	Happy
Grade	Failing	Passing

<table>

This parent tag opens and closes the table. Its child tags will be nested between its opening and closing tags.

<caption>

This element represents the title and/or description of the data in the table, such as its purpose or result. It can be placed above the thead or below the tfoot.

<thead>

This is the table header, which nests the labels/headings for each column and/or row using `<th>` nested within a row(s) `<tr>`.

<tbody>

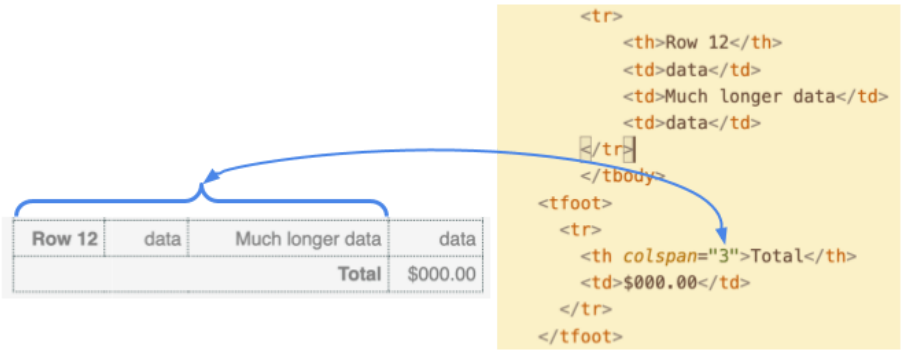
The table's body contains rows and columns of data and typically use the row `<tr>` with data `<td>` elements.

<tr>

Denotes a row in the table. It contains either data `<td>` elements or header `<th>` elements. A row must have either a data or header element to complete a "cell" in the table.

<td>

Denotes a data cell in a column of data in the table. It can include the column span `colspan` attribute (or row span `rowspan` if a cell must span more than one column or row. The number of spanned columns plus the number of normal columns must be equal to the number of `<th>` or `<th>` columns specified in the `<thead>` at the top of the table.



Read more about colspan and rowspan. (https://developer.mozilla.org/en-US/docs/Learn/HTML/Tables/Basics)

<th>

Denotes a label/heading cell in a column of data in the table. It gets nested within a row(s) of the `<thead>`.

<tfoot>

This is the table footer, which is placed below the `<tbody>`. It includes one or more row with data cells. It is a great location for the results of calculations below columns.

Exercise

Try to make a moderately complex table. To do this you will certainly want a reference to look up all the options. I strongly recommend MDN's **table content reference (https://developer.mozilla.org/en-US/docs/Web/HTML/Element#table_content)**. Your task is to as close as possible, recreate the table **shown here (https://m334.coecs290.repl.co/)**.

Go to **this replit (https://replit.com/@coecs290/m333)** and add HTML elements in the `index.html` file that will create the example table. If you get stuck, take a look at the solution: **index.html. (https://replit.com/@coecs290/m334)**

Note that a Cascading Style Sheet (CSS) file is provided in the replit. Since the table elements do not provide borders by default, it will be hard to read the rows/colums/cells without the stylesheet's border properties. When adding borders to the row and column elements' selectors, we can see all four sides of each cell. We'll learn more about that in Module 4.

Summary

This should give you a good understanding of the basic HTML elements. There are many other elements, but these are the common ones to start with. We'll use sections, articles, paragraphs, lists, and tables in future assignments.

Additional Resources

- MDN is an excellent reference for learning more about **HTML elements (https://developer.mozilla.org/en-US/docs/Web/HTML/Element)**.
- For a listing of block and inline elements, see **the page on W3Schools website (https://www.w3schools.com/html/html_blocks.asp)**.