Intro to HTML

WEB TECHNOLOGIES

Behind every web page you see (and that you'll create) are three distinct technologies:

- **HTML** (Hyper Text Markup Language)
- **CSS** (Cascading Stylesheets)
- **JavaScript** (aka ECMAScript or ES)

Each of these technologies plays a distinct role. The analogy of a human body is often used to illustrate their roles.

HTML provides the skeleton of the web page. HTML is all about *structure*.

CSS allows us to *style* the HTML. It deals with questions of where and how things appear on a page.

JavaScript lets the page come to life with *actions*. HTML and CSS are limited in their abilities. With JavaScript, the possibilities for what a webpage can do are almost limitless.

A HIERARCHY OF TAGS

An HTML page is nothing more than a text file in which the web developer has added *tags* chosen from a pre-defined set. Tag names are wrapped in *angle brackets* and usually exist in pairs.

Opening tag
Child tags go in between
Closing tag

```
<html>

</html>
```

Different tag names provide different functionality. Here is a sampler of a few commonly-used tags.

Outermost (parent) tag

<head> section encloses tags that provide information about the page

<body> section encloses tags that display the page

Largest of 6 header tags

Paragraph

Image

Unordered list

List items

Text input box

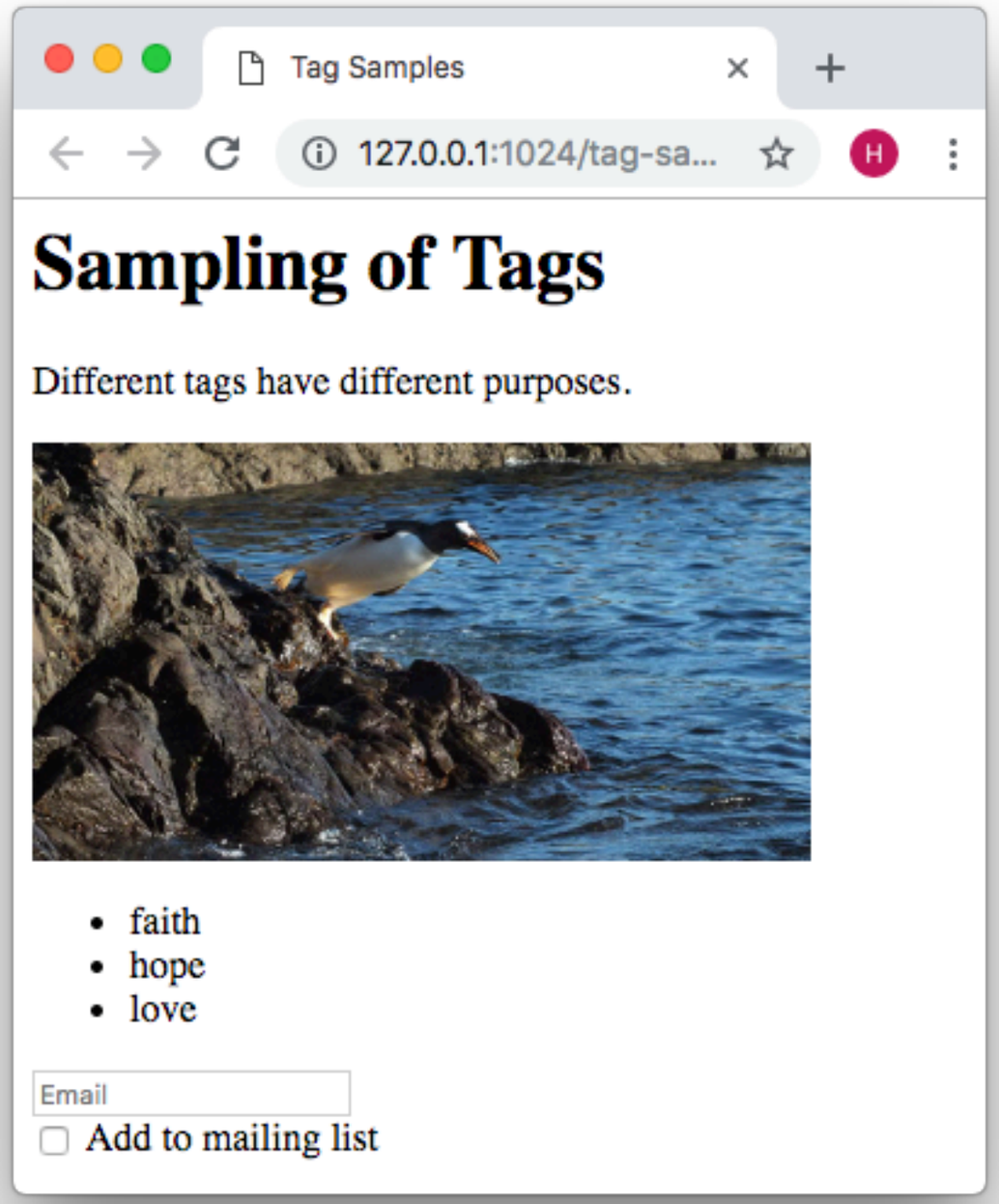
Line break

Checkbox

Form label

```
<html>

<head>
  <title>Tag Samples</title>
</head>

<body>
  <h1>Sampling of Tags</h1>
  <p>Different tags have different purposes.</p>
  <img src="./images/penguins_feat.jpg" width="320">

  <ul>
    <li>faith</li>
    <li>hope</li>
    <li>love</li>
  </ul>

  <input type="text" placeholder="Email">
  <br>
  <input type="checkbox">
  <label>Add to mailing list</label>
</body>
```

As displayed by the browser with no styling (CSS)



How do we get from typing a website URL into our browser to a page being displayed? The fascinating answer is found in the Guide: From HTML to the Browser

TAG ATTRIBUTES

For a full list of HTML tags and their associated attributes, go to: https://www.w3schools.com/tags/ref_byfunc.asp

Some tags are so obvious, they need no further information (*e.g.* the `<br>` tag). Others, though, may need addition information to provide clarification to the browser as to how to handle them. Take the `<img>` tag. The name of the tag lets the browser know that it should display an image. But *which* image should be displayed?

To provide the browser additional information needed to properly process a tag, web developers use *tag attributes*. These take the form of a *key* and a *value* linked with the = sign.

Tag attribute pattern

```
<tagname key="value">
```

`src` attribute added to `<img>` tag provides location information on the image file

```
<img src="./images/penguins_feat.jpg">
```

Tags (and attributes) aren't arbitrary — they're maintained by the World Wide Web Consortium (W3C). While the W3C sets the standard, various browser makers (Google, Apple, Microsoft...) are responsible for implementing the standard and the interpretation of it can vary between makers.

THE `id` AND `class` ATTRIBUTES

While most attributes are specific to only certain tags, two attributes can be used for *any* HTML tag. Those attributes are `id` and `class`. You'll be using this very commonly when writing HTML code.

The `id` attribute will be used by both CSS and JavaScript to select a *single* element while the `class` attribute selects *multiple* elements. Why would you want to "select" an element at all? Let's say you're working on an HTML page for an online shopping site that you've almost finished. Your boss (or client) sees it and asks you, "Can you make all the prices appear as red? And bold? Yeah, that would be great..."

Off they stroll. You look at the page. There are dozens of prices scattered around the page. "Ah!", you say. "I know what I'll do. I'll use CSS!"

That's a great idea. But for CSS to work its magic, it needs to be pointed towards the portion/s of the page you want to affect. What to do? Let's look at a portion of the code on the page you've been working on...

```
<p class="description">Green widget, 2mm<p>
<p class="price">3.75</p>
<p class="description">Green widget, 3.5mm
<p class="price">4.55</p>
```

Having had the forethought to provide class attributes for your items, your CSS becomes trivial.

The "dot class-name" pattern identifies all elements on a page with a `class` attribute of `price`

```
.price {
 color: red;
 font-weight: bold;
}
```

Although you may not know CSS (yet!), you can probably get a sense of what's going on: the "dot class-name" pattern identifies all elements on the page that have a `class` attribute with a value of `price`. Then the CSS rules (change the `color` to `red` and the `font-weight` to `bold`) are applied to all elements on the page. As you'll come to learn about CSS, you'll be dazzled by the variety of things you can do — everything from changing *where* items appear on the page to toggling the *visibility* of a particular element to adding *rounded corners* to — well, I did say it was a dazzling variety.

For a taste of just how radically different CSS can transform exactly the same HTML structure, check out http:csszengarden.com

The difference between the `class` attribute we've been looking at and the `id` attribute is simple: many tags can all share the same `class`es but an `id` can only be used once on any page. You use `id`s when you want to be able to pick a *single* tag out of a page. You use `class`es when you want to affect multiple tags. If this still seems unclear, that's understandable. You really need to see how CSS and JavaScript use both of these attributes to understand where and why you would provide them in your HTML markup.

HTML BOILERPLATE

While the HTML specification is set by the Worldwide Web Consortium (W3C), its implementation is left to browser makers such as Google, Firefox, Apple, and Microsoft. In general, browsers are very forgiving in their interpretation of HTML code. Still, to ensure your code looks as good as possible in all browsers, we recommend you start off your web pages with some boilerplate code that you can then adapt as you need to.

```
!DOCTYPE html
<html
 <head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <meta http-equiv="X-UA-Compatible" content="ie-edge">
   <title>Tab title here</title>
 </head>
 <body>
   <!-- Your display tags here -->
 </body>
</html>
```

Contents of the `<title>` section appear in the browser's tab

Code formatted like this is known as a "comment". Browsers will not display comments. You should use comments to clarify your code when its use is not obvious.

WRITING YOUR HTML CODE

We have a guide for using VS Code: Intro to VS Code.

Because HTML is simply a text file, you can use any pure text editor to write code (although something like Microsoft Word is problematic as its output is not pure text). Various tool makers have provided specialized editors to make programming easier. They often include auto-completion, tool tips, syntax helps, *etc*.

We recommend using Microsoft's Visual Studio Code. This is a state-of-the-art code editor — and it's free. You can download it for Windows™, MacOs™, and Linux at `https://code.visualstudio.com/`.