WEB AUTHORING

What is a website?

- Its a collection or related webpages
- It contains three layers: structure layer (HTML), presentation layer (CSS), behaviour layer (JS).
- Hypertext Markup Language and Cascading stylesheet are two common languages used in developing static webpages.

Description of CSS Precedence

- CSS can be incorporated into a webpage in three main ways (***arranged in order of precedence***):
  - **Inline CSS**: Inline styles are applied directly to individual HTML elements using the `style` attribute. Styles defined inline override any other CSS rules.

```
<p style="color: blue;">This is a paragraph with inline CSS.</p>
```

  - **Internal CSS**: Internal styles are defined within the `<style>` element in the `<head>` section of the HTML document. These styles apply to the entire document or specific elements within it.

```
<head>
    <style>
        p {
            color: red;
        }
    </style>
</head>
```

  - **External CSS**: External stylesheets are separate CSS files linked to the HTML document using the `<link>` element. These styles apply globally across multiple HTML pages.

```
<head>
    <link rel="stylesheet" type="text/css" href="styles.css">
</head>
```

Explaining the Document Type Definition

The `<!DOCTYPE html>` declaration is an instruction to the web browser about which version of HTML the page is written in. It stands for "Document Type Definition" and is used to ensure that the browser renders the web page correctly according to the specified HTML version.

Specifically, `<!DOCTYPE html>` is used for HTML5 documents, which is the latest version of HTML. When a browser encounters this declaration at the beginning of an HTML document, it switches into

standards mode and renders the document according to the rules specified in the HTML5 standard. Using `<!DOCTYPE html>` is important because it helps ensure consistent rendering of web pages across different browsers and devices

In addition to `<!DOCTYPE html>` for HTML5 documents, there were various other Document Type Definitions (DTDs) used in earlier versions of HTML. These DTDs were used to specify the rules and structure of HTML documents. Example: HTML 4.01 Strict, XHTML 1.1 etc.

These different DTDs served to define the syntax and structure of HTML documents and helped browsers render them correctly. However, with the introduction of HTML5, which has a simplified and more flexible syntax, the need for different DTDs diminished, and `<!DOCTYPE html>` became the standard declaration for modern HTML documents.

The head tag

The heading tag often contains elements that are not displayed in the actual webpage. However, the elements placed within it still plays a vital role in ensuring proper functionality.

- The following are the universally accepted elements to be placed within the head tag:
  - **Title tag:** Displays text in the browser's toolbar, favourites menu, and search results. Here's how the `<title>` tag is typically used:

```
<!DOCTYPE html>
<html>
<head>
    <title>This is the Title of the Web Page</title>
</head>
<body>
    <!-- Page content goes here -->
</body>
</html>
```

    The text enclosed within the `<title>` tags, in this case, "This is the Title of the Web Page," will be displayed at the top of the browser window or tab when the page is opened. It's essential to provide a descriptive and concise title that accurately reflects the content of the web page, as it helps users understand the purpose of the page and improves search engine optimization (SEO).
    Search engines use the title tag to understand the topic and relevance of a webpage. A well-crafted title tag can improve a page's visibility in search engine results pages (SERPs) and attract more clicks from users.
  - **Base tag:** The `<base>` tag specifies the base URL and target for all relative URLs within a document. It's placed in the `<head>` section of an HTML document.

```
<head>
    <base href="https://example.com/" target="_blank">
</head>
```

- **Link tag:** The `<link>` tag is used to link an external resource to the HTML document, such as a stylesheet, favicon, or other resources. It's commonly used to link CSS stylesheets

```
<head>
    <link rel="stylesheet" type="text/css" href="styles.css">
</head>
```

- **Meta tag:** The `<meta>` tag provides metadata about the HTML document, including character encoding, viewport settings, description and keywords for SEO purposes.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="description" content="This is a description of the webpage.">
    <meta name="keywords" content="HTML, CSS, JavaScript, web development">
    <meta name="author" content="Your Name">
</head>
<body>
    <!-- Page content -->
</body>
</html>
```

- **Script tag:** The `<script>` tag is used to embed or reference JavaScript code within an HTML document. It can be placed in the `<head>` or `<body>` section.

```
<head>
    <script src="script.js"></script>
</head>
```

- **Style tag:** The `<style>` tag is used to define internal CSS styles within an HTML document. It's placed within the `<head>` section.

```
<head>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
        }
        /* Other CSS styles */
    </style>
</head>
```

Understanding comments

Comments in HTML are used to add explanatory notes or annotations within the code that are not displayed in the browser when the page is rendered. They are helpful for developers to document their code, provide reminders, or temporarily disable specific sections of code without deleting them entirely.

HTML comments are written using the `<!--` opening tag and the `-->` closing tag. Anything between these tags is treated as a comment and is ignored by the browser during rendering.

The body tag

The body tag contains elements whose contents will be rendered on the browser as instructed by the elements themselves. The rest of the tags described in this document must only be placed in the body tag.

Preset Styles in HTML

In HTML, there are no "preset" styles inherent to the language itself. HTML is a markup language used for structuring content, while styles are applied using CSS (Cascading Style Sheets). However, browsers do have default stylesheets, which are applied to HTML elements unless overridden by specific CSS rules.

These default styles vary slightly between different browsers, but they generally aim to ensure that web pages are displayed consistently and are readable even if no CSS styles are explicitly applied.

Some common default styles applied by browsers to HTML elements include:

- **Text styling**: Browsers typically set default font sizes, font families, line heights, and margins for text elements like `<p>`, `<h1>`, `<h2>`, `<h3>`, etc.
- **Lists**: Default styles for `<ul>` (unordered lists) and `<ol>` (ordered lists) include padding, margins, and list item markers (bullets or numbers). Each list item is added by `<li>`.
- **Links**: Default link styles typically include underlining and changing the color to indicate visited and unvisited links. However, these styles can vary depending on the browser and the state of the link.
- **Tables**: Default table styles include border spacing, padding, and border styles for `<table>`, `<tr>`, `<td>`, and related elements.
- **Form elements**: Basic styling is applied to form elements like `<input>`, `<select>`, `<textarea>`, etc., to ensure they are visually distinguishable and accessible.

Text Enhancements

The following tags can be used to enhance texts in a html document.

- `<b>` tag: This stands for "bold." It is used to make the enclosed text bold.

```
<b>This text will be bold</b>
```

- `<i>` tag: This stands for "italic." It is used to make the enclosed text italicized.

```
<i>This text will be italicized</i>
```

- `<sup>` tag: This stands for "superscript." It is used to render text in a smaller size and slightly above the normal line of text, typically used for exponents or footnotes.

```
x<sup>2</sup> represents x squared.
```

- `<sub>` tag: This stands for "subscript." It is used to render text in a smaller size and slightly below the normal line of text, often used for chemical formulas or mathematical expressions.

```
H<sub>2</sub>O represents water.
```

HTML Tables

<table> - Used to define the outline of a table. It is somewhat graphically responsible for the external grid lines. The <table> has an attribute called **border** which can be set to "0"/"" for no visible gridlines and "1" for visible gridlines. The following are tags that can be placed within <table> (**arranged in the order expected by HTML parsers**).

- <caption> - The first item within the table. Items placed within this tag is centred and appears above the table it self.
- <thead> - Stands for "Table header". Defines the first section of a table. Must be the second tag within the table
- <tfoot> - Stands for "Table footer". Defines the last section of a table. Must be the third tag within the table
- <tbody> - Stands for "Table body". Defines the second section of a table. Must be the last tag within the table.
    - <tr> - Defines a row. Can go inside <thead>, <tfoot>, <tbody> or just within the table itself.
    - <td> - Defines a **normal** cell. Goes inside <tr>. There can be any tag within <td>.
    - <th> - Defines a **header** cell. Goes inside <tr> like <td>. Header cells appear emboldened and centred. There can be any tag within <th>.

This is a caption

This is a heading cell in the header section.	
This is a row heading using heading cell.	This is a normal cell.
This is a row heading using heading cell.	This is a normal cell

The General form of HTML Attributes and CSS Style Rules

In HTML, attributes provide additional information about an element and are typically defined within the opening tag of an element. The general form of an **HTML attribute** is*:*

The general form of HTML attributes

attribute	=	"value"

In CSS, **style rules** consist of one or more selectors followed by a block of declarations enclosed in curly braces. The general form of a style rule is:

The general form of style rules

selector	*CSS Combinators (' ', + >)*	{	property	:	"value"	; (Seperator)	*Other Declarations*	}

The Width and Height of a Table

The width of a column can be adjusted by changing exactly the width one cell in the required column.

Example:

```
<tr>
    <th colspan="2" style="width: 1000px;"> <!-- Take up two coloumns -->
        This is a heading cell in the header section.
    </th>
</tr>
```

This is a caption

This is a heading cell in the header section.	
This is a row heading using heading cell.	This is a normal cell.
This is a row heading using heading cell.	This is a normal cell

The height of the rows needs to be changed manually for each row. Changing the height of a single row only affects it, and not any other row in the table.

Example:

```
<tfoot>
    <tr>
        <th style="height: 100px;">
            This is a row heading using heading cell.
        </th>
        <td>
            This is a normal cell
        </td>
    </tr>
</tfoot>
```

This is a caption

This is a heading cell in the header section.	
This is a row heading using heading cell.	This is a normal cell.
This is a row heading using heading cell.	This is a normal cell

Note: The column width **has to be** modified in the `<td>` or `<th>` elements; changing the width of the `<tr>` (row) **will not** affect the width of the column -- in fact, it has no effect in this context. On the other hand, the height of the row **can be** changed by either adjusting the height of a cell (`<td>` or `<th>`) in the row or by modifying the height of the row itself (`<tr>`).

Centre Alignment using Margins

Setting the left and right margins of an element to "auto" in HTML/CSS is a common technique used to horizontally center that element within its containing parent element. When applied to a block-level element like a table, setting both margins to "auto" effectively distributes the remaining space equally on both sides, thus centering the table horizontally within its container.

Here's how it works:

- **Setting both margins to "auto"**: By setting both the left and right margins to "auto", you're telling the browser to calculate equal margins on both sides of the table, effectively centering it within its containing element.
- **Block-level element behavior**: Tables are block-level elements by default, which means they occupy the full width of their containing parent element unless specified otherwise.
- **Automatic margin calculation**: When you set a margin to "auto", the browser automatically calculates the margin value to distribute the available space equally. In the case of setting both left and right margins to "auto", the browser calculates and applies equal margins on both sides, thus centering the table horizontally.

This technique is not specific to tables; it can be applied to other block-level elements like divs, paragraphs, and headings to center them horizontally within their containing elements as well.

Collapsing Borders in tables

In HTML tables, the `<tables>` contain borders called **table borders**, and the cell contains its own border called **cell border**. However, conventional tables have a single **external border** and **internal gridlines.** To give an HTML table the conventional table look, you should use the `border-collapse` property in CSS.

Adding this `<table border="1" style="border-collapse: collapse;">` changes the table from the left to the right

This is a caption

This is a heading cell in the header section.	
This is a row heading using heading cell.	This is a normal cell.
This is a row heading using heading cell.	This is a normal cell

This is a caption

This is a heading cell in the header section.	
This is a row heading using heading cell.	This is a normal cell.
This is a row heading using heading cell.	This is a normal cell

Styling Cell and Table borders

Styling to cell borders must be added to the <td> or <th> tags. And styling to the table borders must be added to <table> tag. Adding styling to the row (<tr>), aiming to style the cells within the rows, will be futile. It may be important to note that, adding styling to a <td> tag only modifies the border around that tag. To style the borders for all the cells within a table, all the <td> and <th> tags within the table must be individually styles, either by redundantly styling the individual cells using **inline CSS,** or by using an **external stylesheet**.

Text Alignment in Table cells

Text can be aligned in two ways – horizontally and vertically.

- To **horizontally align** text, use the `text-align` property.
- To **vertically align** text, use the `vertical-align` property.

Merging HTML cells

Like text alignment, html cells can be merged in two ways – horizontally (merging columns) vertically (merging rows).

- To merge **rows,** use the `rowspan` attribute. The result will be similar to the picture below.

- To merge **columns,** use the `colspan` attribute. The result will be similar to the picture below.

Understanding Padding and Border-spacing in HTML

In HTML, "padding" is a term used to describe the space between the content of an element and its border. It allows you to control the amount of space between the content and the border of an element. Padding is commonly used to improve the visual appearance of elements by adding space around them.

Padding can be applied to various HTML elements such as `<div>`, `<p>`, `<span>`, etc. It is typically defined using CSS (Cascading Style Sheets), either inline or in an external stylesheet

On the other hand, `border-spacing` is a CSS property used only with the `<table>` element in HTML. `border-spacing` adds spacing between the borders of adjacent cells within a table. In addition, it also adds spacing between the cells and the outer border of the table itself.

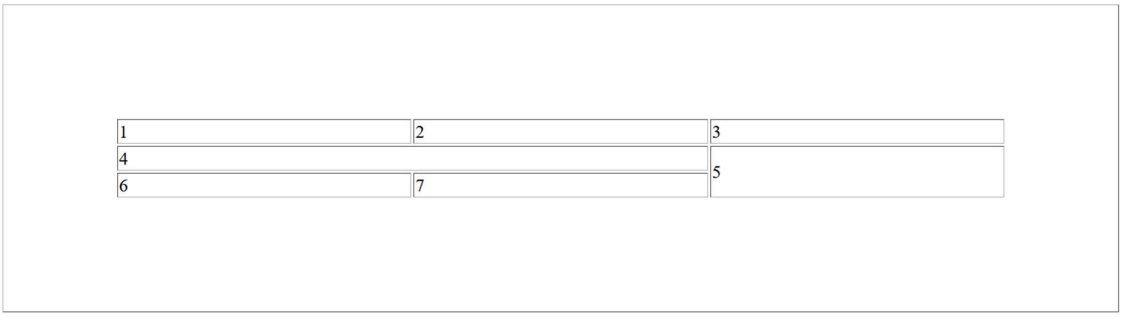
Note: Both `border-spacing` and `padding` **require** `border-collapse` to be set to `separate`. If this requirement is not met, `border-spacing` and `padding` will have no effect on the table.

Despite the similarities in the ability of the **selectors** the effect of one cannot effectively be recreated by the other.

1. For example, when styled for **padding** like this:

```
<table style="border-collapse: separate; width: 1000px; padding: 100px" border="1">
```

The result is:



2. And, when styled for **border-spacing** like this:

```
<table style="border-collapse: separate; width: 1000px; border-spacing: 100px 10px" border="1">
```

The result is:

Horizontal Rule and Line Break

Horizontal Rule: The `<hr>` element is used to create a thematic break or horizontal rule in the content. It typically renders as a horizontal line or rule across the width of the containing element.

```
<p>This is paragraph one.</p>
<hr>
<p>This is paragraph two.</p>
```

Line Break: The
 element is used to insert a line break or newline within text content. It forces the text or content following it to start on a new line. It's useful for creating line breaks within paragraphs, addresses, or other text where you want to control the line breaks manually.

```
<p>This is the first line.<br>This is the second line.</p>
```

Graphics within HTML

Images: Images can be presented in HTML using the <img> tag. The <img> tag takes a **mandatory** src attribute and the **optional** alt, width, and height attributes.

Video: Videos in HTML can be presented using two tags - <video> and <source>. The <video> tag takes in the **mandatory** width and height attributes as well as the **optional** autoplay and control attributes. Within the <video> tag, you must place the <source> tag as well as a text expecting the playback failures that arise due to incompatible HTML version or video formats. The <source> tag takes the **mandatory** src and type attributes.

Audio: Audio in HTML is rather similar to video in HTML. It is presented using the <audio> tag which takes the **optional** attributes autoplay, controls, width and height similar to the <video> tag. In addition, it also utilizes the <source> tag, which takes the **mandatory** src and type attributes. A text expecting playback failures is also placed within the <source> tag for audio, just as it is for the <video> tag.

Example:

```
<center><img src="graphics_and_audio/giphy.gif" alt="Philosopher Emoji Gif" style="width: 10%"></center>

<hr>

<center>
<video width="40%" controls autoplay muted loop>
    Too bad! Your browser cannot play hello.mp4
    <source src="graphics_and_audio/hello.mp4" type="video/mp4">
</video>
</center>

<hr>
<center>
<audio controls autoplay loop>
    Too bad! Your browser sucks!
    <source src="graphics_and_audio/8bit.mp3" type="audio/mp3">
</audio>
<center>
```

<img>, <audio> and <video> tags take more attributes than described in the paragraphs above. The picture shows a few of these in action. In particular, loop and muted attributes are especially useful. Additionally, ***the*** *autoplay* ***attribute requires the*** *muted* ***attribute*** in most modern browsers to function as most users despise uncontrolled automatic audio playback.

Colours – A Brief Introduction

Colours can only be defined in CSS. And in CSS, colours are generally defined as hexadecimal. Hexadecimal is a base-16 number system denoted by the characters 0 to F. In CSS, hexadecimal integers must be preceded by a hashtag ('#').

The general form of defining colours in CSS is as follows:

	Red	Green	Blue
#	00 - FF	00 - FF	00 - FF

Example: White in hex would be #`FFFFFF`

Note: In CSS, the background colour property can be used to shade cells (<`td`>, <`th`>), table rows (<`tr`>), table sections (<`thead`>, <`tfoot`>, <`tbody`>), and the entire body section (<`body`>) of the web page itself!

Lists a summary

Lists in HTML can be ordered using the <`ol`> tag, or unordered using the <`ul`> tag. Each item in a list must be enclosed within the list item <`li`> tag in order to inherit its respective styling, which is **numbers** for ordered lists, and **bullets** for unordered lists. If items within the <`ol`> or <`ul`> tags are not placed within <`li`> tag, they will still be indented appropriately. However, they won't be styled.

Hyperlinks in Detail

Hyperlinks in HTML is achieved by using the anchor <a> tags along with the hyperlink reference (`href`) attributes. You can use text and images for hyperlinks.

The hyperlink reference attribute can also be used for emails. The following properties can be used in the hyperlink reference string:

1. ***URL Scheme (mailto):***
    - *This specifies that the link is intended to open the user's default email client to send an email.*
2. ***Query Properties:***
    - `to` (Required): Specifies the recipient's email address. Multiple recipients can be separated by commas.
    - `cc` (Optional): Specifies the email addresses of recipients who will receive a copy (carbon copy) of the email. Multiple recipients can be separated by commas.
    - `bcc` (Optional): Specifies the email addresses of recipients who will receive a blind carbon copy of the email (i.e., their addresses are not visible to other recipients). Multiple recipients can be separated by commas.
    - `subject` (Optional): Specifies the subject of the email.
    - `body` (Optional): Specifies the body of the email.

3. ***Special Characters:***

    - `%20`: Represents a space character in URLs.
    - `&`: Separates different query parameters.
    - `?`: Indicates the beginning of the query string. A query string is a part of a URL that is used to pass data to web applications via parameters. These parameters are in the form of key-value pairs and are separated by '&' symbols.

Example:

```
<a
href="mailto:recipient@example.com?cc=cc1@example.com,cc2@example.com&bcc=b
cc@example.com&subject=Subject%20Line&body=Hello%20World">Send Email</a>
```

Target Windows

Target windows to open hyperlinks can be set in the base `(<base>)` tag or anchor `(<a>)` tag using the `target` attribute.

The following are the most common values for the target attribute:

- `_blank`: Open the hyperlink in a new window
- `_self`: Open the hyperlink in the current window
- Additionally, any string can be succeeded by the underscore character and be used as a value for the target attribute. In this case, the browser will check if a window by the specified string exists, if it does it opens the link there, otherwise, the browser creates the window and opens the link in there.

Background Colour

In HTML, the `background-color` property (which is a CSS property) can be applied to the body of the webpage (<`body`>), entire tables (<`table`>), table sections (<`thead`>, <`tfoot`>, <`tbody`>), table rows (<`tr`>), and individual table cells (<`td`> and <`th`>).

Note: If any section in a table i.e table header, table footer or table body contains only one table row. Then, applying the `background-color` property to the table section is equivalent to applying it to the only table row within the section.

Background Images

The `background-image` CSS property can be applied to the body section of the webpage. In modern browsers, more than one image can be applied as background at an instance for a webpage. The background-image property can be applied to a other styles than just the body.

The following CSS properties can be used to provide additional information on how to present the background:

- `background-image`: Takes the image as returned by the `url()` function.
- `background-size`**:** The `width` and `height` properties do not affect the dimensions of a background image. Thus, this property must be used in order to resize a background image.

- `background-repeat`: Allows the image to be displayed only once (`no-repeat`), or repeated (`repeat`) for a specified number of times as defined by `repeat-x` and `repeat-y`.
- `background-attachment`: The `background-attachment` property in CSS specifies whether a background image should scroll with the rest of the page's content or remain fixed in place as the page is scrolled. It can take one of the following values:
    - `scroll`: This is the default value. It means that the background image will scroll along with the content of the element.
    - `fixed`: With this value, the background image will remain fixed in place relative to the viewport. This means that as the user scrolls the page, the background image will stay in the same position.
    - `local`: This value is less commonly used and it's not widely supported. It means that the background image will scroll along with the content of the element, but not the entire page.
- `background-position`: The `background-position` property in CSS allows you to specify the starting position of a background image within its containing element. It determines where the background image will be placed relative to the top-left corner of the element. The position of the image can be provided by the default keywords, pixel values, or percentage values. Additionally, more than one value can be provided to the property to provide both the horizontal and vertical positions separately.

Example:

```
/* Set the background image */
background-image: url('example.jpg');

/* Resize the background image */
background-size: cover;

/* Do not repeat the background image */
background-repeat: no-repeat;

/* Fix the background image in place relative to the viewport */
background-attachment: fixed;

/* Position the background image */
background-position: center center;
```

Fonts Sizes

Font sizes can be defined in CSS as **absolute** or **relative** values.

Absolute Values: Font sizes in CSS can be defined in the following absolute value measurements:

- **Centimetre** `(cm)` **and Millimetre** `(mm)` **:** Font sizes can be defined in centimetres and millimetres.

- **Inch** `(in)` **:** Used to defines the font sizes in inch. One inch is approximately 2.54cm.
- **Points** `(pt)` **:** Used to define font sizes in points. There are 72 points to an inch.
- **Picas** `(pc)` **:** Font sizes can be defined in picas. One pica is about 12 points.
- **Pixels** `(px)` **:** This is the only measurement whose apparent size is entirely dependent on the ratio of pixel densities of different monitors. Thus, they may look different on every monitor.

Relative Values: Font sizes in CSS can also be defined as relative values (sometimes to the size of a previously specified absolute size).

- **em:** In CSS, the `em` unit is a relative unit of measurement that is based on the computed value of the font size of an element's parent. Specifically, it represents the calculated font size of the element itself. The `em` unit is commonly used for setting sizes of various elements in a layout, such as margins, padding, widths, heights, and font sizes.
  When you specify a size using `em`, it's relative to the font size of the element's nearest parent element that has a defined font size. If no parent element has a specified font size, the `em` unit will be relative to the default font size of the browser, which is typically 16 pixels.
  For example, if the font size of a paragraph (`<p>`) element is set to `1.2em`, it means the font size will be 1.2 times the font size of its parent element. If the parent element's font size is 16 pixels, the paragraph's font size will be 19.2 pixels (1.2 * 16 = 19.2 pixels).

  Note: A **parent** element, is an element that has other elements nested within it. The elements that are nested one level deeper from the parent is called a **child.** The elements that are nested any level deeper from the parents called **descendents.**
- **ex:** In CSS, the `ex`-unit is a relative unit of measurement that represents the height of the lowercase letter 'x' in the ***current font***. It's primarily used for setting sizes relative to the font's x-height, which is the height of lowercase letters in a font. The x-height can vary between different fonts, but using `ex` as a unit ensures that the size remains proportional to the font's design. For example, if you set the height of an element to `2ex`, it will be twice the height of the lowercase 'x' in the current font.
  In the context of ex measurement, the ***current font***, is the font that is applied to the element where you're using the `ex` unit. The `ex` unit measures the height of the lowercase letter 'x' in the font currently applied to that element. This font size is inherited from the parent element unless explicitly overridden.
  So, if you have an element with a font size specified in `ex`, it will be relative to the font size of its parent element. If the parent element also has a font size specified in `ex`, then the size will be relative to the font size of its parent element, and so on, following the chain of parent elements up to the root of the document.
  In other words, unlike most measurements, the ex-measurement is relative to the size of the letter ‘x’ in the current font-family, and is not relative to any other absolute font size.
- **Percentages:** When percentage values are used for font sizes in CSS, they are relative to the font size of the parent element, not the containing block.
- **larger and smaller:** In CSS, the keywords `larger` and `smaller` are used to set font sizes relative to the font size of the parent element. These keywords can be used as values for the `font-size` property.

When you set the font size of an element to `larger`, it increases the font size relative to the font size of its parent element. Conversely, when you set the font size to `smaller`, it decreases the font size relative to the font size of the parent element.

- **xx-small to xx-large**: In CSS, the xx-small to xx-large keywords are used to specify font sizes relative to the browser's default font size. These keywords provide a way to set font sizes at different levels of scale, from very small to very large, without needing to specify exact pixel or percentage values.

  Here's an overview of each of these keywords:

  - `xx-small`: This sets the font size to the smallest available size. It's typically smaller than the browser's default font size.
  - `x-small`: This sets the font size to a size slightly larger than xx-small.
  - `small`: This sets the font size to a size slightly larger than x-small.
  - `medium`: This sets the font size to the browser's default font size.
  - `large`: This sets the font size to a size slightly larger than medium.
  - `x-large`: This sets the font size to a size slightly larger than large.
  - `xx-large`: This sets the font size to the largest available size. It's typically larger than the browser's default font size.

CSS Properties for text

The following are CSS properties that can be applied to text such as `<p>` and `<h1> ... <h6>`. They have been grouped below for convenience.

The font properties

- `font-family`: Defines the font family to use.
- `font-size`: Defines the size of a text using absolute or relative measurements.
- `font-weight`: Defines the extent to embolden a text.
- `font-style`: Defines how the text will be italicized.
- `font-variation`: Can be used to display text in small capitals, where lowercase letters are replaced with smaller uppercase letters

The text properties

- `text-align`: Defines the horizontal alignment of a text
- `text-decoration`: Defines if the text has to be underlined, overlined or struck-through.

The colour properties

- `color`: Defines the font colour.
- `background-color`: Defines the shading of the text.

The vertical property

- `vertical-alignment`: Defines how a text must be aligned vertically.

Some text enhancements such as superscript and subscript can only be defined in the HTML. Others can such as emboldening and italicizing can be done both in HTML and CSS.

Note: The `inherit` value for any CSS property instructs the browser to inherit the computed value of that property from its parent element. It means that the property will take on the same value as its parent element, effectively inheriting its styling. When you apply `inherit` to a property, it overrides any value set directly on the element itself, causing it to adopt the value of the same property from its parent element.

Attribute Selectors in CSS

You can select elements based on their attributes using attribute selectors in CSS. Attribute selectors allow you to target elements based on the presence or value of HTML attributes.

There are several types of attribute selectors:

- **Exact Attribute Selector**: Selects elements that have a specified attribute with a specific value. It uses the syntax `[attribute=value]`
- **Substring Attribute Selector**: Selects elements that have a specified attribute with a value containing a specific substring. It uses the syntax `[attribute*=value]`
- **Prefix Attribute Selector**: Selects elements that have a specified attribute with a value starting with a specific string. It uses the syntax `[attribute^=value]`
- **Suffix Attribute Selector**: Selects elements that have a specified attribute with a value ending with a specific string. It uses the syntax `[attribute$=value]`
- **Exact Attribute Selector with Case Insensitivity**: Selects elements that have a specified attribute with a value matching a specified string, regardless of case. It uses the syntax `[attribute=value i]`.

Attribute selectors can be preceded by **element selectors** to select elements that meet the condition defined by the attribute selector.

Pseudo-classes in CSS

In CSS, a pseudo-class is a keyword added to a selector that specifies a special state or condition of the selected elements. Pseudo-classes allow you to style elements based on factors other than their name or attributes. In CSS, when you use a space between a selector and a colon (`:`), it indicates that you are targeting a pseudo-class of the elements matched by the selector.

Some common pseudo-classes include:

- `:hover`: Styles an element when it's being hovered over by the mouse cursor.
- `:active`: Styles an element when it's being activated, typically when it's being clicked on.
- `:focus`: Styles an element when it has received focus, such as when it's selected by tabbing through form elements.
- `:visited`: Styles links that have been visited by the user.
- `:first-child`: Styles an element if it's the first child of its parent element.
- `:last-child`: Styles an element if it's the last child of its parent element.
- `:nth-child(n)`: Styles elements based on their position in the document tree, where `n` is a formula or keyword.

Combinators in CSS

Combinators are special characters used to specify relationships between selectors in CSS.

```
<style>

    table td /* Select all the <td> tags that are present within the <table> tags */
    {
        border-color: red;
    }

    table, td /* Select all the <td> and <table> tags */ {
        border-style: dotted;
    }

</style>
```

The following are a general descriptions of CSS combinators:

- **Descendant combinator**: Selects all elements that are descendants of a specified ancestor. Selects all elements that are descendants of a specified ancestor element, regardless of how deeply nested they are.

```
ancestor descendant {
    /* CSS rules */
}
```

- **Child combinator**: Selects all direct children of the specified parent. It targets elements that are immediately nested within the parent, excluding any deeper descendants.

```
parent > child {
    /* CSS rules */
}
```

- **Adjacent sibling combinator**: Selects an element that is directly preceded by a sibling element. The following rule selects the `adjacent` element that immediately follows `element`.

```
element + adjacent {
    /* CSS rules */
}
```

- **General sibling combinator**: Selects all sibling elements that follow a specified element. It targets all elements that come after the specified element, not just the immediately following one unlike the **Adjacent sibling combinator.** The following selects all `sibling` elements that come after `element` in the document structure.

```
element ~ sibling {
    /* CSS rules */
}
```

- **Grouping combinator:** In CSS, the grouping combinator allows you to apply the same set of styles to multiple selectors. It's denoted by a comma (,), and it combines multiple selectors into a single rule set, allowing you to apply the same styles to all the selectors listed.

```
selector1, selector2, selector3 {
    /* Styles shared by all selectors */
}
```

DOM (Document Object Model) Traversal

The following concepts in the Document Object Model (DOM) are known as DOM Navigation or DOM Traversal. They refer to properties used to navigate through the structure of an HTML or XML document.

- **Ancestor**: Ancestor elements are those that are higher up in the document tree relative to a specific element. They are the elements that contain the element in question. Analogously, you can think of them as grandparents, parents, or other ancestors in a family tree.
- **Parent**: The parent element is the immediate container of a specific element. It's directly above the element in the document tree. This is similar to a parent in a family tree, who has children.
- **Child**: A child element is directly contained within another element. It's immediately below the parent element in the document tree. Analogously, children in a family tree are offspring of their parents.
- **Descendant**: A descendant element is any element that is contained within another element, regardless of its depth in the document tree. It includes children, grandchildren, and all subsequent generations. In a family tree analogy, descendants are all the offspring of an ancestor, no matter how many generations removed.
- **Sibling**: Sibling elements are elements that share the same parent. They are on the same level in the document tree and are adjacent to each other. Analogously, siblings in a family tree share the same parents.
- **The root node**: It represents the top-level container for all other nodes in the document. In an HTML document, the root node is the `<html>` element.
  The root node is unique because it serves as the starting point from which all other nodes in the document are descended. It's the ancestor of all other nodes, but it doesn't have an ancestor itself. Analogously, this must be Adam.