**Introduction to CSS and Styling Basics**

**Theory assignments;**

**1 Explain the difference between inline, internal, and external CSS.**

**1. Inline CSS**

* **Definition**: CSS is applied directly within the HTML element using the style attribute.
* **Syntax Example**:

html

<p style="color: blue; font-size: 16px;">This is inline CSS.</p>

* **Use Case**: Quick, one-off styles for specific elements.
* **Pros**:
  + Easy to apply for small changes.
  + Overrides internal and external styles (due to higher specificity).
* **Cons**:
  + Hard to maintain.
  + Reduces separation of content and presentation.
  + Not reusable.

**2. Internal CSS**

* **Definition**: CSS is written within a <style> tag in the <head> section of the HTML document.
* **Syntax Example**:

html

<head>

<style>

p {

color: green;

font-size: 18px;

}

</style>

</head>

<body>

<p>This is internal CSS.</p>

</body>

* **Use Case**: Styling a single HTML document.
* **Pros**:
  + Keeps CSS in one place for that document.
  + Easier to manage than inline if multiple elements are styled.
* **Cons**:
  + Still mixes content and style.
  + Not reusable across multiple pages.

**3. External CSS**

* **Definition**: CSS is written in a separate .css file and linked to the HTML using the <link> tag.
* **Syntax Example**:

html

<head>

<link rel="stylesheet" href="styles.css">

</head>

**In styles.css**:

css

p {

color: red;

font-size: 20px;

}

* **Use Case**: Styling multiple HTML documents with the same stylesheet.
* **Pros**:
  + Promotes reusability and cleaner code.
  + Great for site-wide consistency.
  + Easier to maintain and update.
* **Cons**:
  + Requires an additional HTTP request (though this can be cached).

**Summary Table:**

| **Feature** | **Inline CSS** | **Internal CSS** | **External CSS** |
| --- | --- | --- | --- |
| Location | Inside HTML element | <style> in <head> | Separate .css file |
| Scope | One element | One document | Multiple documents |
| Reusability | None | Low | High |
| Maintenance | Hard | Moderate | Easy |
| Best For | Quick fixes/testing | Small single-page sites | Large/multi-page sites |

**2. Describe CSS selectors and list the types of selectors (e.g., element, class, id).**

**Common Types of CSS Selectors**

Here are the main types of CSS selectors with examples:

**1. Element Selector**

* **Targets** all instances of a specific HTML tag.
* **Syntax**:

css

p {

color: blue;

}

* **Example**: Styles all <p> elements.

**2. Class Selector**

* **Targets** elements with a specific class attribute.
* **Syntax**:

css

.highlight {

background-color: yellow;

}

* **Example**:

html

<div class="highlight">This will be highlighted.</div>

**3. ID Selector**

* **Targets** a single element with a specific id.
* **Syntax**:

css

#header {

font-size: 24px;

}

* **Example**:

html

<h1 id="header">Main Title</h1>

**4. Universal Selector**

* **Targets** all elements on the page.
* **Syntax**:

css

\* {

margin: 0;

padding: 0;

}

**5. Group Selector**

* **Targets** multiple selectors at once.
* **Syntax**:

css

h1, h2, p {

color: gray;

}

**6. Descendant Selector**

* **Targets** elements nested inside another element.
* **Syntax**:

css

div p {

font-style: italic;

}

* **Example**: Styles <p> elements **only inside** <div> tags.

**7. Child Selector**

* **Targets** direct child elements.
* **Syntax**:

css

ul > li {

list-style-type: square;

}

**8. Adjacent Sibling Selector**

* **Targets** an element immediately following another.
* **Syntax**:

css

h1 + p {

margin-top: 0;

}

**9. Attribute Selector**

* **Targets** elements based on attribute presence or value.
* **Syntax**:

css

input[type="text"] {

border: 1px solid #ccc;

}

**10. Pseudo-class Selector**

* **Targets** elements in a specific state.
* **Syntax**:

css

a:hover {

color: red;

}

**11. Pseudo-element Selector**

* **Targets** specific parts of elements.
* **Syntax**:

css

p::first-line {

font-weight: bold;

}

**Summary Table:**

| **Selector Type** | **Syntax Example** | **Targets** |
| --- | --- | --- |
| Element | p | All <p> tags |
| Class | .box | All elements with class="box" |
| ID | #main | The element with id="main" |
| Universal | \* | All elements |
| Group | h1, p | Both <h1> and <p> elements |
| Descendant | div p | <p> inside <div> |
| Child | ul > li | Direct <li> children of <ul> |
| Adjacent Sibling | h2 + p | First <p> after <h2> |
| Attribute | input[type="text"] | <input> elements with type="text" |
| Pseudo-class | a:hover | <a> when hovered |
| Pseudo-element | p::first-line | First line of <p> |

Let me know if you want a visual example using some of these!

**3. Discuss the CSS box model and its components.**

**CSS Box Model Explained**

**The CSS Box Model is a fundamental concept in web design and layout. Every HTML element on a web page is considered a rectangular box, and the box model describes how its size and spacing are calculated.**

**Components of the Box Model**

**Each element box has four main parts, from the innermost to outermost:**

**1. Content**

* **This is the actual content of the element, such as text, images, or other media.**
* **Size: Controlled with width and height.**

**2. Padding**

* **The space between the content and the border.**
* **Transparent and pushes the border outward.**
* **Set with: padding, padding-top, padding-right, etc.**

**3. Border**

* **A visible line around the padding and content.**
* **Set with: border, border-width, border-style, border-color.**

**4. Margin**

* **The space outside the border, separating the element from others.**
* **Set with: margin, margin-top, margin-left, etc.**
* **Can collapse vertically between block elements.**

**📐 Visual Representation**

**lua**

**+-------------------------------+**

**| Margin |**

**| +-------------------------+ |**

**| | Border | |**

**| | +-------------------+ | |**

**| | | Padding | | |**

**| | | +-------------+ | | |**

**| | | | Content | | | |**

**| | | +-------------+ | | |**

**| | +-------------------+ | |**

**| +-------------------------+ |**

**+-------------------------------+**

**Total Size Calculation (Default Box Model)**

**By default, the total size of an element is:**

**sql**

**Total Width = content width + padding (left + right) + border (left + right)**

**Total Height = content height + padding (top + bottom) + border (top + bottom)**

**Note: Margin is *outside* this box and is not included in the element’s actual dimensions.**

**Box-Sizing Property**

**You can change how the box is calculated using box-sizing:**

* **content-box (default): Total size does not include padding or border.**
* **border-box: Padding and border are included in the width and height.**

**css**

**/\* Recommended for easier layout control \*/**

**\* {**

**box-sizing: border-box;**

**}**

**✅ Summary Table**

| **Component** | **Description** | **Affects Size** | **Affects Spacing** |
| --- | --- | --- | --- |
| **Content** | **Actual element data (text, image, etc.)** | **✅** | **❌** |
| **Padding** | **Space inside the border** | **✅** | **✅** |
| **Border** | **Line around padding/content** | **✅** | **✅** |
| **Margin** | **Space outside the border** | **❌** | **✅** |