Intro to HTML & Git

HTML Intro

What is HTML?

- **H**yper**T**ext **M**arkup **L**anguage
- HTML is used to structure and display the contents of a web page.
- HTML is a *markup* language, not a programming language.

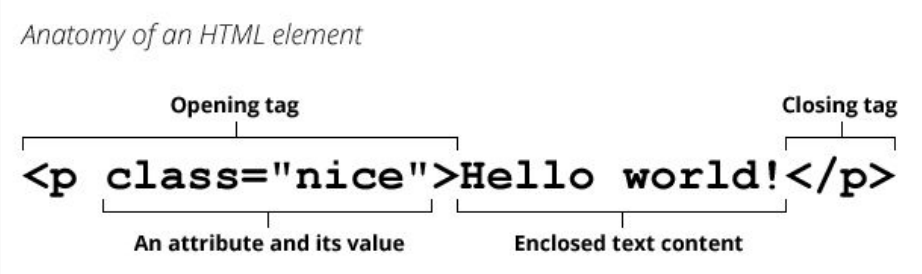
HTML Elements

- HTML is made up of different *elements* that can contain data, text, links, images
- Elements wrap content or text to make it appear and act a certain way.

```
<p>Hello World!</p>
```

HTML Elements

- A typical element can contain different parts or data that describe its characteristics to the browser.
  - An opening tag
  - A closing tag *
  - Attributes *
  - Enclosed content *



* optional

HTML Elements: Tags

- A tag is used to create an HTML element.
- The tag name is enclosed in angle brackets, and specifies what kind of HTML element it will be.
- Elements will have an *opening* tag, and often (not always) a closing tag.
- Closing tags start with a forward slash (/) to indicate the end of the HTML element.
- Tags are often used to wrap around text and other content.

```
<p>Hello World!</p>
```

HTML Elements: Attributes

- Elements can also have tag attributes that contain additional information about the element.
- The attribute information is meant to be *descriptive* -- it is not visible on the web page.
- An attribute contains a name and a value.
- Elements can have more than one attribute.

```
<p class="greeting">Hello World!</p>
```

HTML Elements: Empty tags

- Not all tags must enclose text or other content. Theses are known as *empty* tags.

```
<img src="wave.gif" class="wave-img" />
```

- For an empty tag, you can choose to include the ending forward slash (/) before the closing angle bracket, but it is not required.

```
<br />  OR  <br>
```

HTML Elements: Nesting Elements

You can put HTML elements *inside* other elements…

```
<p><em>You</em> get a car, and <em>you</em> get a car</p>
```

… Just make sure you are closing elements before you end another

```
<p><em>This is incorrect because there is no closing ‘em’ tag</p>
```

HTML Comments

- Comments in HTML are used for the developer to add text to the HTML file that does not get read by the browser.
- Comments in code are useful for leaving notes, explaining bits of code, or providing documentation.

```
<!-- This is a comment in HTML -->
```

Questions?

HTML Intro

HTML Elements

Element Tags

Tag Attributes

Empty Tags

Nesting Elements

The HTML Document

- An HTML file is saved with the *.html* extension.
- Every HTML file requires certain elements to be interpreted by the browser correctly.
- All of these individual elements combined form the entire HTML page.
- Let's break down each component individually...

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <p>Hello, World!</p>
  </body>
</html>
```

DOCTYPE

- The first part of the HTML document is the *doctype*.
- Tells the browser that this is an HTML file and it should handle it as such.
- Required at the beginning of any HTML file.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <p>Hello, World!</p>
  </body>
</html>
```

html Element

- The *html* element is the next part of the page.
- Wraps around the entire HTML page.
- Root of the HTML file.
- Any code within the *html* element should be nested.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <p>Hello, World!</p>
  </body>
</html>
```

head Element

- Container for any information, files, data, scripts, etc. that shouldn't be included in the page *content*.
- *meta* elements provide data to the browser about the document.
- The *title* element is the title of the web page -- often used for browser tabs, search engine results, etc.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <p>Hello, World!</p>
  </body>
</html>
```

body Element

- The beginning and the end of the main content of your HTML page.
- Should contain *all* of the content of your file:
  - Text
  - Images
  - Videos
  - URL Links

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <p>Hello, World!</p>
  </body>
</html>
```

Questions?

Anatomy of an HTML document

Let’s Code.

Other HTML Elements

- There are many HTML elements available to use, each carrying their own meaning.
- We use these elements to create the links, images, lists, text, etc. of a web page.

Headings

- Heading elements allow you to specify that parts of your content are headings
- Can be used to specify titles, subtitles, etc.
- There are six heading levels available, in order of importance `<h1>` - `<h6>`

```
<h1>My main title</h1>

<h2>My top level heading</h2>

<h3>My subheading</h3>

<h4>My sub-subheading</h4>

<h5>My sub-sub-subheading</h5>

<h6>My sub-sub-sub-subheading</h6>
```

Paragraph Elements

- Paragraph elements are used for regular text.
- Sentences, paragraphs of text, or any other context when plain text without any special meaning is included.

```
<p>This is just a regular sentence. Nothing special about it.</p>
```

Images

The *img* element embeds an image into the web page.

```
<img src="wave.gif" />
```

The *src* attribute of the image element specifies the location of the image. This can be a filename or path found locally in your website directory (above)...

```
<img src="https://media.giphy.com/wave.gif" />
```

... Or it can be a full URL where an image file is saved.

Images

All images should include an *alt* attribute

```
<img src="wave.gif" alt="Someone waving hello" />
```

- *alt* attributes are the alternative text for the image. This text should be descriptive of the image it pertains to.
- The *alt* text is often displayed as the pop up text when hovering over an image on the page, and will also be displayed if the image file is missing.
- *alt* text is also read by screen readers and other software that assists users with visual impairments or other disabilities.

Links

Links are specified by the *anchor* element

```
<a href="www.facebook.com">Facebook</a>
```

- The *href* (***h***ypertext ***ref***erence) attribute is the URL that the link will take you to.
- *href* can either be a full website URL, or a path/file to another page in your website directory.

```
<a href="about.html">My About Page</a>
```

- Anchor tags must wrap around some text or other content to create a clickable/intractable link.

Lists

- Lists are used for content that can be grouped into bullet points, a numbered group, etc.
- There are two basic types of lists available in HTML
  - Unordered list `<ul>`
  - Ordered list `<ol>`
- Each item in a list is wrapped in a `<li>` (list item).

```
<ul>
  <li>Unordered Item 1</li>
  <li>Unordered Item 2</li>
  <li>Unordered Item 3</li>
</ol>
<ol>
  <li>Ordered Item 1</li>
  <li>Ordered Item 2</li>
  <li>Ordered Item 3</li>
</ol>
```

Unordered List

- Unordered lists are generally used for bullet points or list items where the order does not matter

```
<ul>

  <li>Unordered Item 1</li>

  <li>Unordered Item 2</li>

  <li>Unordered Item 3</li>

</ol>
```

Ends up looking like this...

- Unordered Item 1
- Unordered Item 2
- Unordered Item 3

Ordered List

- Ordered lists should be used for list items that need to have a specified order to them.
- Preceded by the item’s number

```
<ol>

  <li>Ordered Item 1</li>

  <li>Ordered Item 2</li>

  <li>Ordered Item 3</li>

</ol>
```

Ends up looking like this...

1. Ordered Item 1
2. Ordered Item 2
3. Ordered Item 3

Questions?

Heading elements

Paragraph elements

Images

Links

Lists

HTML Validation

In order to ensure that our HTML documents are valid and error free, we can use the W3C Markup Validation Service website. This service checks for:

- Properly formatted elements.
- Missing element attributes.
- Semantically correct code.

You should validate every HTML document you create. This makes sure that our web pages will be understood by the browser as expected and minimize bugs.

Git & GitHub

What is Git?

- Git is a version control system.
- A version control system keeps track of changes to your files so that you can refer or go back to older versions at a later time.
- Version control systems can track all kinds of files, but for the purposes of this class we will use it for our assignment/project files and source code.

Creating a Git project

- Git projects are known as *repositories*. Repositories house and track all of the files and directories in it.
- To create a git repository using the terminal/command line, type in the following command from your project directory

```
$ git init
```

Tracking files and committing changes

To start tracking the files in your git repository, run:

```
$ git add --all
```

This tells git to track all the files in your project.

Committing your changes

To commit the changes you've made, you run the command:

```
$ git commit
```

This indicates that you are happy with the state of your changes, and git should commit everything up to this point to it's memory.

Running this command will prompt you for a commit message...

Commit messages

- When you commit a change to your git repository using the *commit* command, you will be prompted for a *commit message*.
- Commit messages should describe the changes you're making in the commit. Make sure they are clear, concise, and descriptive enough in case you need to look back at your changes at a later date.
- You can also specify the commit message inline with your commit command by using the *-m* flag:

```
$ git commit -m ‘This is my commit message’
```

Making more changes...

Once you commit your changes, you'll have a clean slate in your git repository and nothing available to track.

Once you make additional changes to your files, you can check the progress/status with the *status* command

```
$ git status
```

This will display the files you have changed since your last commit.

Adding and committing additional changes...

To add files you want to include in your next commit, use the *add* command, followed by the name of the file(s)/directory.

```
$ git add <filename>
```

If you want to add all of your changes to your next commit:

```
$ git add --all  OR  $ git add .
```

After, run the *commit* command to commit the files you've added.

Syncing your local repo with a hosted repo...

To add a link to a hosted online repo, like a GitHub repo, you can use the following command

```
$ git remote add <remote_name> <remote_url>
```

A *remote* refers to a remote repository - one that is hosted online. The `remote_name` is a unique identifier you specify. The `remote_url` is the URL where your online repository can be found.

Pulling changes from a remote repo

To update your local repository with any changes made to your remote repo, run the command

```
$ git pull <remote_name> master
```

The <remote_name> is the name you have given your remote repo. This will keep your local repository in sync with your remote repository.

Pushing changes to your remote repo

To update your remote repository with changes made to your local…

```
$ git push <remote_name> master
```

This keeps your remote repo in sync when changes are made to your local repository.

Questions?

Git

Creating a git repository

Tracking changes

Committing changes and commit messages

Adding files to commit

GitHub

- GitHub is a git repository hosting service.
- Gives users a space online to keep all of their git repositories, making them publicly or privately available.
- Offers several additional features to assist private hosting, collaboration, development workflows, etc.

To use GitHub and the Class Repo

- Create a GitHub account
- Request access to the GitHub Student Developer Pack. This gives you access to a ton of free services, such as being able to create private GitHub repositories.
- Once you have a GitHub account, complete the class form posted to the slack channel to share your GitHub username with me. I will then add you as a collaborator to the class repo.
- After you complete these tasks, you'll be able to follow the assignment prompt for further instructions on submitting assignments.

Our Class GitHub Repository

Our class repository can be found here. It is a private repository, so you will all need to create a GitHub account and request access to it.

The class repo is where I will be posting all assignment prompts, keeping in-class code, and where you will be submitting your completed assignments.

Helpful Resources

- Mozilla Developer Network: Getting started with the Web