Web Development Pick-Me-Ups

# HTML

## Basic

* **HTML** stands for **H**yper**T**ext **M**arkup **L**anguage and is used to create the structure and content of a webpage.
* Most HTML elements contain opening and closing tags with raw text or other HTML tags between them.
* HTML elements can be nested inside other elements. The enclosed element is the child of the enclosing parent element.
* Any visible content should be placed within the opening and closing <body> tags.
* Headings and sub-headings, <h1> to <h6> tags, are used to enlarge text.
* <p>, <span> and <div> tags specify text or blocks.
* The <em> and <strong> tags are used to emphasize text.
* Line breaks are created with the <br> tag.
* Ordered lists (<ol>) are numbered and unordered lists (<ul>) are bulleted.
* Images (<img>) and videos (<video>) can be added by linking to an existing source.
* Syntax: <img src="smiley.gif" alt="Smiley face" height="42" width="42">
* “alt” is used to specify stuff when the image doesn’t load or give a description of the image
* Metadata is information about the page that isn’t displayed directly on the web page. Unlike the information inside of the <body> tag, the metadata in the head is information about the page itself.
* The <!DOCTYPE html> declaration should always be the first line of code in your HTML files. This lets the browser know what version of HTML to expect.
* The <html> element will contain all of your HTML code.
* Information about the web page, like the title, belongs within the <head> of the page.
* You can add a title to your web page by using the <title> element, inside of the head.
* A webpage’s title appears in a browser’s tab.
* Anchor tags (<a>) are used to link to internal pages, external pages, or content on the same page.
* You can create sections on a webpage and jump to them using <a> tags and adding ids to the elements you wish to jump to.
* Whitespace between HTML elements helps make code easier to read while not changing how elements appear in the browser.
* Indentation also helps make code easier to read. It makes parent-child relationships visible.
* Comments are written in HTML using the following syntax: <!-- comment -->.

## Tables

* The <table> element creates a table.
* The <tr> element adds rows to a table.
* To add data to a row, you can use the <td> element.
* Table headings clarify the meaning of data. Headings are added with the <th> element.
* Table data can span columns using the colspan attribute.
* Table data can span rows using the rowspan attribute.
* Tables can be split into three main sections: a head, a body, and a footer.
* A table’s head is created with the <thead> element.
* A table’s body is created with the <tbody> element.
* A table’s footer is created with the <tfoot> element.

## Forms

* The purpose of a <form> is to allow users to input information and send it.
* The <form>‘s action attribute determines where the form’s information goes.
* The <form>‘s method attribute determines how the information is sent and processed.
* To add fields for users to input information we use the <input> element and set the type attribute to a field of our choosing:
  + Setting type to "text" creates a single row field for text input.
  + Setting type to "password" creates a single row field that censors text input.
  + Setting type to "number" creates a single row field for number input.
  + Setting type to "range" creates a slider to select from a range of numbers.
  + Setting type to "checkbox" creates a single checkbox which can be paired with other checkboxes.
  + Setting type to "radio" creates a radio button that can be paired with other radio buttons.
  + Setting type to "list" will pair the <input> with a <datalist> element if the id of both are the same.
  + Setting type to "submit" creates a submit button.
  + The “placeholder” attribute is displayed in the field while there is no input and disappears after the user interacts. It can be used instead of the label
* A <select> element is populated with <option> elements and renders a dropdown list selection.
* A <datalist> element is populated with <option> elements and works with an <input> to search through choices.
* A <textarea> element is a text input field that has a customizable area.
* When a <form> is submitted, the name of the fields that accept input and the value of those fields are sent as name=value pairs.

### Validation

* Client-side validations happen in the browser before information is sent to a server.
* Adding the required attribute to an input related element will validate that the input field has information in it.
* Assigning a value to the min attribute of a number input element will validate an acceptable minimum value.
* Assigning a value to the max attribute of a number input element will validate an acceptable maximum value.
* Assigning a value to the minlength attribute of a text input element will validate an acceptable minimum number of characters.
* Assigning a value to the maxlength attribute of a text input element will validate an acceptable maximum number of characters.
* Assigning a regex to pattern matches the input to the provided regex.
* If validations on a <form> do not pass, the user gets a message explaining why and the <form> cannot be submitted.

## Semantic HTML

* Semantic HTML introduces meaning to a page through specific elements that provide context as to what is in between the tags.
* Semantic HTML is a modern standard and makes a website accessible for people who use screen readers to translate the webpage and improves your website’s SEO.
* <header>, <nav> , <main> and <footer> create the basic structure of the webpage.
* <section> defines elements in a document, such as chapters, headings, or any other area of the document with the same theme.
* <article> holds content that makes sense on its own such as articles, blogs, comments, etc.
* <aside> contains information that is related to the main content, but not required in order to understand the dominant information.
* <figure> encapsulates all types of media.
* <figcaption> is used to describe the media in <figure>.
* <video>, <embed>, and <audio> elements are used for media files.

# CSS

CSS, or **Cascading Style Sheets**, is a language that web developers use to style the HTML content on a web page. If you’re interested in modifying colors, font types, font sizes, shadows, images, element positioning, and more, CSS is the tool for the job!

## Intro

* CSS can change the look of HTML elements. In order to do this, CSS must select HTML elements, then apply styles to them.
* CSS can select HTML elements by tag, class, or ID.
* Multiple CSS classes can be applied to one HTML element.
* Classes can be reusable, while IDs can only be used once.
* IDs are more specific than classes, and classes are more specific than tags. That means IDs will override any styles from a class, and classes will override any styles from a tag selector.
* Multiple selectors can be chained together to select an element. This raises the specificity, but can be necessary.
* Eg: { .nutrition ui }– means ui child inside nutrition class
* { ui.nutrition} – means ui tag *with* class nutrition
* Nested elements can be selected by separating selectors with a space.
* The !important flag will override any style, however it should almost never be used, as it is extremely difficult to override.
* Multiple unrelated selectors can receive the same styles by separating the selector names with commas.

## Rule-Sets

* CSS declarations are structured into property and value pairs.
* The font-family property defines the typeface of an element.
* font-size controls the size of text displayed.
* font-weight defines how thin or thick text is displayed.
* The text-align property places text in the left, right, or center of its parent container.
* Text can have two different color attributes: color and background-color. color defines the color of the text, while background-color defines the color behind the text.
* CSS can make an element transparent with the opacity property.
* CSS can also set the background of an element to an image with the background-image property. Syntax: background-image: url(“location-of-image.png”);
* A page’s styling (font sizes, colors, etc.) is outlined in a design spec, which is a standard document you’d expect to receive as a freelance web developer.

Eg:

## Data Paths

* Form: “D:/Folder/AnotherFolder/file.typ”
* Notations: “./” used to denote the directory of the current file you’re working on.

Eg. For index.html, the folder which contains it is denoted as “./”

* Notation: “../” used to denote the directory containing the current files directory (if the current is not the root).

Eg. If path to file is “D:/folder/root/sub-root/file.html”, then “../” denotes: “D:/folder/root/”

And “./” denotes “D:/folder/root/sub-root”

## The Box Model

1. The box model comprises a set of properties used to create space around and between HTML elements.
2. The height and width of a content area can be set in pixels or percentage.
3. Borders surround the content area and padding of an element. The color, style, and thickness of a border can be set with CSS properties.
4. Padding is the space between the content area and the border. It can be set in pixels or percent.
5. Margin is the amount of spacing outside of an element’s border.
6. Horizontal margins add, so the total space between the borders of adjacent elements is equal to the sum of the right margin of one element and the left margin of the adjacent element.
7. Vertical margins collapse, so the space between vertically adjacent elements is equal to the larger margin.
8. margin: 0 auto horizontally centers an element inside of its parent content area, if it has a width.
9. The overflow property can be set to display, hide, or scroll, and dictates how HTML will render content that overflows its parent’s content area.
10. The visibility property can hide or show elements.

Box Model: Content-box (box-sizing: content-box)

Box Model: Border-Box

1. In the default box model, box dimensions are affected by border thickness and padding.
2. The box-sizing property controls the box model used by the browser.
3. The default value of the box-sizing property is content-box.
4. The value for the new box model is border-box.
5. The border-box model is not affected by border thickness or padding.

* In use <class/element/tag name>:hover to make it change during hover, and have 2 properties, eg:
  + .class{
  + //Default look
  + }
  + .class:hover{
  + //hover look
  + }

## Positioning

* A browser will render the elements of an HTML document that has no CSS from left to right, top to bottom, in the same order as they exist in the document. This is called the flow of elements in HTML.
* The position property allows you to specify the position of an element in three different ways.
* When set to relative, an element’s position is relative to its default position (not the PARENTS position) on the page.
* When set to absolute, an element’s position is relative to its closest positioned parent element (or nearest non-static element). It can be pinned to any part of the web page, but the element will still move with the rest of the document when the page is scrolled.
* When set to fixed, an element’s position can be pinned to any part of the web page. The element will remain in view no matter what.
* The z-index of an element specifies how far back or how far forward an element appears on the page when it overlaps other elements. Ignored by static elements
* The display property allows you control how an element flows vertically and horizontally a document.
* inline elements take up as little space as possible, and they cannot have manually-adjusted width or height.
* block elements take up the width of their container and can have manually-adjusted heights.
* inline-block elements can have set width and height, but they can also appear next to each other and do not take up their entire container width. Doesn’t start new lines, can flow horizontally with siblings
* The float property can move elements as far left or as far right as possible on a web page.
* You can clear an element’s left or right side (or both) using the clear property.

## Colors

There are four ways to represent color in CSS:

* Named colors — there are 147 named colors, which you can review [here](https://msdn.microsoft.com/en-us/library/aa358802(v=vs.85).aspx).
* Hexadecimal or hex colors
  + Hexadecimal is a number system with has sixteen digits, 0 to 9 followed by “A” to “F”.
  + Hex values always begin with # and specify values of red, blue and green using hexademical numbers such as #23F41A.
* RGB
  + RGB colors use the rgb() syntax with one value for red, one value for blue and one value for green.
  + RGB values range from 0 to 255 and look like this: rgb(7, 210, 50).
* HSL
  + HSL stands for hue (the color itself), saturation (the intensity of the color), and lightness (how light or dark a color is).
  + Hue ranges from 0 to 360 and saturation and lightness are both represented as percentages like this: hsl(200, 20%, 50%).
* You can add opacity to color in RGB and HSL by adding a fourth value, a, which is represented as a percentage.

## Typography

* Typography is the art of arranging text on a page.
* Text can appear in any number of weights, with the font-weight property.
* Text can appear in italics with the font-style property.
* The vertical spacing between lines of text can be modified with the line-height property.
* Serif fonts have extra details on the ends of each letter. Sans-Serif fonts do not.
* Fallback fonts are used when a certain font is not installed on a user’s computer.
* Google Fonts provides free fonts that can be used in an HTML file with the <link> tag or the @font-face property.
* Local fonts can be added to a document with the @font-face property and the path to the font’s source.
* The word-spacing property changes how far apart individual words are.
* The letter-spacing property changes how far apart individual letters are.
* The text-align property changes the horizontal alignment of text.
* Line Spacing



* Serif and Sans Serif:





# JavaScript

JavaScript is a powerful, flexible, and fast programming language now being used for increasingly complex web development and beyond!

## Intro

* Data is printed, or logged, to the console, a panel that displays messages, with console.log().
* We can write single-line comments with // and multi-line comments between /\* and \*/.
* There are 7 fundamental data types in JavaScript: strings, numbers, booleans, null, undefined, symbol, and object.
* Numbers are any number without quotes: 23.8879
* Strings are characters wrapped in single or double quotes: 'Sample String'
* The built-in arithmetic operators include +, -, \*, /, and %.
* Objects, including instances of data types, can have properties, stored information. The properties are denoted with a . after the name of the object, for example: 'Hello'.length.
* Objects, including instances of data types, can have methods which perform actions. Methods are called by appending the object or instance with a period, the method name, and parentheses. For example: 'hello'.toUpperCase().
* We can access properties and methods by using the ., dot operator.
* Built-in objects, including Math, are collections of methods and properties that JavaScript provides.

## Variables

* Variables hold reusable data in a program and associate it with a name.
* Variables are stored in memory.
* The var keyword is used in pre-ES6 versions of JS.
* let is the preferred way to declare a variable when it can be reassigned, and const is the preferred way to declare a variable with a constant value.
* Variables that have not been initialized store the primitive data type undefined.
* Mathematical assignment operators make it easy to calculate a new value and assign it to the same variable.
* The + operator is used to concatenate strings including string values held in variables
* In ES6, template literals use backticks ` and ${} to interpolate values into a string.
* The typeof keyword returns the data type (as a string) of a value.

## Conditionals

* An if statement checks a condition and will execute a task if that condition evaluates to true.
* if...else statements make binary decisions and execute different code blocks based on a provided condition.
* We can add more conditions using else if statements.
* Comparison operators, including <, >, <=, >=, ===, and !== can compare two values.
* The logical and operator, &&, or “and”, checks if both provided expressions are truthy.
* The logical operator ||, or “or”, checks if either provided expression is truthy.
* The bang operator, !, switches the truthiness and falsiness of a value.
* The ternary operator is shorthand to simplify concise if...else statements.
* A switch statement can be used to simplify the process of writing multiple else if statements. The break keyword stops the remaining cases from being checked and executed in a switch statement.

The list of falsy values includes:

* 0
* Empty strings like "" or ''
* null which represent when there is no value at all
* undefined which represent when a declared variable lacks a value
* NaN, or Not a Number

## Short-circuit evaluation:

Let name= username || ‘stranger’;

Here, if value of username is falsy, ‘stranger’ gets assigned to ‘name’

## Functions



* A function is a reusable block of code that groups together a sequence of statements to perform a specific task.
* A parameter is a named variable inside a function’s block which will be assigned the value of the argument passed in when the function is invoked:



* ES6 introduces new ways of handling arbitrary parameters through default parameters which allow us to assign a default value to a parameter in case no argument is passed into the function.
* To return a value from a function, we use a return statement.
* To define a function using function expressions:
* Function definition can be made concise using concise arrow notation: