**Week 01**

**W01 Setup Activities**

1. Editing Code: [Visual Studio Code](https://byui-cse.github.io/wdd130-ww-course/week01/setup-visual-studio-code.html)
2. Testing: [Web Developer Browser Extension](https://byui-cse.github.io/wdd130-ww-course/week01/setup-webdev-extension.html)
3. Hosting Your Work: [GitHub Account](https://byui-cse.github.io/wdd130-ww-course/week01/setup-github.html)
4. Uploading: [Installing Git](https://byui-cse.github.io/wdd130-ww-course/week01/setup-git.html)
5. Collaboration: [Microsoft Teams](https://byui-cse.github.io/wdd130-ww-course/week01/setup-msteams.html)

**W01 Learning Activities**

1. [File and Folder Naming Conventions](https://byui-cse.github.io/wdd130-ww-course/week01/prepare-naming-conventions.html)
2. [An Introduction to HTML](https://byui-cse.github.io/wdd130-ww-course/week01/prepare-html-introduction.html)
3. [Working with Web Images](https://byui-cse.github.io/wdd130-ww-course/week01/prepare-images.html)

**W01 Assignment: Student Home Page**

1. Course Home Page: [Building your Course Home](https://byui-cse.github.io/wdd130-ww-course/week01/prove-html-start.html)
2. Publishing: [Enabling GitHub Pages](https://byui-cse.github.io/wdd130-ww-course/week01/setup-github-pages.html) | ✔️ [Page Evaluation Tool](https://byui-cse.github.io/wdd130-ww-course/grader/w01-homepage.html)

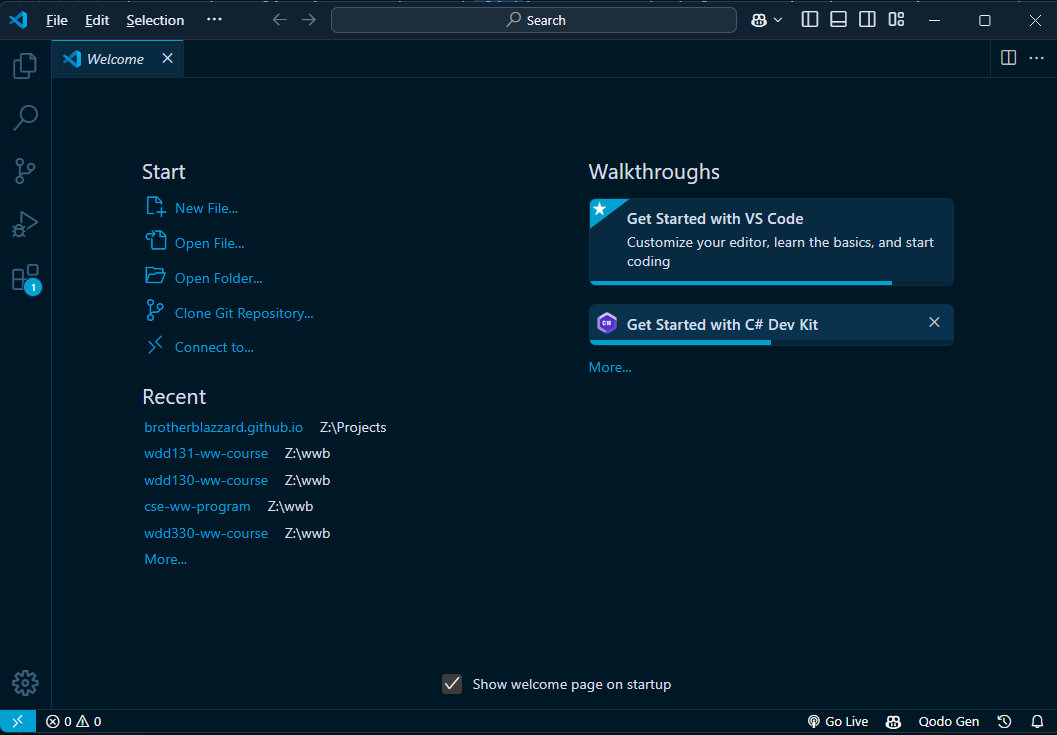
### Instructions

**If** you already have VS Code installed:

* Make sure it is up to date by going to **Help** on the VS Code application menu bar and clicking **Check for Updates...**.  
  Note that VS Code will [automatically try to update itself](https://code.visualstudio.com/docs/setup/setup-overview#_update-cadence) each month.
* Skip to **Step 2** in the Instructions section below.



#### Download and Install VS Code

1. In your browser, navigate to [Visual Studio Code](https://code.visualstudio.com/).
2. You will be prompted to download VS Code for your operating system.  
   Click **download** and follow the directions to download and install VS Code.  
   There are no required settings to change during the installation process.
3. Verify that VS Code is installed and can be run on your computer.Visual Studio Code Opening Welcome Screen

#### Set VS Code to Automatically Save Files

1. Start VS Code, and in the VS Code menu bar, click  
   **File ▶️ Preferences ▶️ Settings** [Windows] or  
   **Code ▶️ Settings... ▶️ Settings** [Mac]

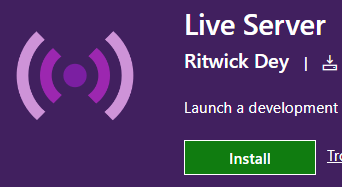
You can always get into the **Settings** panel by pressing the two key shortcut **⌘,** [Mac] or **Ctrl,** [Windows/Linux].

1. Type **auto save** in the search box.
2. In the **Files: Auto Save** panel, change the dropdown menu from **off** to **afterDelay**.

Option: You can set the delay to your preferred time in milliseconds in the **Files: Auto Save Delay** setting.

#### Install the Live Server Extension

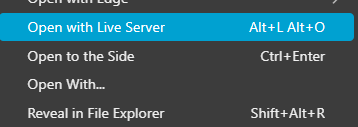
One of the reasons that VS Code is popular and useful is that custom extensions can be added to allow developers to customize the application. In this class we will save time by using the **Live Server** extension to display web pages locally, in the browser, for testing purposes before publishing.

1. In your browser, navigate to the [Live Server](https://marketplace.visualstudio.com/items?itemName=ritwickdey.LiveServer) extension in Extension Marketplace.
2. Select **Install** and follow the prompts carefully to install the extension within VS Code.Extension Install Button

[Manage extensions](https://code.visualstudio.com/docs/editor/extension-marketplace) in VS Code using the Extensions View. The Extensions View icon is found in the Activity Bar.

Extension View Icon in the VS Code Activity Bar

You may need to close and restart VS Code in order to start using the extension.

1. Any file with an **.html** extension can be opened using Live Server from within VS Code. The user's default browser will launch and render the page as if on a web server.After right mouse clicking on an HTML file in the VS Code explorer, this menu option appears.

This tool allows pages to be tested locally without going through the publication process to the remote web server.

[**Next: Web Developer Extension**](https://byui-cse.github.io/wdd130-ww-course/week01/setup-webdev-extension.html)

## Setup: Browser Web Developer Extension

### Overview

To aid our development cycle and reviews, we will use a browser extension named **Web Developer**. This extension provides access to some endorsed validation and page checking tools including HTML markup validation, CSS code validation, and page accessibility testing.

"The Web Developer extension adds various web developer tools to a browser. The extension is available for Chromium based browsers, Firefox, and Opera, and will run on any platform that these browsers support including Windows, macOS, and Linux." - chrispederick.com

### Prepare

#### Extensions

Browser extensions are small, dependent, software modules that allow users to customize and enhance their browsing experience. Browser extensions are managed within the browser settings. Access to the settings vary but are similar between browsers using the upper right menu access button ( ••• ≡ ) in the browser window.

You can also access your extensions by using the address bar; for example, in Google Chrome, you can manage extensions by navigating to **chrome://extensions**.

For a comprehensive overview of browser extensions, visit the Wikipedia's Browser extension article in the optional resources provided below.

#### Document Validation

We will use the **Web Developer** extension to validate our HTML markup and CSS. Validation means that we check our document to see if it follows the programming language's rules. This is similar to checking the spelling and grammar in a document of a spoken language.

Tools like the Web Developer extensions help developers find issues that may cause the page to render in unexpected ways. Browsers also have built-in tools like DevTools, that will be used later in this course and in subsequent courses.

### Instructions

In this activity, you will install the extension on your browser. The extension will be used throughout this course.

1. Install the [Web Developer](https://chrome.google.com/webstore/detail/web-developer/bfbameneiokkgbdmiekhjnmfkcnldhhm?hl=en-US) browser extension.

You can follow the instructions in the link above to install the extension, or you can add the extension to your preferred browser. This [Web Developer](https://chrispederick.com/work/web-developer/) resource provides information on downloading and installing on other browsers.

1. Follow the directions and install the browser extension.
2. Once installed, you should see the extension icon on your browser toolbar.

Video Demonstration: ▶️ [Using the Web Developer Extension](https://www.loom.com/share/011b610fa093414e9745398365177acf) [5:00 minutes]

#### Optional Resources

* [W3C HTML Validation](https://validator.w3.org/nu/) - w3.org (Worldwide Web Consortium)
* [W3C CSS validation](https://jigsaw.w3.org/css-validator/) - w3.org - (Worldwide Web Consortium)
* [WAVE Accessibility](https://wave.webaim.org/) - webaim.org - (Institute for Disability Research, Policy & Practice Utah State University)

[**Next: GitHub Account**](https://byui-cse.github.io/wdd130-ww-course/week01/setup-github.html)

## Setup: GitHub Account

### Overview

The purpose of this setup activity is to help you obtain a free **GitHub** account. GitHub is a popular and free online collaboration and version control service. You can think of GitHub repositories as a cloud-based storage system for your projects. GitHub can be used as a web hosting service with a special web service named **GitHub Pages**.

If you already have a GitHub account, you do **not** need to create a new account with GitHub for this course. **You can skip this module and move on.**

### Instructions

#### Sign Up for a free GitHub Account

1. In your browser, visit the [github.com](https://github.com/) site.
2. Select the **Sign Up** button.
3. Enter an email address and select **Continue**.
4. Create a **password** that has at least 15 characters or 8 characters with a number and lowercase letter. Then select **Continue**.
5. Enter a **username** using only lowercase letters. Your username will show up in the domain of your web projects. If you get a red '❌' to the left of your username, then someone else has already used that username and you need to choose a different one.

Choose a professional name given that you may use your GitHub account as a portfolio of your work.

1. Record your username and password in a secure location.
2. Enter 'n' for the next step unless you want email from GitHub.
3. At this point, GitHub will ask you solve a few simple puzzles to ensure that you are a human and not a bot. Click **Start** puzzle and follow the directions. Then click **Create Account**.
4. GitHub will send a launch code to the email address you provided. Follow the instructions in your email to use that launch code.
5. You can skip any personalization questions by selecting the Skip Personalization link near the bottom.

You do **not** need to create any repositories at this time. Directions about the course repository will be provided in later modules.

Video Demonstration: ▶️ [Setting Up a GitHub Account](https://youtu.be/eMbF5kYqPt0)

[**Next: Installing Git**](https://byui-cse.github.io/wdd130-ww-course/week01/setup-git.html)

## Setup: Git Installation

### Overview

Git is a version control software framework that will run behind the scenes, tracking additions, updates, and deletions to your project files and folders. You will use **git** to commit files in your **wdd131** project and push/upload your work to your GitHub, remote repository.

### Instructions

#### Install Git on your computer

Select only **one** of the following instruction sets depending on your operating system.

**Windows Users**

1. Before downloading **Git**, check to make sure you don't already have Git installed on your computer.
   1. Open a **command prompt** by clicking the Start button and typing cmd in the search box. The command prompt terminal will open.

Alternatively, in VS Code, you can use the terminal panel to complete any command line task. use the **Ctrl+`** keyboard shortcut to toggle the terminal panel.

* 1. Enter the following command and then press **Enter**:

git --version

* 1. If you see a version number, you already have Git installed. If you don't see a version number, you need to install Git.

1. To install Git for Windows. Go to [git-scm.com/downloads](https://git-scm.com/downloads) and select **Download for Windows**.  
   An .exe file will be downloaded.
2. Double click that .exe file to open and run the installation executable. The process of installing Git will start.
   1. Allow the installer to make changes to your computer.
   2. Select **Next** through all of the setup windows, leaving all the defaults. There will be many windows.
   3. The last window will let you select **Install** and then click **Finish**.
3. Open a **new** Command Prompt window by closing the first Command Prompt window and start a instance by clicking the Start button and typing cmd in the search box.
4. Enter the git --version command and then press **Enter** to see if Git is installed. You should see the version number.
5. While in the Command Prompt, type in two more commands to set up our username and email that are associated with our GitHub account. Again, since these are global settings so you can type them from any path prompt.

Make sure you to use your own username and email between the "" quotation marks. Use the username and email you used for the GitHub account. These will be different for each student.

* 1. Type in this configuration command to set the global git username:

git config --global user.name "yourusername"

and select **Enter**. Nothing will happen if you did it right. If you get an error, you need to fix it.

* 1. Enter this configuration command to set the email:

git config --global user.email "youremail@byui.edu"

**Nothing** will happen if you did it right. If you get an error, you need to fix it.

* 1. Type in this configuration command in order to not ignore case changes in files and folder names:

git config --global core.ignorecase false

**Nothing** will happen if you did it right. If you get an error, you need to fix it.

Video Demonstration: ▶️ [Windows Git Installation](https://youtu.be/3KpzB3ZZCZ4)

**macOS Users**

1. Before downloading **Git**, check to make sure you don't already have Git installed on your computer.
   1. Select the **Search** icon (it looks like an magnifying glass) on your screen near the top right of your screen.
   2. Type in **"Terminal"** and open the Terminal application. It doesn't matter what path our command prompt is showing. All the commands we type will work anywhere.
   3. Type in git --version at the prompt and select **Enter**.
   4. You should now either see a git version number or a pop up window that says something along the lines of **'The git command requires the command line developer tools ...'**. If you see the git version number, you already have git installed and do not need to install git, but you will want to set your config setting as shown below. If you get the pop-up message, select **Install and Agree**.
   5. The installation may take a few minutes. With the installation, you will get a set of development tools, including **Git**.
   6. Type in git --version at the prompt and select **Enter**. You should now see the version number.
   7. While in the command prompt, type in additional commands to configure our username and email that are associated with our GitHub account. Again, since these are global settings you can type them from any path prompt.

Make sure you use your own username and email between the "" quotation marks. Use the username and email you used to create your GitHub account. These will be different for each student.

* + 1. Type in this configuration command to set the global git username:

git config --global user.name "yourusername"

and select **Enter**. Nothing will happen if you did it right. If you get an error, you need to fix it.

* + 1. Enter this configuration command to set the email:

git config --global user.email "youremail@"

**Nothing** will happen if you did it right. If you get an error, you need to fix it.

* + 1. Type in this configuration command in order to not ignore case changes in files and folder names:

git config --global core.ignorecase false

**Nothing** will happen if you did it right. If you get an error, you need to fix it.

* 1. If you want to check the global configurations, type

git config *--list*

to review the user.name and user.email. Do not worry about the other configuration settings reported.

Video Demonstration: ▶️ [Git Install for Mac](https://youtu.be/kJLYtoj4VpA)

**Linux Users**

1. Guide: [Linux Git Install](https://github.com/git-guides/install-git#install-git-on-linux)
2. Install Git using the link above and then follow the last few steps from the Windows or Mac instructions to set up your global config username and email settings to match your GitHub account.

[**Next: Microsoft Teams**](https://byui-cse.github.io/wdd130-ww-course/week01/setup-msteams.html)

## Setup: Microsoft Teams

### Overview

Team learning is a collaborative effort and teams are an essential part of most career fields. In this setup activity, you will install and setup Microsoft Teams.

**"that all may be edified of all" -**[**D&C 88:122**](https://www.churchofjesuschrist.org/study/scriptures/dc-testament/dc/88?lang=eng&id=p122#p122)

One of the most powerful ways to learn is to teach. The Microsoft Teams collaboration tool supports the principles of love, service, and teach one another as stated in the [BYU Idaho Learning Model.](https://www.byui.edu/learning-model/teach-one-another)

### Activity Instructions

#### Microsoft Teams

Microsoft Teams is a collaboration platform by Microsoft that allows you to communicate, collaborate, and share files with your peers.

1. Download [Microsoft Teams](https://www.microsoft.com/en-us/microsoft-365/microsoft-teams/download-app) on your computer and other device(s) as needed.
2. Use your BYU-Idaho account to sign into Microsoft Teams.
3. Complete the [Introduction to Microsoft Teams](https://byui-cse.github.io/cse-ww-program/student/ms-teams.html) tutorial.
4. Complete the enrollment invitation you receive via email to enroll into this section's WDD 130 MS Team named something like Web Fundamentals.
5. Turn on **Channel Notifications** for the **General** channel by clicking on the kebab menu ••• next to General channel and selecting **Channel notifications**. A settings dialog box will be provided. Select **Banner and feed** to received notifications and check include replies.
6. Go to the **Introductions** channel and post a brief bio and hello to the rest of the class and turn on notifications.

Course-wide discussions, announcements, and other course topic threads will also take place in Teams in the **General** channel. You will be able to ask and answer questions and receive help from your peers and the instructor in the **General** channel.

### Getting Help

If you were not able to complete any of these steps, then you are not prepared yet.  
Please reach out for help.

## Course File and Folder Naming Conventions

### Overview

When working on the web, there are numerous things that affect the operation of the files, such as the browser, the protocol(s), the operating system, the language(s), etc. While many of these are out of your control, you can keep your work consistent and manageable through standard file and folder naming conventions.

"When you're building a website, you need to assemble these files into a sensible structure on your local computer, make sure they can talk to one another, and get all your content looking right before you eventually upload them to a server." - MDN (Mozilla Developer Network Reference)

### Prepare

* The file and folder naming conventions outlined here will be considered 'best-practice' rules for this course. These rules should be applied to all files and folders created in your course work.

Note that most organizations have their own file management guidelines and best practices, including conventions for naming files, folders, and other workflows. These conventions may vary from one organization to another.

* [File Path Essential Training](https://byui-cse.github.io/wdd130-ww-course/week01/resource-file-paths.html)

### Naming Convention Rules

* Use all **lowercase** syntax.

Platforms and systems handle case sensitivity differently. Case sensitivity is an important concept to understand when managing files and folders.

Example: **products.html**

* Do **NOT** use **spaces** in file and folder names. Instead, use hyphens **-**.

Spaces are handled inconsistently by user agents so do not use them. The Hypertext Transfer Protocol (HTTP) ignores spaces, except in file names. In file names, it replaces a space with this syntax: "**%20**" which syntax is confusing. Avoid using spaces and instead, if you want to create visual space, use hyphens.

Example: **design-document.html**

* Do **NOT** use **special characters**.

Special characters often mean specific things to computers, so do not use them in the naming of files and folders.

Special character examples (avoid these): **<,>, \, /, #, ?, !**

* File and folder names should be as short and meaningful (**semantic**) as possible.

Short names save you, other developers, and site visitors from having to remember long complicated names for files and folders. When meaningful, names can also help predict the purpose or nature of the file or folder contents.

Example: **winter-scene-sm.png** vs. **image13-v123523brokenbranchlifeimagery w200x200.png**

* In this class, the **standard folder names** for sub-folders are:
  + **styles** - Folders with this name contain CSS files.
  + **images** - Folders with this name contain images.

#### Optional Resources

* [Dealing With Files](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/Dealing_with_files) - MDN (Mozilla Developer Network Reference)

[**Back**](https://byui-cse.github.io/wdd130-ww-course/week01/index.html)[**I-Learn**](https://byui.instructure.com/calendar/)

## An Introduction to HTML

### Overview

HTML stands for **HyperText Markup Language** and is one of the three core technologies of the web, HTML, CSS, and JavaScript. HTML defines the **structure** and **meaning** of a web document. Hypertext refers to the way hypertext links are placed in the document that allows users to move from one page to another. Markup is a set of symbols or codes for displaying content on the Internet. Web browsers, like Google Chrome, use HTML markup and content to render pages.

"HTML (HyperText Markup Language) is the most basic building block of the web. It defines the meaning and structure of web content. Other technologies besides HTML are generally used to describe a web page's appearance/presentation (CSS) or functionality/behavior (JavaScript)." - MDN (Mozilla Developer Network Reference)

### Prepare

Prepare for the assignment by carefully pondering, noting, and referencing the materials provided.

* Video: ▶️ [What is HTML?](https://www.youtube.com/embed/jxYtjiuPfLQ) - (3:25 mins, 📄 [Transcript](https://byui-wdd.github.io/wdd130/text/what_is_html.pdf))
* video: ▶️ [HTML Elements and Attributes](https://youtu.be/GpZmiD8WlFo) - (5:52 mins, 📄 [Transcript](https://byui-wdd.github.io/wdd130/text/beg_html_elements.pdf))

### Activity Instructions

In this activity, answer the following questions about HTML and ponder the provided solutions.

1. The following HTML elements are required for every HTML document created in this course.  
   What is the purpose of each of the following HTML elements?
   * <!DOCTYPE html>

Answer

"It is a required preamble. In the mists of time, when HTML was young (around 1991/92), doctypes were meant to act as links to a set of rules that the HTML page had to follow to be considered good HTML, which could mean automatic error checking and other useful things. However, these days, they don't do much and are basically just needed to make sure your document behaves correctly. That's all you need to know for now." - MDN

✔ This is a development checklist item for all HTML documents produced in this course.

* + <html lang="en-US"> ... </html>

Answer

"This element wraps all the content on the entire page and is sometimes known as the root element. It also includes the **lang** attribute, setting the primary language of the document." - MDN

✔ This is a development checklist item and must include the lang attribute.

* + <head> ... </head>

Answer

"The **head** HTML element contains machine-readable information (metadata) about the document, like its character support, viewport sizing, title, scripts, and style sheets." - MDN

✔ This is a development checklist item.

* + <meta charset="utf-8">

Answer

Located within the <head> element. "This attribute declares the document's character encoding. If the attribute is present, its value must be an ASCII case-insensitive match for the string "utf-8", because UTF-8 is the only valid encoding for HTML5 documents. <meta> elements which declare a character encoding must be located entirely within the first 1024 bytes of the document." - MDN

✔ This is a development checklist item.

* + <meta name="viewport" content="width=device-width,initial-scale=1.0">

Answer

Located within the <head> element. "This viewport element ensures the page renders at the width of viewport, preventing mobile browsers from rendering pages wider than the viewport and then shrinking them down." - MDN

✔ This is a development checklist item.

* + <title>...</title>

Answer

Located within the <head> element. "This sets the title of your page, which is the title that appears in the browser tab of the page. It is also used to describe the page when you bookmark/favorite it." - MDN

The content of the title should closely match the content of the heading 1 <h1> element in the document <body>.

✔ This is a development checklist item and The content must be relevant to the document.

* + <body> ... </body>

Answer

"The <body> HTML element represents the content of an HTML document. There can be only one body element in a document. It contains all the content that you want to show to web users when they visit your page, whether that's text, images, videos, games, playable audio tracks, or whatever else." - MDN

✔ This is a development checklist item.

1. What is the basic anatomy of an HTML document?Answer
2. <!DOCTYPE html>
3. <html lang="en-US">
4. <head>
5. <meta charset="utf-8">
6. <meta name="viewport" content="width=device-width">
7. <title>Relevant Document Title</title>
8. </head>
9. <body>
10. </body>

</html>

1. Using the following example HTML snippet of code, technically **name** the numbered parts:

<p>A paragraph containing a hyperlink to <a href="https://www.byui.edu">BYU-Idaho</a>.</p>

* + **<p>**
  + **</p>**
  + **href**
  + A paragraph containing a hyperlink to [BYU-Idaho](https://www.byui.edu/).
  + The entire snippet.

Answers

* + **opening or start tag**
  + **closing or end tag**
  + **attribute** - The value of this **href** attribute, https://www.byui.edu, specifies the anchor <a> tag's destination, which destination in this case is an absolute reference, and means **hypertext reference**.
  + **content** or the text node of the paragraph element.
  + An HTML paragraph **Element**

1. Some elements have no content and no closing tag. They are called \_\_\_\_\_\_\_ or self-closing tags.  
   An example element of this type is an image **<img>** element.Answer

[Void elements](https://developer.mozilla.org/en-US/docs/Glossary/Void_element)  
Example:

<img src="images/sample.png" alt="A sample image">

In this class, we follow contemporary recommendations and do **NOT** include a closing forward slash / in the void element's tag even though doing so works in all browsers.

#### Optional Resources

* [HTML: HyperText Markup Language](https://developer.mozilla.org/en-US/docs/Web/HTML) - MDN (Mozilla Development Network)
* [HTML Basics](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics) - MDN
* [HTML Basic Document Anatomy](https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics#Anatomy_of_an_html_document) - MDN
* [doctype](https://developer.mozilla.org/en-US/docs/Glossary/Doctype) - MDN
* [The lang attribute](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes#lang) - MDN

## Working with Web Images

### Overview

Most websites contain not only words, but also images. You will work with many different image formats and some image formats are better suited for the web than others. This activity explores using images for the web and how to optimize images so that they use less memory, thus supporting smaller page weights, which decreases the overall download time.

"The older formats like PNG, JPEG, GIF have poor performance compared to newer formats like WebP and AVIF, but enjoy broader "historical" browser support. The newer image formats are seeing increasing popularity as browsers without support become increasingly irrelevant (i.e. have virtually zero market share)." - MDN

### Prepare

* Video: ▶️ [Image File Types (JPG, PNG, SVG, GIF) - Web Design Tutorial](https://youtu.be/Zx0CacsiDZ4)
* Video: ▶️ [Image Optimization for Web Pages](https://youtu.be/85X16eLvNYY)

#### Basic Steps of Image Optimization

The original photos that we often work with are large in terms of dimensions and file size. Large photos need to be scaled down and optimized to a size that is actually needed on a web page. The optimal size is determined by the design and layout of the page. Software can be used to crop, reduce, and optimize the images. These are the basic steps supported by most photo editing packages:

1. **Crop** the original image to focus on the most important content, keeping in mind design principles (like the [rule of thirds](https://en.wikipedia.org/wiki/Rule_of_thirds)) and the site's purpose.
2. **Resize** the image to the maximum size that is required in the design of the web page. The resized image will then be saved in smaller formats to match the design and layout needed for viewing on smaller screens including on phones and tablets.
3. **Reduce the quality** of the image to an acceptable level which will reduce its file size.

Optimizing images is an easy way to improve page performance without requiring a significant time investment. More techniques for handling responsive images will be covered in future courses.

### Activity Instructions

All images must be optimized in this course in order to reduce overall page weight. The standard is that images need to be less than 100kB in memory size. If you have an image that is larger than 100kB, you will need to reduce its size by optimizing it.

#### Optimize an Image

1. In your browser, navigate to the online application, [Squoosh.app](https://squoosh.app/" \t "_blank).
2. Select the button in the middle of the page and then select an image from your computer.
3. In the **Edit** panel on the right, toggle **Resize** and change the width to ~200 pixels.
4. Under **Compress**, change the file type to **WebP**.
5. Under **Compress**, change the **Quality** to ~70%.
6. Note the overall change in file size displayed in those panels.
7. Click the download button (bottom right) to save the image to your computer.

#### Optional Resources

* [Web Image File Types](https://developer.mozilla.org/en-US/docs/Web/Media/Formats/Image_types) - MDN

[**Back**](https://byui-cse.github.io/wdd130-ww-course/week01/index.html)[**I-Learn**](https://byui.instructure.com/calendar/)

## Building Your Course Home Page

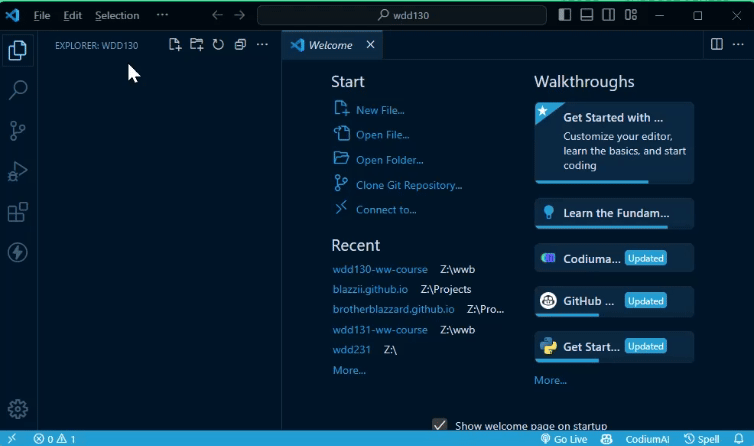
### Overview

In this assignment, you will apply your learning by creating a basic home page in HTML.

### Instructions

1. Create a course folder on your computer named "**wdd130**" (lower case, no spaces).
2. In VS Code, open that **wdd130** folder using **File -> Open Folder**.
3. Create a new file in that **wdd130** folder named "**index.html**". This will be your course home page.How do I create a new file within VS Code?

There are many ways to create a new file in Visual Studio Code, including:

* + Right-click in the Explorer panel and select **New File**.
  + Use the VS Code application pull-down menu by selecting **File** and then select **New File...**
  + Use the keyboard shortcut. Windows: **Ctrl+Alt+Windows+N** | Mac: **Cmd+N**
  + Select the **New File** icon in the Explorer panel (demonstrated below in an animation). 

"**index.html**" is a standard page name on web servers that will load by **default** unless changed by the web server administrator. This page will load if only the folder is referenced in a URL. Try it. In a browser window, go to ibm.com and then go to ibm.com/index.html. The same page will load in your browser.

1. In your index.html file, write the HTML markup to build the **basic HTML page anatomy** including:

For a reminder of the purpose of these elements, refer to the Learning Activity [An Introduction to HTML](https://byui-cse.github.io/wdd130-ww-course/week01/prepare-html-introduction.html)

* + the **document type**,
  + the **html** element with language attribute,
  + the **head** element within the html element's opening and closing tags, and
  + the **body** element within the html element and after the head.

Check Your Understanding

<!DOCTYPE html>

<html lang="en-US">

<head>

</head>

<body>

</body>

</html>

1. Within the **head** element, include the following from the required meta information:
   * Meta charset attribute
   * Meta viewport element
   * Title element
2. Set the **title** content to **[Your Full Name] | WDD 130** where [Your Full Name] is replaced with your actual, preferred full name.Check Your Understanding
3. <!DOCTYPE html>
4. <html lang="en-US">
5. <head>
6. <meta charset="utf-8">
7. <meta name="viewport" content="width=device-width,initial-scale=1.0">
8. <title>Xavier Rodriquez | WDD 130</title>
9. </head>
10. <body>
11. </body>

</html>

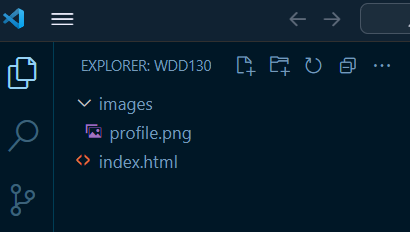
1. Within the **body** element, include three, semantic, child elements:
   * **header**
   * **main**
   * **footer**

to create the major sections of the page.

1. Within the **header** element, add a **nav** element with anchor tag **a** elements with the following **href** attribute values. The Rafting Website link will be used in a future assignment so just prepare now by linking to this future page even though it will not work now.
2. <header>
3. <nav>
4. <a href="#">Home</a>
5. <a href="wwr/">Rafting Website</a>
6. </nav>
7. </header>
8. <main>
9. </main>
10. <footer>

</footer>

The # value for the Home page href anchor link refers to the current page and will not reload the page. This is the behavior and structure that we want.

1. Within the **main** element:
   * add a **h1** heading element that contains **Your Full Name | WDD 130**
   * add an **img** element with the following attributes:
     + **src**: The src specifies the URL of the image file. It consists of the path and the file name with its extension. Use the following path: **src="images/profile.\_\_\_"** where \_\_\_ will be your file extension.
     + **alt**: The alt attribute is required as part of accessibility to provide alternative text to display when the image is not rendered by the browser. Go ahead and include your name in the alternative text.
     + **width**: Set the layout initial width of the image to **200**.
   * add a **p** paragraph element that contains information that you want to share about yourself.
2. To support the image reference, add a new folder named "**images**" to your wdd130 folder.Example File Structure
3. Copy or move an [optimized](https://byui-cse.github.io/wdd130-ww-course/week01/prepare-images.html) for the web **profile picture** file of yourself into this **images** folder.
   * The acceptable image **file types** for this assignment include:
     + png
     + webp
     + jpg
   * The image must **not exceed 100 kB** in file size.
4. Rename the image file **profile.** and use your image file extension.Check Your Understanding
5. <main>
6. <h1>Xavier Rodriquez | WDD 130</h1>
7. <img src="images/profile.jpg" alt="Hello, my name is Xavier Rodriquez" width="200">
8. <p>Hello! My name is Xavier Rodriquez and I am from Piura, Perú. I enjoy ... </p>

</main>

Video Demonstration: ▶️ [Optimizing an Image and Adding it to a Web Page](https://video.byui.edu/media/t/1_metas5fi) [4:51 minutes]

Is the image file not displaying! Check the instructions under the [File Sources and References](https://byui-cse.github.io/wdd130-ww-course/week01/resource-file-paths.html) help page to help resolve the issue.

1. Within the **footer** element, add a **p** paragraph element. The paragraph should contain the following:
   * The copyright symbol © and a copyright year.

To display the copyright symbol, use the **HTML entity** **&copy;** or use a built-in emoji.

To display the **emoji** menu, do the following:

* + - **Windows/Linux**: Hold the 🪟 window key on your keyboard and press period (.)
    - **macOS**: Hold down the following three keys at the same time: **Command+Control+Spacebar** or press the **fn** key.
  + Your name.
  + Your state or country.

Each of these items in the footer will be separated by syntax of your choice.Check Your Understanding

<footer>

<p>©️2025 🌴 Xavier Rodriquez 🌴 Piura, Perú</p>

</footer>

or, using HTML entities/signs of **&copy;** (©) and **&#10070;** (❖):

<footer>

<p>©2025 ❖ Xavier Rodriquez ❖ Piura, Perú</p>

</footer>

1. Check your rendered page by right mouse clicking on the index.html file name in the Explorer panel and selecting **Open with Live Server** or by using the menu item at the bottom of VS Code. Live Server will open up the page in your default browser.

Video Demonstration: ▶️ [Render Web Page Locally using Live Server in VS Code](https://www.loom.com/share/52a8250c6b494a4284aad9497770cd85) [0:07 minutes]

#### Complete Example

<!DOCTYPE html>

<html lang="en-US">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1.0">

<title>Xavier Rodriquez | WDD 130</title>

</head>

<body>

<header>

<nav>

<a href="#">Home</a>

<a href="wwr/">Rafting Site</a>

</nav>

</header>

<main>

<h1>Xavier Rodriquez | WDD 130</h1>

<img src="images/profile.jpg" alt="Hello, my name is Xavier Rodriquez" width="200">

<p>Hello! My name is Xavier Rodriquez and I am from Piura, Perú. I enjoy ... </p>

</main>

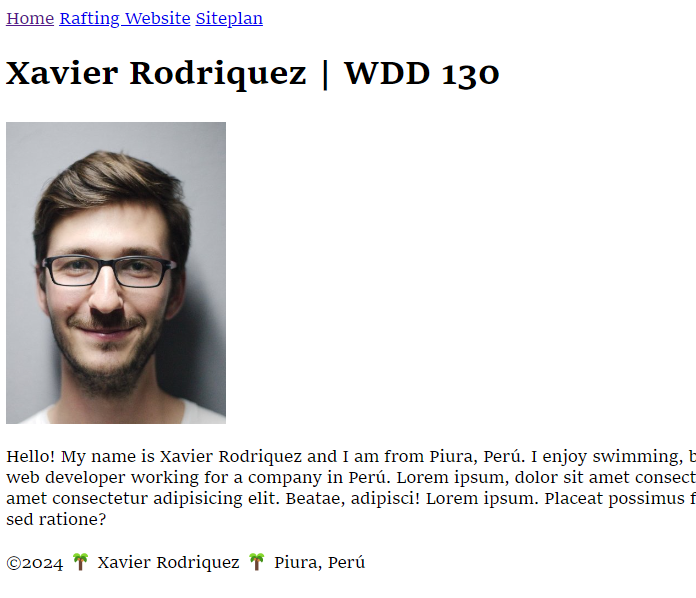
<footer>

<p>©2025 🌴 Xavier Rodriquez 🌴 Piura, Perú</p>

</footer>

</body>

</html>

Screenshot of Example Student Course Home Page

1. **Validate** your HTML by running the **Web Developer** extension in your browser.
   * Render the page on your browser using the **Live Server** tool in VS Code.
   * Using the Web Developer extension, select **Tools -> Validate Local HTML** to get an HTML validation report.
   * **Correct** any issues that you observe after testing the page and then test again.

Post your questions or look to help others in Microsoft Teams.

1. Continuously **save** your work.

It is recommended that you turn on Auto Save in VS Code. Go to **File -> Preferences -> Settings** and then type "Auto Save" in the search bar at the top of the panel. Use the pull-down menu to select **afterDelay**. Set the **Files: Auto Save Delay** field to something like 1000 ms (1 second).

• [Save / Auto Save](https://code.visualstudio.com/docs/editor/codebasics#_save-auto-save) - code.visualstudio.com

### Submission

1. You will be committing and pushing/uploading your work to your remote wdd130 repository on GitHub for publication in the next activity using a service called **GitHub Pages**.

#### Optional Resources

* [Images](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/img) - MDN
* [Image File Types](https://developer.mozilla.org/en-US/docs/Web/Media/Formats/Image_types) - MDN
* [HTML Symbols / Entities](https://www.w3schools.com/html/html_symbols.asp) - w3schools.com

[**Next**](https://byui-cse.github.io/wdd130-ww-course/week01/setup-github-pages.html)[**Back**](https://byui-cse.github.io/wdd130-ww-course/week01/index.html)[**I-Learn**](https://byui.instructure.com/calendar/)

## Setup: GitHub Pages

### Overview

In this course, you will be publishing your work to your own website which is freely served through GitHub Pages. Your peers, graders, and the instructor will then be able to access your site for collaboration and feedback. You have already established a GitHub account and installed git technology so that you can commit and push/upload your work to your GitHub repository. We will be doing that inside of VS Code.

"GitHub Pages is a static site hosting service designed to host your personal, organization, or project pages directly from a GitHub repository" - docs.github.com

### Instructions

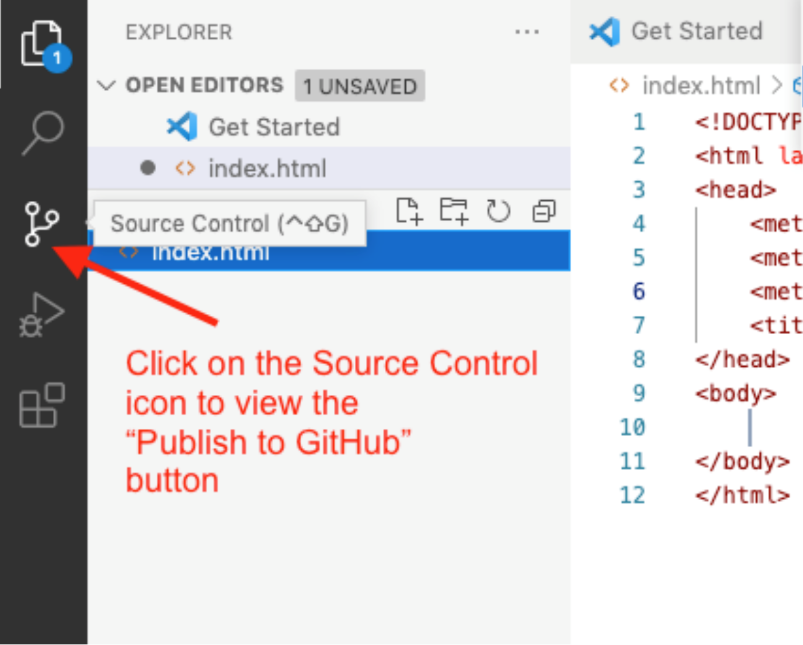
#### Publish the wdd130 folder to GitHub

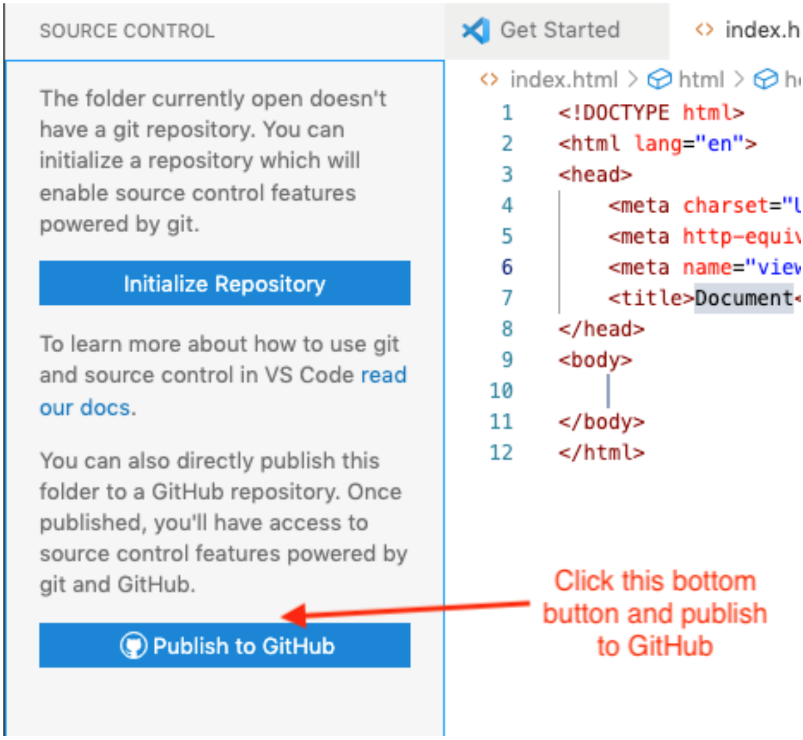
1. In VS Code, open your **wdd130** folder if it is not already open.

The system may prompt you to trust the file source. Click **Trust** to continue.

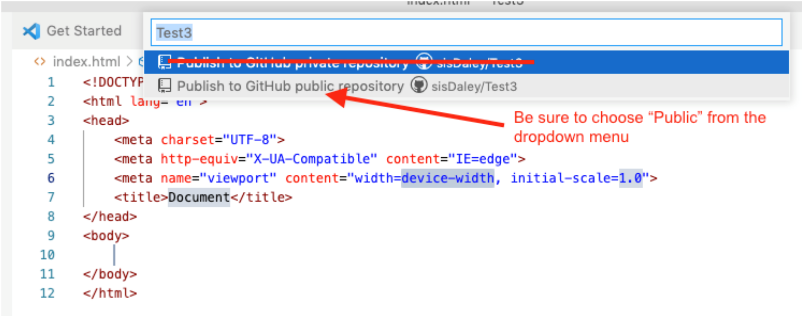
1. Select the **Source Control** icon in VS Code's Activity Bar.

By default, the **Activity Bar** is found on the left-hand side of the screen and will have up to five icons. If the Activity Bar is not visible, click **View -> Appearance -> Activity Bar** to turn it visible.

Source Control Icon in VS Code's Activity Bar

1. Click the blue button that says **Publish to GitHub**. If you don't see this button, make sure you have all other git repo folders closed.Publish to GitHub Button in the Source Control panel
2. Click **Allow** if you get a message saying "The extension 'GitHub' wants to sign in using GitHub."
3. If you are prompted for an authorization for Visual Studio Code to Access GitHub, click **Continue**.
4. If you are asked to allow the page to open "Visual Studio Code", click **Allow**.
5. You may also be asked to allow an extension to open this URI, click **Open**.
6. Visual Studio Code will choose the folder you have open, the **wdd130** folder, as the repository and give you the option to "Publish to GitHub public repository". Choose the one that says **public**, not private. This will make a new public repository in our GitHub account.

Private repositories are **not allowed** for course work.

Publish to GitHub public repository

1. It will show a list of the files in the wdd130 folder. They will be selected by default so click **OK** to include the files in the new repository.
2. You may get a prompt to "Authorize Git Credential Manager". Click **Authorize**.
3. If you get a message asking if you'd like to periodically run fetch you can say "Yes".

If you look at your GitHub account on github.com, a new repository named wdd130 will be there along with any files or folders that were in your local **wdd130** folder. This local folder is now connected to the remote repository on GitHub and is a repository itself.

If you have any issues with the publication process, please reach out to your instructor for help **immediately**. Publishing your wdd130 folder to GitHub is a critical step in the course as your **wdd130** repository is used for tracking your progress and for grading each week.

It is fine to delete a repository on GitHub and start over if needed. The **Delete Repository** option is found in the repository's **Settings** menu.

Video Demonstration: ▶️ [Publish the wdd130 repository folder to a GitHub repository](https://youtu.be/mrGMxZkkIzg)

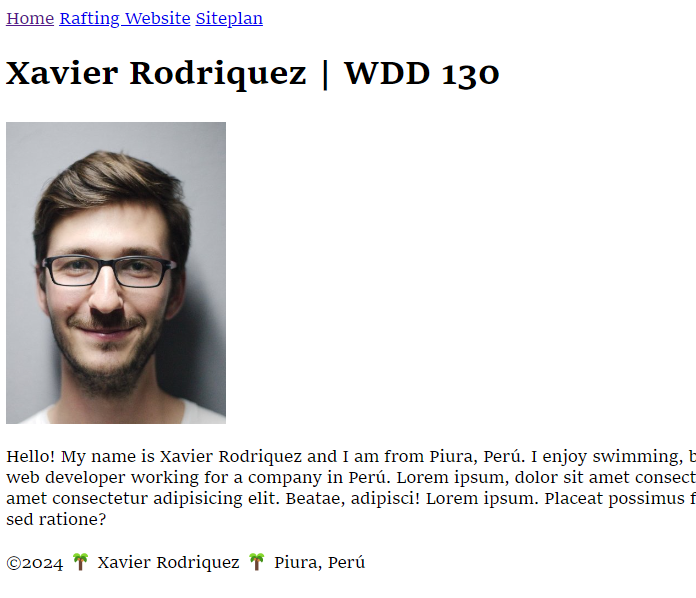
#### Enable GitHub Pages

1. Login to your [GitHub](https://github.com/) account if needed.
2. Open your **wdd130** repository.
3. Click **⚙️ Settings** in the repository menu.
4. Click **Pages** in the submenu that appears on the right of the settings panel.
5. Under the **Build and deployment** section of GitHub Pages, select the **main** branch.  
   Leave the next pull down menu with the default value of **📁 / root**.

If your branch is named **master** instead of **main**, that is OK.

1. In the top section, a message should display something like **"Your site is live at https://yourgithubusername.github.io/wdd130"**. This is your course home page URL and this URL domain will be part of every submission you make in the course.

Any time you publish to your GitHub Pages enabled wdd130 repository, the updates will take a few minutes before the pages are available for viewing.

1. Note and bookmark the URL link above for quick access in your browser.
2. That is all that needs to be done to enable GitHub Pages site publishing. After a few minutes, **test** the URL. Your course home page should open, rendered in the browser.Example Student Course Home Page Rendered in the Browser
3. **Test** the content of your page using the ✔ [Page Evaluation](https://byui-cse.github.io/wdd130-ww-course/grader/w01-homepage.html) tool.
4. **Share** your GitHub Pages wdd130 URL with a family member or friend.
5. Remember to **Bookmark** or **Favorite** your GitHub Pages enabled wdd130 URL.

Video Demonstration: ▶️ [Enable GitHub Pages](https://video.byui.edu/media/t/1_p8kxx5os)

### Submission

1. After you have completed the work, return to I-Learn to submit your GitHub Pages URL which will have the following format:

**https://yourgithubusername.github.io/wdd130/**

#### Optional Resources

* [Getting Started with GitHub Pages](https://docs.github.com/en/pages/getting-started-with-github-pages/about-github-pages) - GitHub Documentation

## Page Evaluation Tool: W01 - Home Page

Enter your GitHub Username:  Run Report

<https://byui-cse.github.io/wdd130-ww-course/grader/w01-homepage.html>