# Pre-Course

## Units & Lessons - Intro to website, lessons, etc.

## Exercises, Quizzes & Assignments -Save work on JSBin

# Developer Tools I - - How the Internet works (HTTP, DNS, Servers, Clients, IPs)

## Text Editors and Browsers - Text Editors edit plain text files, Web Browsers comm with servers and display files.

## Sublime Text - Sublime shortcuts

## Quiz Done!

# Introduction to HTML - Overview HyperText Markup Language - Element is any content between annotations/tags - -

## HTML Elements - <h1>, <p>,<a> - Attributes

### Exercise

## HTML Boilerplate

### Exercise

## Quiz

## Cheat Sheet

**WDI Fundamentals Unit 2**

# HTML Cheat Sheet

#### World Wide Web

* An information sharing model built on top of a global system of interconnected computer networks called the internet.

#### Websites

* Collections of files, accessible through the world wide web.

#### Clients

* Personal devices - such as computers, phones, or tablets - used to access the web.

#### Servers

* Used to host the massive amounts of files that make up websites and “serve” files to end users.

#### Elements

* The building blocks of HTML. Consists of a start tag, content, and end tag.
  + **Headline Elements** are used for important text — like page titles — that describe content that comes after it. They range in order from the largest (h1) to the smallest (h6).

<h1>This would make a big headline</h1>

* + **Paragraph Elements** use one of the most basic tags in HTML. They indicate blocks of text.

<p>This would add a block of text to a page</p>

* + **Anchor Elements** create links. In order to make a working link, we need to add more information to the anchor tag using an (href) attribute that detemines the link's destination.

<a href="www.generalassemb.ly"> This would make this whole sentence a link to General Assembly's home page.</a>

#### Tags

* HTML syntax used to create elements. They use angle bracket bookends (< >) to tell a web browser how to present content.
  + **Heading Tags:** <h1> (largest size) through <h6> (smallest size) titles the content that follows it.
  + **Paragraph Tags:** <p> tags indicate blocks of text.
  + **Anchor tags:** <a> tags create links.

#### HTML Attribute

* Adds information to an HTML element. An HTML tag can have one or many attributes. Attributes are always included in the opening tag. They are preceded by a space, include the name of the attribute, an equal sign, and a value in quotes.

#### HTML Boilerplate

In order to organize tags properly, start with a set of structural elements called the HTML boilerplate. It should look like this:

<!DOCTYPE html> <html> <head> </head> <body> </body> </html>

## Unit 2 Assignment

# Introduction to CSS

## Adding Color

## Fonts and Texts

### Exercise

## Classes and IDs

## Files and Folders

## Working with Images

## Choosing the Right MarkUp

### Exercise

## Quiz

# Cheat Sheet CSS Cheat Sheet

#### Rule

The building block of a CSS stylesheet. A rule consists of a selector and a declaration block (one or more declarations).

#### Declaration

A declaration is made up of a property and a value, separated by a colon and punctuated by a semi-colon.

#### Selector

The actual HTML object that the declaration(s) apply to.

#### Property

The characteristic of the selector that will be changed.

#### Value

The amount or type of change to be applied to the corresponding property of the matched selector.

#### Marking-Up

The process of assigning HTML tags to given text content in order to indicate its relation to the rest of the text or dictate how it should be displayed.

#### Serif Font

One of two general categories of fonts (typefaces) that uses marks (called “serifs”) to embellish characters at the end of strokes. A common serif font is Times New Roman.

#### Sans-Serif Font

One of two general categories of fonts that do not embellish characters at the end of strokes ("sans serif" means “without serif”). A common sans-serif font is Helvetica.

#### Class

A class attribute is added to an HTML element in order to provide a “hook” to refer to that element in your CSS. CSS class selectors begin with a “.”. Classes can be used multiple times per page.

#### ID

An ID attribute is added to an HTML element in order to provide a “hook” to refer to that element in your CSS. CSS ID selectors begin with a “#”. Unlike classes, IDs can only be used one time per page.

#### Image

Add images to your HTML using the img tag. Tell the browser the source of the image file with an srcattribute.

#### Absolute File Path

A path to a website or file that includes a full web address (starting with “http”) that the browser loads from the remote location directly. For example:

<img src="http://imgur.com/awesomedog.jpg">

#### Relative File Path

A path to a website or file that gives you the path to the resource you are looking for as it relates to your website's local file structure. For example, if you wanted to retrieve an image called "newlogo.png" that resides in a directory called images, you would enter the following relative address:

<img src="images/newlogo.png">

#### Why Separate HTML from CSS?

Separating HTML from CSS offers you scalability and versatility. If you separate how your site looks from what your site says, things become more flexible. You can make a change in one place and have it apply to your whole site, and apply any number of different styles to the same content.

#### CSS Color Treatment

While color names are fine when you're just beginning, there's a number of reasons you'll want to switch over to something more advanced. First, color names are rendered differently by different browsers. Second, there are only 147 color names accepted as standard, meaning your options are pretty limited.

Instead, you'll want to use either RGB or hexadecimal codes. Both of these are built on a system of entering values for the colors red, green, and blue.

By mixing different intensities of these three colors, you can create millions of different colors and shades. Intensity values range from 0 (no intensity) to 255 (full intensity) in the RGB system.

In hex, they range from 0-9, then continue from A-F, with two characters each for red, green, and blue. This is clearer with examples, so here are some common colors with their RGB and Hex equivalents.

The format for color names, RGB, and hexadecimal should look like the following, respectively:

p { color: red; } p { color: (255,0,0); } p { color: #FF0000; }

#### CSS Text Treatment

**font-family**

To adjust the font of your selected text element, use the font-family property. For the value, enter the name of the font to which you’d like to alter your text. To be safe, put a comma after your selected font and enter a generic family as a fallback. If the web browser doesn’t support the font you selected, it will choose the fallback.

h1 { font-family: Arial, sans-serif; }

If your selected font is more than one word, capitalize both words and put them in quotation marks.

h1 { font-family: "Courier New", monospace; }

**font-size**

To increase or decrease the spacial dimensions of your chosen text, use the font-size property. As a beginner, you should enter pixel values for your font-size values.

h3 { font-size: 24px; }

As you become more advanced, try using percentages or ems instead of pixels, which we'll cover in detail in a later unit.

**font-weight**

To adjust the thickness of your selected text, use the font-weight property.

As a beginner, you can enter values like “normal” to make your text thin and “bold” to make your text thick. As these values aren’t very specific, different browsers may interpret their display with slightly different outputs.

h1 { font-weight: normal; } h1 { font-weight: bold; }

As you become more advanced, try using the numbers 100, 200, 300, 400, 500, 600, 700, 800, and 900 as values to gain more granular control. With this system, 400 is roughly equivalent to “normal” and 700 roughly equals “bold.”

h1 { font-weight: 400; } h1 { font-weight: 700; }

**font-style**

To make normal text italic, use the property font-style and the value “italic.” To reverse this effect, use the value “normal.”

a { font-style: italic; } a { font-style: normal; }

**text-align**

To adjust the positioning of a text element, use the CSS property "text-align" and one of the following values: left, right, center, or justify.

body { text-align: center; }

**text-decoration**

To add an underline to normal text, use the CSS property "text-decoration" and the value “underline.”

h1 { text-decoration: underline; }

To remove underlines, use the value “none.” This declaration is often applied to anchor tags.

a { text-decoration: none; }

Less commonly used values include “overline” to add a line above text and “line-through” to strike a line through your text.

**text-transform**

To adjust capitalization in a selected text element, use the "text-transform" property.

Values for this property include “uppercase” to make every letter capitalized, “lowercase” to make every letter uncapitalized, and “capitalize” to make the first letter of every word in the selected text capitalized.

h1 { text-transform: uppercase; }

#### Classes vs. IDs

Classes and IDs, also called selectors, are ways of targeting the style of specific HTML elements on your page.

So, what’s the difference between them? In short, classes allow you to style many elements with a particular style, while IDs are only capable of styling a single element.

Using these selectors allows you flexibility and control of styling individual, as well as groups, of elements on your page.

#### HTML and Images

When you’d like to add images to your site, you should use the tag with the “src” attribute. “src” stands for “source” and works just like the tag’s “href” attribute. It tells the image tag where to find the image you’d want to include on your page.

#### Relative vs. Absolute Addressing

Relative addressing basically tells your browser: “Look in our project folder for the file specified." Or, if a folder is specified: "Look within that folder for the file specified.”

On the other hand, when we link to a source outside of our project, it is referred to as an absolute address.

## Unit 3 Assignment

# Layout Basics

## Divs and Span

## The Box Model

### Exercise

## Header, Footer, Nav

## Semantic Elements

### Exercise

## Quiz

Cheat Sheet  
  
**WDI Fundamentals Unit 4**

# Layout Basics Cheat Sheet

#### Span

* A generic wrapper for any inline content. Spans are generally used to group small sections of content for styling purposes.

#### Div

* A generic wrapper for any block content. Divs are used to group other elements together or to provide style to a specific area.

#### Header

* These elements kick off your page with introductory content like logos, headlines, titles, and links.

#### Footer

* This element wraps the content at the bottom of your web page. This element often contains copyright information, links to career pages, contact information, terms of use, etc.

#### Main

* These elements contain all the primary content — articles, blog posts, images, videos, etc. — between the <header> and the <footer>.

#### Aside

* These elements contain all the secondary content between the <header> and <footer>, including supplemental information like recommended stories or archived blog links.

#### Nav

* Any navigational links that help users get around your page should be wrapped in <nav> tags.

#### Section

* These elements are used to group the content of a page into related chunks.

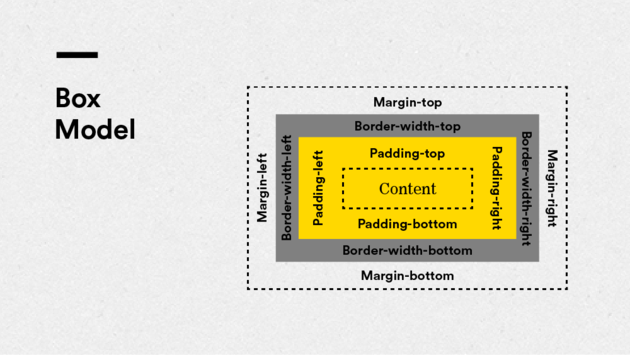
#### Article

* These elements are used to contain standalone blocks of text, such as news articles, blog posts, etc.

#### Box Model

* Every HTML element on the page is in a box, and the box model dictates what the element will look like. The model consists of the content, padding, border, and margin.

Here's a diagram to help you remember the box model.



#### Content

* The text and images that are included within any element's opening and closing tags.

#### Padding

* The area outside the content, but within the border. Padding will extend the background-color and contribute to the overall width of the object.

#### Border

* A line of variable thickness, color, and style that goes around the padding and content.

#### Margin

* The area outside of the border that separates the object from other objects around it. Margin has a transparent background-color.

#### Inline Elements

* HTML elements that take up the width of their contents and do not begin with new line. They cannot have height or width assigned.

#### Block Elements

* HTML elements that break to a new line before and after the element and they take up the width of their containers by default (this is often the browser window itself). They can have padding, margin, height, and width assigned.

#### Span

* Inline and used to apply CSS to inline HTML.

#### Div

* A block element used to apply CSS to blocks of HTML. Divs typically wrap big sections of content on a page. Use semantic elements over divs whenever possible.

## Unit 4 Assignment

# Navigation and Webpage Layout

## Vertical Navigation

## Floating and Clearing

## Horizontal Navigation

### Exercise

## Column Layout

### Exercise

## Quiz

Cheat Sheet  
  
**WDI Fundamentals Unit 5**

# Navigation and Layout Cheat Sheet

#### Navigation Bar

* A section on a web page with links to other pages or parts of the site. It can appear as a vertical, horizontal, or alternatively styled block.

#### Pseudo-class

* A keyword added to selectors that indicates a particular condition of the element in which you would apply CSS styling. For example, :hover will apply a style when a user hovers over the element specified by the selector.

#### Nav

* A semantic element used to define a set of navigation links (<nav> </nav>).

#### Clearfix

* A technique in CSS used to automatically clear after an element. It's generally used in column layouts where elements are floated. It involves giving the floated elements the class "clearfix," and creating a corresponding CSS rule for that class using the pseudo-class :after.

.clearfix:after { visibility: hidden; display: block; font-size: 0; content: " "; clear: both; height: 0; }

#### CSS Reset

* There is no true standard for how unstyled HTML elements are rendered in various browsers. In other words, each browser assigns its own padding, margin, line-height, font-size, etc. to every element. To combat this, we reset the CSS, essentially wiping out all of its default styles and starting with a clean slate. Always link to reset.css in the head section of your HTML above to your main CSS file, like so:

<!DOCTYPE html> <html> <head> <title>CSS Reset Links</title> <link rel="stylesheet" type="text/css" href="css/reset.css"> <link rel="stylesheet" type="text/css" href="css/style.css"> </head> <body> </body> </html>

#### Floats and Clears

By default, block elements in HTML stack on top of each other. Floats let us change this behavior, allowing elements to sit horizontally on the page.

This CSS property is called float, and its common values are left, right, and none (the default is none).

Clears are used when we want to “turn off” floating — essentially preventing elements from flowing around the floating element.

A common method involves using the CSS property clear with the value "both." When this method is ineffective, the clearfix method — as seen above — should resolve any issues.

#### Column Layout

Column layout refers to the design of pages composed of multiple block sections with the same vertical height.

Columns are created by wrapping a block of content with a div element, applying a float, and modifying the padding, margin, and border dimensions to adjust their size.

## Unit 5 Assignment

## Pulse Check

# Developer Tools II

## Think Like a Programmer

### Exercise

## Navigating the Command Line

### Exercise

## Moving and Removing Things

### Exercise

## Quiz

Cheat Sheet  
  
**WDI Fundamentals Unit 6**

## Command Line Cheat Sheet

#### Command Line

* A text-based interface.
* Synonyms: command line interface (CLI) console

#### Terminal

* An OS X application that provides text-based access to a computer's operating system.
* Any device or application used for data entry and display in a computer system.
* Synonyms: client, computer terminal, terminal emulator

#### File System

* A file system is a systematic way to control how information is stored and retrieved on a computer. It describes where one piece of information stops and the next one begins. Each file system has its own structure and logic.
* Synonyms: NTFS (Windows' File System), HFS+ (Apple's File System), file allocation table, GFS (Global File System)

#### Directory

* A unit, or container, used to organize computer files into a hierarchical structure.
* Synonyms: folder, catalog, drawer

#### Path

* A sequence of symbols and names that identifies a file or directory. The path always starts from your working directory or from the root directory, and each sub-directory is followed by a forward slash.
* An absolute, or full, path begins with the root directory and specifies every directory above the terminating file or directory name.
* A relative path does not include the root or parent directory names and refers to a file or directory directly below the current working directory.
* Synonyms: path name

#### Command

* The action we want the computer to take; always a single word.
* Synonyms: utility

#### Option

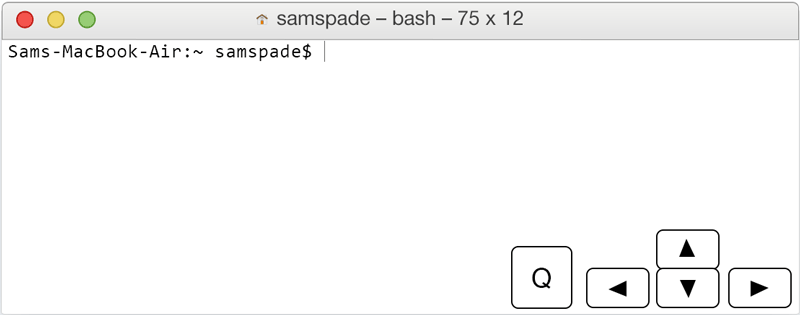
* Follows the "command" in a command line; used to modify the behavior of the command in some way.
* Synonyms: flag

#### Argument

* Follows the "command" and "options" (if any) in a command line and is used to explain what we want the command to act upon.
* The number of arguments used generally depends on the command; some don't need arguments, some require exactly one argument, some require many arguments, and some are flexible in the number they can take.

| **Command** | **Description** |
| --- | --- |
| pwd -options | Prints the working directory; returns the absolute path name of the current directory |
| ls [-options] [path/to/directory] | Lists directory contents |
| cd [-options] [path/to/directory] | Changes the current working directory to the specified directory |
| mkdir [-options] [path/to/directory] | Makes a new directory |
| rm -r [path/to/file] [path/to/file] ... | Removes directories or files permanently |
| mv [-options] [path/to/file] [path/to/directory] | Moves directories or files to a new locale |
| mv [-options] [path/to/file] [NEW\_FILE\_NAME] | Renames a file or directory |

On Mac, your terminal comes with a manual. To access more (a lot more) information about any command, type man, followed by the command name, and press Enter:



You can scroll through a manual entry using the arrow keys or space bar. To quit this view and return to your prompt, type q.

## Unit 6 Assignment

# Developer Tools III

## Track Changes with Git

### Exercise

## Introducing GitHub

### Exercise

## Quiz

## Cheat Sheet

## Unit 7 Assignment

# Introduction to JavaScript

## Thinking Programatically

### Exercise

## Expressions and Evaluations

### Exercise

## Variables

### Exercise

## Boolean Logic

### Exercise

## Quiz

Cheat Sheet  
  
**WDI Fundamentals Unit 8**

# Expressions & Variables Cheat Sheet

Here are some notes on what's been covered in this chapter. Feel free to copy this information and extend it to make your own cheat sheet.

#### Pseudocode

* A way to plan out your program before coding it. It is a detailed, step-by-step description of what a computer must do, expressed in plain English rather than in a programming language.

#### Expressions

* An expression is a statement composed of values/data and operators.
* Some common data types are numbers, strings, and Booleans.
* An operator takes in a number of inputs but outputs/evaluates to a single value.
* To determine how an expression is evaluated, look at what each operator's inputs are and, if necessary, generate an expression tree to illustrate the expression's structure.

#### Variables

* The purpose of variables is to store and reuse the values created from a computation.
* A variable is assigned a value using the = operator. First, the expression to the right of the = is evaluated. Then, this value is assigned to the variable to the left of the =. Finally, the = operator evaluates to the value that has just been assigned.
* To use the value that a variable is storing, simply include that variable in an expression. An expression containing variables will evaluate just like one without variables, except that the variables will themselves be evaluated as part of the expression. As before, it is possible to draw an expression tree to illustrate the expression's structure.
* When a variable is redefined, it retains **no knowledge** of any prior values it may have held.
* A variable may be redefined "in place" using an expression like x = x + 1 (or its shorthand, x += 1).
* An expression like x = y only means that the value that y had been holding is now also held in x. **It does not imply any lasting relationship between x and y**.

#### Special Cases

* When a variable is created but is not assigned a value, it will be evaluated as undefined.
* Any type of value, including null, can be passed into a logical operator as an input; based on whether these inputs are either "truthy" or "falsey" and the type of operator you're dealing with, the operator will behave in different ways.

#### Comparison Operators

| **Operator** | **Meaning** | **True expressions** |
| --- | --- | --- |
| == | Equality | 10 == '10' |
| === | Strict equality | (2 \* 5) === 10 |
| != | Inequality | 9 != 10 |
| !== | Strict inequality | '10' !== 10 |
| > | Greater than | 20 > 10 |
| >= | Greater than or equal to | '10' >= 10 |
| < | Less than | 10 < 30 |
| <= | Less than or equal to | '10'<= 10 |

#### Logical Operators

Logical operators work on Boolean values to produce Boolean results.

**AND operator &&**

| **Condition 1** | **Condition 2** | **Result** |
| --- | --- | --- |
| true | true | true |
| true | false | false |
| false | true | false |
| false | false | false |

**OR operator ||**

| **Condition 1** | **Condition 2** | **Result** |
| --- | --- | --- |
| true | true | true |
| true | false | true |
| false | true | true |
| false | false | false |

**NOT operator !**

| **Condition** | **Result** |
| --- | --- |
| true | false |
| false | true |

We can use parentheses to change the order of operations for logical operators, just like we do in mathematics.

## Unit 8 Assignment

# Control Flow in JS

## Conditionals

### Exercise

## Loops

### Exercise

## Quiz

## Cheat Sheet

## Unit 9 Assignment

# Functions & The DOM

## Defining and Calling Functions

### Exercise

## Problem Solving with Functions

### Exercise

## Accessing the DOM

### Exercise

## Manipulating the DOM

### Exercise

## Quiz

## Cheat Sheet

## Unit 10 Assignment

# Collections

## Arrays

### Exercise

## Iterating over Arrays

### Exercise

## Objects

### Exercise

## Quiz

## Cheat Sheet

## Unit 11 Assignment

## Pulse Check