# WEB DEVELOPMENT AND DESIGN

## MODULE 1 – INTRO TO HTML

# Table of Contents

Section 1. Course Overview....2
Section 2. HTML Basics....2
Lesson 1: What is HTML?....2
Lesson 2: Say Hello to HTML Elements....3
Lesson 3: Headline with the h2 Element....3
Lesson 4: Paragraph Element....4
Lesson 5: Placeholder Text....4
Lesson 6: Comments....5
Section 3. Introduction to HTML5 Elements....5
Lesson 1: Add Images to Your Website....6
Lesson 2: Link to External Pages with Anchor Elements....7
Lesson 3: Link to Internal Sections of a Page with Anchor Elements....7
Lesson 4: Nest an Anchor Element within a Paragraph....8
Lesson 5: Make Dead Links Using the Hash Symbol....8
Lesson 6: Turn an Image into a Link....8
Lesson 7: Create a Bulleted Unordered List....9
Section 4. HTML Forms....9
Lesson 1: Create a Text Field....9
Lesson 2: Add Placeholder Text to a Text Field....10
Lesson 3: Create a Form Element....10
Lesson 4: Add a Submit Button to a Form....10
Lesson 5:HTML5 to Require a Field....10
Lesson 6: Create a Set of Radio Buttons....11
Lesson 7: Create a Set of Checkboxes....11
Lesson 7: Use the value attribute with Radio Buttons and Checkboxes....12
Lesson 8: Check Radio Buttons and Checkboxes by Default....13
Lesson 9: Nest Many Elements within a Single div Element....13
Lesson 10: Declare the Doctype of an HTML Document....13
Section 5. HTML Document Structure....14
Lesson 1: Define the Head and Body of an HTML Document....14
Further Readings....16

# Section 1. Course Overview

In this first section, we're going to explore what you'll learn in this course. You'll learn the Basics of HTML (*HyperText Markup Language)* and we'll explore what's covered in this class.

# Section 2. HTML Basics

## Lesson 1: What is HTML?

**HTML**, or *HyperText Markup Language*, is a markup language used to describe the structure of a web page. It uses a special syntax or notation to organize and give information about the page to the browser. Elements usually have opening and closing tags that surround and give meaning to content. For example, there are different tag options to place around text to show whether it is a heading, a paragraph, or a list.

For example:

```
<h1>Top level heading: Maybe a page title</h1>

<p>A paragraph of text. Some information we would like to communicate to

<ol>
  <li>Number one on the list</li>
  <li>Number two</li>
  <li>A third item</li>
</ol>
```

Becomes:

# Top level heading: Maybe a page title

A paragraph of text. Some information we would like to communicate to the user. This can be as long or short as we would like.

1. Number one on the list
2. Number two
3. A third item

The *HyperText* part of **HTML** comes from the early days of the web and its original use case. Pages usually contained static documents that contained references to other documents. These references contained hypertext links used by the browser to navigate to the reference document so the user could read the reference document without having to manually search for it.

As web pages and web applications grow more complex, the W3 Consortium updates the HTML specification to ensure that a webpage can be shown reliably on any browser. The latest version of HTML is HTML5.

This section introduces how to use HTML elements to give structure and meaning to your web content

## Lesson 2: Say Hello to HTML Elements

Welcome to HTML coding. These will walk you through web development step-by-step.

First, you'll start by building a simple web page using HTML. You can edit code in your code editor.

Do you see the code <h1>Hello</h1>? That's an HTML element.

Most HTML elements have an opening tag and a closing tag.

*Opening* tags look like this:

<h1>

*Closing* tags look like this:

</h1>

The only difference between opening and closing tags is the forward slash after the opening bracket of a closing tag.

## Lesson 3: Headline with the h2 Element

Over the next few lessons, we'll build an HTML5 Personal Portfolio piece-by-piece.

The *h2* element you will be adding in this step will add a level *two* heading to the web page.

This element tells the **browser** about the structure of your website. h1 elements are often used for main headings, while h2 elements are generally used for subheadings. There are also h3, h4, h5 and h6 elements to indicate different levels of subheadings.

***Challenge #1*:**

Add an h2 tag that says "About Me" to create a second HTML element below your "Hello World" h1 element.

## Lesson 4: Paragraph Element

p elements are the preferred element for paragraph text on websites. p is short for "paragraph".

You can create a paragraph element like this:

I'm a p tag!

***Challenge #2:***

Create a p element below your h2 element, and give it the text "Hello Paragraph".

**Note:** As a convention, all HTML tags are written in lowercase, for example  and not <P></P>.

## Lesson 5: Placeholder Text

Web developers traditionally use *lorem ipsum* text as placeholder text. The *lorem ipsum* text is randomly scraped from a famous passage by Cicero of Ancient Rome.

*Lorem ipsum* text has been used as placeholder text by typesetters since the 16th century, and this tradition continues on the web.

Well, 5 centuries is long enough, let's use something called "*perso ipsum text*".

***Challenge #3:***

*Replace* the text inside your *p* element with the first few words of this lorem ipsum text: *perso ipsum* dolor sit amet, shed everywhere shed everywhere stretching attack your ankles chase the red dot, hairball run catnip eat the grass sniff.

## Lesson 6: Comments

Commenting is a way that you can leave comments for other developers within your code without affecting the resulting output that is displayed to the end user.

Commenting is also a convenient way to make code inactive without having to delete it entirely.

Comments in HTML start with <!-- and end with a -->

Remember that in order to start a comment, you need to use <!-- and to end a comment, you need to use -->

Here you'll need to end the comment before your h2 element begins.

***Challenge #4:***

Comment out your h1 element and your p element, but not your h2 element.

# Section 3. Introduction to HTML5 Elements

HTML5 introduces more descriptive HTML tags. These include main, header, footer, nav, video, article, section and others.

These tags give a descriptive structure to your HTML, make your HTML easier to read, and help with Search Engine Optimization (SEO) and accessibility. The main HTML5 tag helps search engines and other developers find the main content of your page.

Example usage, a main element with two child elements nested inside it:

```
<main>
  <h1>Hello World</h1>
  <p>Hello Paragraph</p>
</main>
```

Note: Many of the new HTML5 tags and their benefits are covered in the Applied Accessibility section.

*Challenge #5:*

Create a second p element after the existing p element with the following perso ipsum text: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Or you can generate your own by visiting this link.

Then, create a main element and nest the two p elements inside the main element.

## Lesson 1: Add Images to Your Website

You can add images to your website by using the img element, and point to a specific image's URL using the src attribute.

An example of this would be:

<img src="https://www.your-image-source.com/your-image.jpg">

Note that img elements are self-closing.

All img elements must have an alt attribute. The text inside an alt attribute is used for screen readers to improve accessibility and is displayed if the image fails to load.

**Note:** If the image is purely decorative, using an empty alt attribute is a best practice.

Ideally the alt attribute should not contain special characters unless needed.

Let's add an alt attribute to our img example above:

<img src="https://www.your-image-source.com/your-image.jpg" alt="Author standing on a beach with two thumbs up.">

Let's try to add an image to our website:

Within the existing main element, insert an img element before the existing p elements.

Now set the src attribute so that it points to this url:

https://bit.ly/2FUlHz7

Finally, don't forget to give your img element an alt attribute with applicable text.

## Lesson 2: Link to External Pages with Anchor Elements

You can use a (anchor) elements to link to content outside of your web page.

a elements need a destination web address called an href attribute. They also need anchor text. Here's an example:

<a href="https://www.google.com">this links Google</a>

Then your browser will display the text "this links to google" as a link you can click. And that link will take you to the web address https://www.google.com.

*Challenge #6:*

Create an element that links to https://www.google.com and “never gonna give you up never gonna let you down” as its anchor text.

## Lesson 3: Link to Internal Sections of a Page with Anchor Elements

a (anchor) elements can also be used to create internal links to jump to different sections within a webpage.

To create an internal link, you assign a link's href attribute to a hash symbol # plus the value of the id attribute for the element that you want to internally link to, usually further down the page. You then need to add the same id attribute to the element you are linking to. An id is an attribute that uniquely describes an element.

Below is an example of an internal anchor link and its target element:

<a href="#contacts-header">Contacts</a>
...
<h2 id="contacts-header">Contacts</h2>
When users click the Contacts link, they'll be taken to the section of the webpage with the Contacts header element.

Change your external link to an internal link by changing the href attribute to "#footer" and the text from "Photos" to "Jump to Bottom".

Remove the target="_blank" attribute from the anchor tag since this causes the linked document to open in a new window tab.
Then add an id attribute with a value of "footer" to the <footer> element at the bottom of the page.

## Lesson 4: Nest an Anchor Element within a Paragraph

You can nest links within other text elements.

Here's a <a target="_blank" href="http://www.google.com”> link to google.com</a> for you to follow.

Let's break down the example: Normal text is wrapped in the p element:
 Here's a ... for you to follow.  Next is the anchor element <a> (which requires a closing tag </a>):
<a> ... </a> target is an anchor tag attribute that specifies where to open the link and the value "_blank" specifies to open the link in a new tab href is an anchor tag attribute that contains the URL address of the link:
<a href="http://_www.google.com "> ... </a> The text, "link to freecodecamp.org", within a element called anchor text, will display a link to click:
<a href=" ... ">link to google.com </a> The final output of the example will look like this:

Here's a link to google.com for you to follow.

## Lesson 5: Make Dead Links Using the Hash Symbol

Sometimes you want to add a elements to your website before you know where they will link.

This is also handy when you're changing the behavior of a link using JavaScript, which we'll learn about later.

---

***Challenge #7:***

The current value of the href attribute is a link that points to "https://bit.ly/2FUlHz7". Replace the href attribute value with a #, also known as a hash symbol, to create a dead link.

For example: href="#"

## Lesson 6: Turn an Image into a Link

You can make elements into links by nesting them within an a element.

Nest your image within an a element. Here's an example:

<a href="#"><img src="https://bit.ly/2FUlHz7" alt="Never gonna give you up never gonna let you down never gonna turn around and dessert you"></a>

Remember to use # as your a element's href property in order to turn it into a dead link.

---

***Challenge #8:***

Place the existing image element within an a (*anchor*) element.

Once you've done this, hover over your image with your cursor. Your cursor's normal pointer should become the link clicking pointer. The photo is now a link.

## Lesson 7: Create a Bulleted Unordered List

HTML has a special element for creating *unordered lists*, or bullet point style lists.

Unordered lists start with an opening <ul> element, followed by any number of <li> elements. Finally, unordered lists close with a </ul>

For example:

```
<ul>
  <li>milk</li>
  <li>cheese</li>
</ul>
```

would create a bullet point style list of "milk" and "cheese".

---

***Challenge #9:***

Remove the last two p elements and create an unordered list of three things that you love at the bottom of the page.

# Section 4. HTML Forms

## Lesson 1: Create a Text Field

Now let's create a web form.

input elements are a convenient way to get input from your user.

You can create a text input like this:

<input type="text">

Note that input elements are self-closing.

---

*Challenge #10:*

Create an input element of type text below your lists.

## Lesson 2: Add Placeholder Text to a Text Field

Placeholder text is what is displayed in your input element before your user has inputted anything.

You can create placeholder text like so:

<input type="text" placeholder="this is placeholder text">

**Note:** Remember that input elements are self-closing.

---

*Challenge #11:*

Set the placeholder value of your text input to "your photo URL".

## Lesson 3: Create a Form Element

You can build web forms that actually submit data to a server using nothing more than pure HTML.

You can do this by specifying an action on your form element.

For example:

<form action="/url-where-you-want-to-submit-form-data"></form>

---

*Challenge #12:*

Nest your text field inside a form element, and add the action="https://bit.ly/2FUlHz7" attribute to the form element.

## Lesson 4: Add a Submit Button to a Form

Let's add a submit button to your form. Clicking this button will send the data from your form to the URL you specified with your form's action attribute.

Here's an example submit button:

<button type="submit">this button submits the form</button>

---

*Challenge #13:*

Add a button as the last element of your form element with a type of submit, and "Submit" as its text.

## Lesson 5:HTML5 to Require a Field

You can require specific form fields so that your user will not be able to submit your form until he or she has filled them out.

For example, if you wanted to make a text input field required, you can just add the attribute required within your input element, like this: <input type="text" required>

---

*Challenge #14:*

Make your text input a required field, so that your user can't submit the form without completing this field.

Then try to submit the form without inputting any text. See how your HTML5 form notifies you that the field is required?

## Lesson 6: Create a Set of Radio Buttons

You can use *radio buttons* for questions where you want the user to only give you one answer out of multiple options.

Radio buttons are a type of input.

Each of your radio buttons can be nested within its own label element. By wrapping an input element inside of a label element, it will automatically associate the radio button input with the label element surrounding it.

All related radio buttons should have the same name attribute to create a radio button group. By creating a radio group, selecting any single radio button will automatically deselect the other buttons within the same group ensuring only one answer is provided by the user.

Here's an example of a radio button:

```
<label>
  <input type="radio" name="indoor-outdoor">Indoor
</label>
```

It is considered best practice to set a for attribute on the label element, with a value that matches the value of the id attribute of the input element. This allows assistive technologies to create a linked relationship between the label and the child input element. For example:

```
<label for="indoor">
  <input id="indoor" type="radio" name="indoor-outdoor">Indoor
</label>
```

Add a pair of radio buttons to your form, each nested in its own label element. One should have the option of indoor and the other should have the option of outdoor. Both should share the name attribute of indoor-outdoor to create a radio group.

## Lesson 7: Create a Set of Checkboxes

Forms commonly use *checkboxes* for questions that may have more than one answer.

Checkboxes are a type of input.

Each of your checkboxes can be nested within its own label element. By wrapping an input element inside of a label element it will automatically associate the checkbox input with the label element surrounding it.

All related checkbox inputs should have the same name attribute.

It is considered best practice to explicitly define the relationship between a checkbox input and its corresponding label by setting the for attribute on the label element to match the id attribute of the associated input element.

Here's an example of a checkbox:

<label for="loving"><input id="loving" type="checkbox" name="personality"> Loving</label>

---

*Challenge #14:*

Add to your form a set of three checkboxes. Each checkbox should be nested within its own label element. All three should share the name attribute of personality.

## Lesson 7: Use the value attribute with Radio Buttons and Checkboxes

When a form gets submitted, the data is sent to the server and includes entries for the options selected. Inputs of type radio and checkbox report their values from the value attribute.

For example:

```
<label for="indoor">
  <input id="indoor" value="indoor" type="radio" name="indoor-outdoor">Indoor
</label>
<label for="outdoor">
  <input id="outdoor" value="outdoor" type="radio" name="indoor-outdoor">Outdoor
</label>
```

Here, you have two radio inputs. When the user submits the form with the indoor option selected, the form data will include the line: indoor-outdoor=indoor. This is from the name and value attributes of the "indoor" input.

If you omit the value attribute, the submitted form data uses the default value, which is on. In this scenario, if the user clicked the "indoor" option and submitted the form, the resulting form data would be indoor-outdoor=on, which is not useful. So, the value attribute needs to be set to something to identify the option.

*Challenge #15:*

Give each of the radio and checkbox inputs the value attribute. Use the input label text, in lowercase, as the value for the attribute.

## Lesson 8: Check Radio Buttons and Checkboxes by Default

You can set a checkbox or radio button to be checked by default using the checked attribute.

To do this, just add the word "checked" to the inside of an input element. For example:

<input type="radio" name="test-name" checked>

---

*Challenge 16:*

Set the first of your radio buttons and the first of your checkboxes to both be checked by default.

## Lesson 9: Nest Many Elements within a Single div Element

The div element, also known as a division element, is a general-purpose container for other elements. The div element is probably the most commonly used HTML element of all.

Just like any other non-self-closing element, you can open a div element with  and close it on another line with .

---

*Challenge #17:*

Nest your "Things i love" and "Things i hate" lists all within a single div element.

**Hint:** Try putting your opening div tag above your "Things i love" p element and your closing div tag after your closing ol tag so that both of your lists are within one div.

## Lesson 10: Declare the Doctype of an HTML Document

The challenges so far have covered specific HTML elements and their uses. However, there are a few elements that give overall structure to your page, and should be included in every HTML document.

At the top of your document, you need to tell the browser which version of HTML your page is using. HTML is an evolving language, and is updated regularly. Most major browsers support the latest specification, which is HTML5. However, older web pages may use previous versions of the language.

You tell the browser this information by adding the <!DOCTYPE ...> tag on the first line, where the ... part is the version of HTML. For HTML5, you use <!DOCTYPE html>.

The ! and uppercase DOCTYPE is important, especially for older browsers. The html is not case sensitive.

Next, the rest of your HTML code needs to be wrapped in html tags. The opening <html> goes directly below the <!DOCTYPE html> line, and the closing </html> goes at the end of the page.

Here's an example of the page structure:

```
<!DOCTYPE html>
<html>
  <!-- Your HTML code goes here -->
</html>
```

Add a DOCTYPE tag for HTML5 to the top of the blank HTML document in the code editor. Under it, add opening and closing html tags, which wrap around an h1 element. The heading can include any text.

# Section 5. HTML Document Structure

## Lesson 1: Define the Head and Body of an HTML Document

You can add another level of organization in your HTML document within the html tags with the head and body elements. Any markup with information about your page would go into the head tag. Then any markup with the content of the page (what displays for a user) would go into the body tag.

Metadata elements, such as link, meta, title, and style, typically go inside the head element.

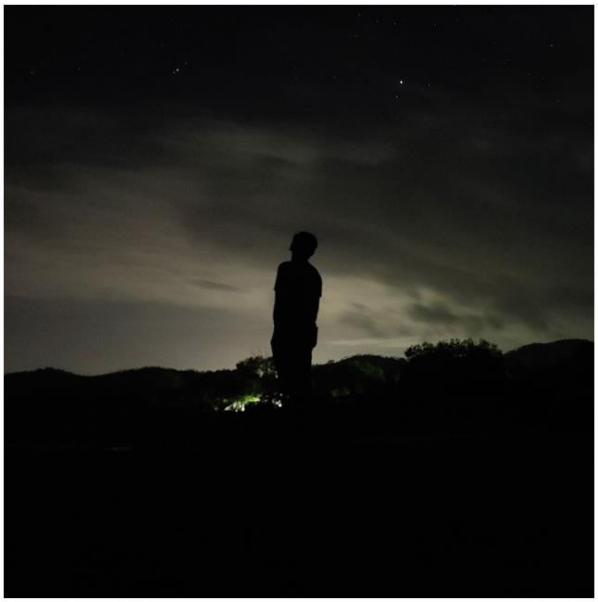
Here's an example of a page's layout:

```
<!DOCTYPE html>
<html>
  <head>
    <!-- metadata elements -->
  </head>
  <body>
    <!-- page contents -->
  </body>
</html>
```

Edit the markup so there's a head and a body. The head element should only include the title, and the body element should only include the h1 and p.

## Activity #1

For this activity apply what you have learned from the previous discussions to recreate the web page below with your own personal info:



# Dane Go

### Instructor & Freelance Web Developer School of Information and Computing Science

MS Computer Science, AMA University

---

### About Me

**Aspiring Web Developer, Designer and College Instructor**

**I.T. Instructor**

I.T. Instructor with 3 Years of Experience

**Development Skills**

- HTML
- CSS
- Bootstrap
- Javascript
- Python
- Php
- .Net

---

**Hobbies**

- Coding
- Watching Netflix
- Balisong Flipping
- Gaming
- Mobile Photography
- Sleeping
- Net Surfing

---



## Further Readings

You can check out the links below for references on HTML Usage

MDN: https://developer.mozilla.org/en-US/docs/Web/HTML

W3schools: https://www.w3schools.com/

DevDocs: https://devdocs.io/

### End of Module 1