CSS Selectors

CSS selectors are patterns used to select and style elements on a web page. They are a fundamental part of CSS, allowing you to target specific HTML elements and apply styles to them. We can divide CSS selectors into five categories:

- Simple selectors
- CSS Combinators
- Pseudo-classes
- Pseudo-elements
- Attribute Selectors

1. **Simple Selectors:**

Simple selectors target elements based on their type, class, ID, or universal scope.

- **Type Selector:** Targets all elements of a specific type.
  - **Syntax:** element_name
  - **Example:** p { color: blue; }
    - This targets all <p> elements and sets their text color to blue.
- **Class Selector:** Targets elements based on their class attribute.
  - **Syntax:** .class_name
  - **Example:** .header { font-size: 20px; }
    - This targets all elements with the class header and sets their font size to 20px.
- **ID Selector:** Targets a single element based on its id attribute.
  - **Syntax:** #id_name
  - **Example:** #main { background-color: yellow; }
    - This targets the element with the id of main and sets its background color to yellow.
- **Universal Selector:** Targets all elements on the page.
  - **Syntax:** *
  - **Example:** * selects all elements

EXAMPLE

HTML

<!DOCTYPE html>

<html>

```
<head>
  <title>Simple Selectors</title>
  <style>
    p {
      color: blue;
    }
    #special {
      font-weight: bold;
    }
    .highlight {
      background-color: yellow;
    }
  </style>
</head>
<body>
  <p>This is a paragraph.</p>
  <p id="special">This is a special paragraph.</p>
  <div class="highlight">This is a highlighted div.</div>
</body>
</html>
```

2. CSS Combinators

Combinators are used to define relationships between selectors.

- **Descendant Selector (space):** Targets elements that are descendants of a specified element.

    - **Syntax:** parent selector child selector
    - **Example:** div p { color: green; }
        - This targets all <p> elements that are inside any <div>.
- **Child Selector (>):** Targets elements that are direct children of a specified element.
    - **Syntax:** parent > child
    - **Example:** ul > li { list-style-type: none; }
        - This targets <li> elements that are direct children of a <ul> and removes their bullet points.
- **Adjacent Sibling Selector (+):** Targets an element that is immediately preceded by a specified element.
    - **Syntax:** element1 + element2
    - **Example:** h1 + p { margin-top: 0; }
        - This targets the first <p> element that follows an <h1> and removes its top margin.
- **General Sibling Selector (~):** Targets all elements that are siblings of a specified element.
    - **Syntax:** element1 ~ element2
    - **Example:** h2 ~ p { color: grey; }
        - This targets all <p> elements that are siblings of an <h2> and sets their text color to grey.

Example

HTML

```
<div>

  <p>Paragraph 1</p>

  <div>Child div</div>

  <p>Paragraph 2</p>

  <span>Span element</span>

</div>
```

CSS

```
/* Descendant combinator */

div p {
  color: blue;
}

/* Child combinator */
div > p {
  font-weight: bold;
}

/* Adjacent sibling combinator */
p + div {
  background-color: yellow;
}

/* General sibling combinator */
p ~ span {
  text-decoration: underline;
}
```

3. Pseudo-Classes

Pseudo-classes target elements based on their state or position.

- **Syntax:** 'selector:pseudo-class'
- **Example:'** a:hover { color: red; }'
  - This targets links (<a>) when they are hovered over and changes their color to red.
- **Common Pseudo-Classes:**

1.Link Pseudo-Classes:

- :link: Applies to links that haven't been visited.
- :visited: Applies to links that have been visited.
- :hover: Applies when the mouse pointer is over the link.
- :active: Applies while the link is being clicked.

2.Input Pseudo-Classes:

- :enabled: Applies to enabled form elements.
- :disabled: Applies to disabled form elements.
- :checked: Applies to checked radio buttons and checkboxes.
- :focus: Applies when an element has focus.

3. **Structural Pseudo-Classes:**

- :first-child: Applies to the first child of its parent.
- :last-child: Applies to the last child of its parent.
- :nth-child(n): Applies to the nth child of its parent.
- :not(selector): Applies to elements that don't match the specified selector.

4. **Dynamic Pseudo-Classes:**

- :hover: Applies when the mouse pointer is over an element.
- :active: Applies while an element is being activated (e.g., clicked).
- :focus: Applies when an element has focus.

Example

HTML

```
<a href="https://example.com">Visit Example</a>

<input type="checkbox">
```

Css

```
a:link {

  color: blue;

}



a:visited {
  color: purple;
}

a:hover {
  text-decoration: underline;
}

a:active {
  color: red;
}

input:checked + label {
  text-decoration: line-through;
}
```

4. Pseudo-Elements

Pseudo-elements target specific parts of an element.

- **Syntax:** `selector::pseudo-element`
- **Example:** `p::first-letter { font-size: 2em; }`
  - This targets the first letter of every `<p>` element and makes it larger.
- **Common Pseudo-Elements:**

🎬 **::before and ::after:**

- Insert generated content before or after the content of the element.
- Syntax: selector::before { content: "text"; }
- Example: Create a bullet point before each list item.

🎬 **::first-line:**

- Styles the first line of an element.
- Syntax: selector::first-line { font-weight: bold; }
- Example: Make the first line of a paragraph bold.

🎬 **::first-letter:**

- Styles the first letter of an element.
- Syntax: selector::first-letter { font-size: 2em; }
- Example: Enlarge the first letter of a paragraph.

🎬 **::selection:**

- Styles the part of an element that is selected by the user.
- Syntax: selector::selection { background-color: yellow; }
- Example: Change the background color of selected text.

EXAMPLE

HTML

```
<p>This is a paragraph.</p>

<ul>

  <li>List item 1</li>

  <li>List item 2</li>

</ul>
```

CSS

```
p::first-line {

  font-weight: bold;

}



ul li::before {

  content: "* ";

}
```

5. Attribute Selectors

Attribute selectors target elements based on their attributes and values.

- **Syntax:** element[attribute], element[attribute="value"]
- **Example:** input[type="text"] { border: 1px solid #ccc; }
   - This targets all <input> elements with a type attribute of text and gives them a border.

Types of Attribute Selectors

1.Exact Match:

- Selects elements where the attribute value is exactly equal to the specified value.
- Syntax: element[attribute="value"]
- Example: a[href="https://example.com"] selects all anchor elements with the href attribute set to "https://example.com".

2.Partial Match:

- Selects elements where the attribute value contains the specified value.
- Syntax: element[attribute*="value"]
- Example: img[alt*="cat"] selects all image elements with an alt attribute containing the word "cat".

3.Prefix Match:

- Selects elements where the attribute value starts with the specified value.
- Syntax: element[attribute^="value"]
- Example: a[href^="https://"] selects all anchor elements with href attributes starting with "https://".

4.Suffix Match:

- Selects elements where the attribute value ends with the specified value.
- Syntax: element[attribute$="value"]
- Example: img[src$=".jpg"] selects all image elements with src attributes ending with ".jpg".

5.Attribute Existence:

- Selects elements that have a specific attribute, regardless of its value.
- Syntax: element[attribute]

- Example: input[required] selects all input elements with the required attribute.

Example

HTML

```
<img src="cat.jpg" alt="Cute cat">

<a href="https://example.com">Visit Example</a>

<input type="text" required>
```

CSS

```
img[alt*="cat"] {

  border: 1px solid red;

}


a[href^="https://"] {

  color: blue;

}


input[required] {

  border: 1px solid green;

}
```