Unit One
IDE’s, GitHub, File Structures & Intro to HTML

AIM Code School
Vanessa Kasun

IDE

What is an IDE?

- IDE stands for integrated development environment
- Software application for computer programmers to use for development
- Normally consists of at least a source code editor, build automation tools and a debugger

Some IDE's that are out there for developers to use are...

- Visual Studio
- Intellij IDEA
- Aptana Studio 3
- PyCharm
- PhpStorm
- WebStorm
- NetBeans
- Eclipse

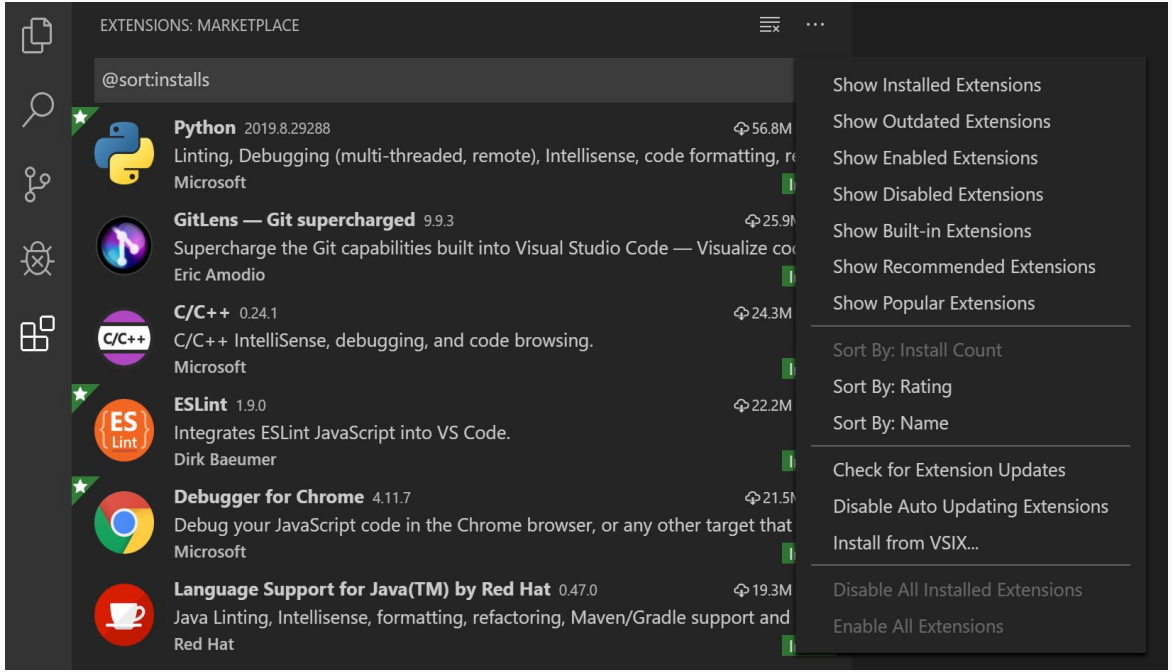
Downloading VS Code

Download the open source files for your corresponding operating system.

Once it's downloaded run the .exe file to install the software on your local machine.

Open VS code and install the following extensions:

- HTML5
- CSS3
- JavaScript
- jQuery



GitHub Version Control

What Exactly is GitHub?

It’s a repository hosting service. It provides a Web-based GUI (graphical user interface) for software development and version control.

It also provides access control and several collaboration features like task management for every project.

Git is a command line tool for those who like use CLI (command line interface)

Creating An Account

Go to the GitHub website and sign up for a free account

Create a Username and Password

Once you are finished, send me your Username so I can add you to your classes repository.

My email: vkasun@aimcodeschool.org or post in chat

Introduction to HTML

Let’s get started!

HTML

- HTML stands for **Hypertext Markup Language**
- Standard language for documents designed to be displayed in a web browser
- HTML documents can also consist of other technologies such as CSS and JavaScript

HTML is often referred to as the bones of your webpage.

HTML is made up of different **elements** and each element is defined by a starting tag, some content, and an end tag.

Example: *OR*

<tagname>Content Goes Here</tagname>

Each tag has its own purpose and all together they determine **ONLY** the basic structure of a webpage.

An HTML Web Page

All web pages have a minimum set of required tags that need to be on the HTML in order for the web page to display and function correctly.

HTML - wraps the entire web page, but its not displayed. Referred to as the root of the page.

HEAD - Contains meta (i.e extra) info for the web page, but not displayed

TITLE - sets the title of the page. This shows up in the bowers

BODY - the visible content of the page

```
<!DOCTYPE html>
<head>
        <title>Page Title Goes Here</title>
        <meta charset="UTF-8">
</head>
<body>
        Page Content Goes Here
</body>
</html>
```

HTML

- Tagname determines the function of the tag (how it will be displayed on web page)
- Some tags has a set of possible **attributes** that help determine what the tag will display.
- Attributes have a **value** (sometimes multiple values) that provide detail about what the attribute actually does.

Examples:

```
<a href="index.html>Home</a>

<img src="../images/photo.jpg" alt="img" />

<article id="home_page">

Home page Content

</article>
```

ID’s and Classes

Id and class are two **attributes** that are considered to be **global.**

They are used to mark their tags.

(Think of this as a bookmark used to reference a certain page in the book once you pick up the book to start reading again)

You’ll see how these attributes are used in css by styling the html

Classes:

- Class selectors allow us to select an element based on the element’s class attribute value. Class selectors are a little more specific than type selectors, as they select a particular group of elements rather than all elements of one type.
- Class selectors allow us to apply the same styles to different elements at once by using the same class attribute value across multiple elements.

.classname

Ids:

- ID selectors are even more precise than class selectors, as they target only one unique element at a time. Just as class selectors use an element’s class attribute value as the selector, ID selectors use an element’s id attribute value as a selector.
- Regardless of which type of element they appear on, id attribute values can only be used once per page. If used they should be reserved for significant elements.

#idname

★ For now, just remember that the attributes id and class can be used on any HTML tag

Inline & Block

The <nav> tag is an HTML5 semantic tag whose purpose is to contain your sites main navigation links, the <a> tags.
Semantic tags are just tags whose name implies its purpose.

Block level elements: always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

Inline level elements: does not start on a new line and it only takes up as much width as necessary.

Here are the block-level elements in HTML:

<address> <article> <aside> <blockquote> <canvas> <dd> <dl> <dt> <fieldset> <figcaption> <figure> <footer> <form> <h1> - <h6> <header> <hr> <main> <nav> <noscript> <p> <pre> <section> <table> <tfoot> <video>

Here are the inline elements in HTML:

**<a> <abbr> <acronym> <b> <bdo> <big>
 <button> <cite> <code> <dfn> <i> <img> <input> <kbd> <label> <map> <object> <output> <q> <samp> <script> <select> <small>   <sub> <sup> <textarea> <time> <tt> <var>**

Parent/Child Tags

Tags that contain other tags inside of them are called **PARENT** tags.

Tags that are DIRECTLY contained inside of those parent tags are called **CHILD** tags.

In this example the <html> tag is the parent tag of both the <head> and <body> tag. There are two child tags in this code between the <html> parent tag. These tags are the <head> and the <body> tag.

So… <HTML> tag is the parent tag and the

<HEAD> & <BODY> tag are the child tags

```
<html>
 <head>
  <title>My Page Title</title>
 </head>
 <body>

  This is where all my web page content goes!

 </body>
</html>
```

Ancestor/ Descendant tags

<head> is also the parent of the <title> tag which would make <title> the child tag. The relationship between the <html> tag & the <title> tag is more indirect but is still present. The <html> tag would be considered the **ancestor** of the <title> tag.

Tags that are DIRECTLY contained inside of those parent tags are called **CHILD** tags.

Likewise the <title> tag would be considered the **descendant** of the <html> tag

Keep these terminologies in mind. They are very important once we get into CSS.

```
<html>
 <head>
  <title>My Page Title</title>
 </head>
 <body>

  This is where all my web page content goes!

 </body>
</html>
```

Self Closing Tags

In the previous example the <meta> tag had only one tag and did not include a closing tag. That is because not all elements consist of opening and closing tags. Some simply receive t
their content or behavior from attributes within a single tag. The <meta> tag is one of these elements.

A few other common self closing elements include..

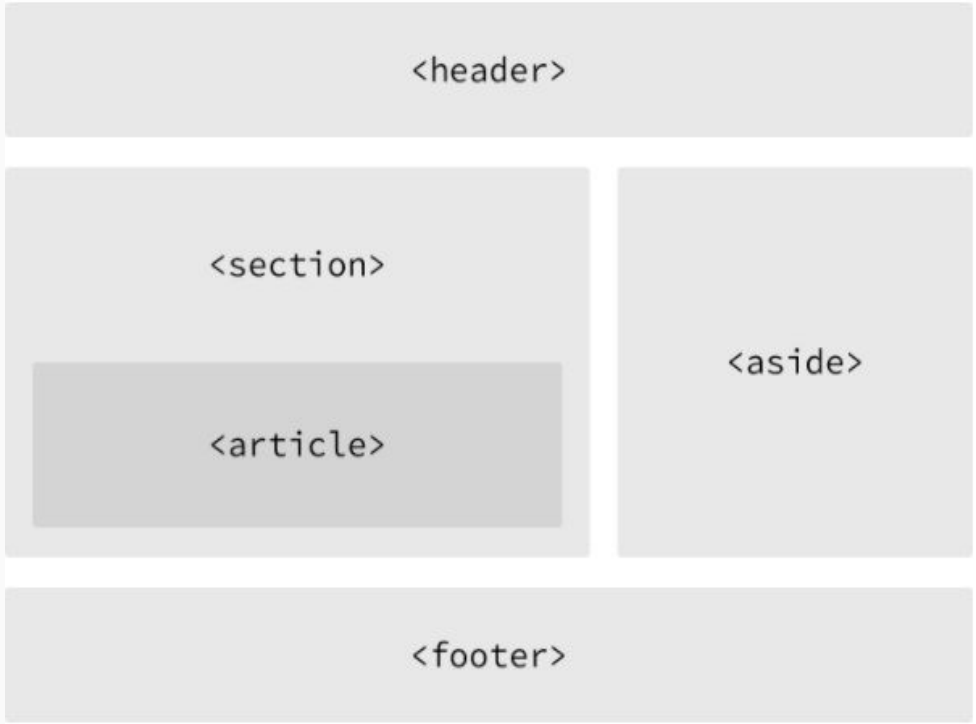
-
- <embed>
- <hr>
- <img>
- <input>
- <link>
- <meta>
- <param>
- <source>
- <wbr>

Building Structure

Structurally Based Elements

<header>
<nav>
<article>
<section>
<aside>
<footer>

All of these new elements are intended to give meaning to the organization of our pages and improve our structural semantics. They are all block-level elements and do not have any implied position or style. Additionally, all of these elements may be used multiple times per page, so long as each use reflects the proper semantic meaning.



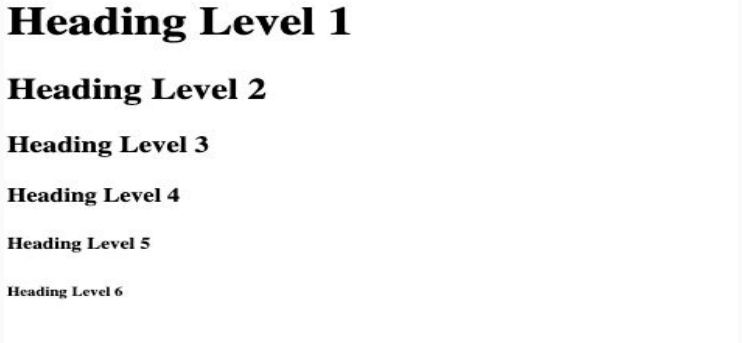
Headings

Heading tags are block level-elements, and they come in six different rankings <h1> - <h6>

Headings should be used in an order that is relevant to the content of a page. The primary heading of a page or section should mark up with an <h1> elements, and subsequent headings should use and from <h2> to <h6> elements as necessary.

HTML

```
<h1>Heading Level 1</h1>
<h2>Heading Level 2</h2>
<h3>Heading Level 3</h3>
<h4>Heading Level 4</h4>
<h5>Heading Level 5</h5>
<h6>Heading Level 6</h6>
```



Paragraphs

Headings are often followed by supporting paragraphs. Paragraphs are defined using the <p> block-level element. Paragraphs can appear one after the other, adding information to a page as desired. Here is an example of how to set up paragraphs

Bold Text with Strong & Italicize with Emphasis

To make text bold and place a strong importance on it, we'll use the inline-level element. There are two elements that will bold text for us: the and <b> elements.

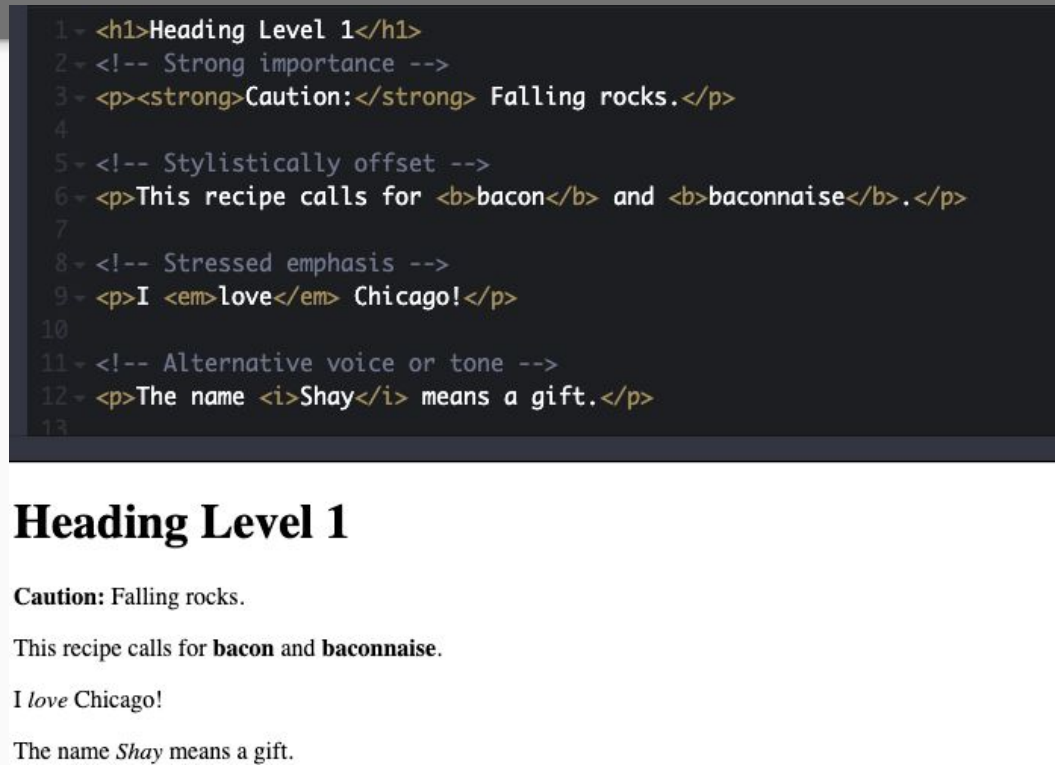
The element is used semantically to place a stressed emphasis on text; it is thus the most popular option for italicizing text. The other option, the <i> element, is used semantically to convey text in an alternative voice or tone, almost as if it were placed in quotation marks. Again, we will need to gauge the significance of the text we want to italicize and choose an element accordingly.

```
<h1>Heading Level 1</h1>
<!-- Strong importance -->
<p><strong>Caution:</strong> Falling rocks.</p>

<!-- Stylistically offset -->
<p>This recipe calls for <b>bacon</b> and <b>baconnaise</b>.</p>

<!-- Stressed emphasis -->
<p>I <em>love</em> Chicago!</p>

<!-- Alternative voice or tone -->
<p>The name <i>Shay</i> means a gift.</p>
```



Let’s Practice!

1. In File Explorer (windows) or Finder (mac) create a new folder named “my_first_html”
2. Open up VS code
3. Click Menu, New File
4. Save File as “index.html” (make sure you save it inside of the folder you just created)
5. Inside the index file, add the following…

Let’s Practice!

1. Inside the <head> tag
2. Add <meta> and <title> elements.
3. The <meta> element should include the proper charset attribute and value, while the <title> element should contain the title of the page -- we’ll say “My First Web Page”

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>My First Web Page</title>
    </head>
    <body>
    </body>
</html>
```

Let’s Practice!

1. Inside the <body> element, let’s add <h1> and <p> elements.
2. The <h1> element should include the heading we wish to include -- let’s say “My First Web Page” again.
3. And in the <p> tag let’s say -- “Hello World! My Name is “_____”, and this is my first web page.”

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>My First Web Page</title>
    </head>
    <body>
        <h1>My First Web Page</h1>
        <p>Hello World! My name is Sally, and this is my first web page.</p>

    </body>
</html>
```

Hyperlinks

- Along with text, one of the core components of the Internet is the hyperlink, which provides the ability to link from one web page or resource to another. Hyperlinks are established using the anchor, <a>, inline-level element. In order to create a link from one page (or resource) to another, the href attribute, known as a hyperlink reference, is required. The href attribute value identifies the destination of the link.

Linking To an Email Address: To create an email link, the href attribute value needs to start with mailto: followed by the email address to which the email should be sent.

```
<a href="mailto:vkasun@aimcodeschool.org">Email</a>
```

Hyperlinks

- Additionally, subject, body text, and other information for the email may be populated. To add a subject line, we'll include the subject= parameter after the email address. The first parameter following the email address must begin with a question mark, ?, to bind it to the hyperlink path. Multiple words within a subject line require that spaces be encoded using %20.

- Adding body text works in the same way as adding the subject, this time using the body= parameter in the href attribute value. Because we are binding one parameter to another we need to use the ampersand, &, to separate the two. As with the subject, spaces must be encoded using %20, and line breaks must be encoded using %0A.

- Altogether, a link to vkasun@aimcodeschool.org with the subject of "Reaching Out" and body text of "How are you" would require an href attribute value of mailto:vkasun@aimcodeschool.org?subject=Reaching%20Out&body=How%20are%20you.

```
<a href="mailto:vkasun@aimcodeschool.org?subject=Reaching%20Out&body=How%20are%20you">Email Me</a>
```

Links in a new window & Linking parts of the same page

New Window Link

To trigger the action of opening a link in a new window, use the **target** attribute with a value of **_blank.** The target attribute determines exactly where the link will be displayed, and the **_blank** value specifies a new window.

```
<a href="http://google.com/" target="_blank">Google</a>
```

Parts of the Same Page

We can create an on-page link by first setting an **id** attribute on the element we wish to link to, then using the value of that **id** attribute within an anchor element's href attribute.

```
<body id="top">.

<a href="#top">Back to Top</a>

</body>
```

This is the sample website we will be creating for the basic html and css concepts.

Navigations and Links

Let's get started putting our Home page together.

BUT first we need to create a folder for this project and the necessary subfolders that we will use for our **images** and **css.** Name this folder **Style_Conference.** The subfolder needs to be named **assets.** Inside assets create two more folder named **css** & **images**
(*You should have a folder created specifically for this class. If you don't please create one and name it **aim_foundations).** Every project/assignment/ in-class work that we do in this course should be saved in this folder.

Now we need to open VS Code and create a new file with the required HTML tags. Once you have the required tags on the file **save** the file first as **boilerplate.html. This needs to be saved directly inside of your boilerplate folder which is inside the Foundations-Of-Web-Development (aka course) folder.**

Once you have saved it to your **Foundations-Of-Web-Development** folder as boilerplate.html...

- Click on file
- Select Save As
- Rename the file **index.html**
- Save to the **Style_Conference**
- Click Save

How to create a class folder for Foundations and project folders

Please watch this video as many times as you need. Pause if you need too, but follow the directions step by step to create the course folder, the boilerplates folder, and your first code challenge folder.

End of Lesson

Next we will dive deeper into HTML and get introduced to CSS!

Complete Code Challenge One & once you are finished...
Please Work on Assignment #1