CSS Selectors in Web Automation

To effectively understand **CSS Selectors** and their use in locating elements on a webpage, it's important to first understand what **CSS** is and its role in web development.

What is CSS?

- **CSS** stands for **Cascading Style Sheets**.
- It is used to style webpages, making them look good and easy to read.
- With CSS, we can control things like colors, fonts, layouts, and more.
- It helps create a visually appealing design for websites.

Now that we know what CSS is, let's learn how CSS Selectors help us find elements on a webpage when we automate testing.

What is a CSS Selector?

A **CSS Selector** in **Selenium WebDriver** is a tool used to find elements on a webpage. Here's why it's useful:

- CSS Selectors allow us to combine different attributes of an element to create a more accurate locator.
- Instead of using just one attribute, like **ID** or **class**, we can combine them to make the locator more specific.

Why Should We Use CSS Selectors?

- Sometimes a single attribute (like ID or class) is not enough to locate an element, especially on complex webpages.
- **CSS Selectors** help by letting us combine multiple attributes to find the element more precisely.
- They give us more control over locating elements, especially when there are many similar elements on a page.

How to Write CSS Selectors

Let's take an example of an HTML element and see how we can write different CSS Selectors.

Here’s an example tag:

Now, let’s see how to write CSS Selectors using different combinations:

1. Tag + ID

- **How to Write**: Combine the tag name with the ID.
- **Example**:
- div#idvalue

  This will find a element with the ID idvalue.

---

2. Tag + Class

- **How to Write**: Combine the tag name with the class name.
- **Example**:
- div.classvalue

  This will find a element with the class classvalue.

---

3. Tag + Attribute

- **How to Write**: Combine the tag name with any other attribute.
- **Example**:
- div[name='namevalue']

  This will find a element with the attribute name set to namevalue.

---

4. Tag + Class + Attribute

- **How to Write**: Combine the tag name, class, and any other attribute.
- **Example**:
- div.classvalue[name='namevalue']

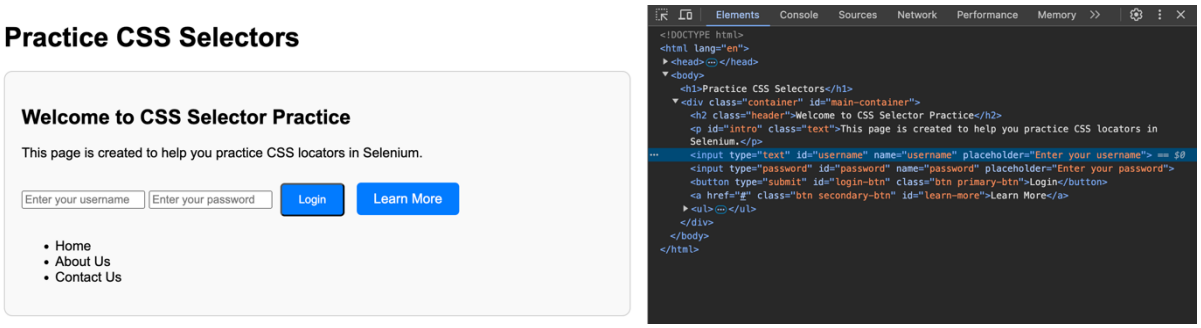
  This will find a element with the class classvalue and the attribute name set to namevalue.

Why These Combinations Are Useful

- Using combinations of **tag names**, **ID**, **class**, and **attributes** helps us create **unique locators**.
- This makes it easier to find elements on complex or dynamic web pages.
- The more specific the locator, the less likely it is to break when the webpage changes.

Example:

Web Page **Dom**



Code snippet

```
// 1. Find element using a combination of tag name and ID.
// The '#' symbol is used to reference the ID in CSS selectors.
driver.findElement(By.cssSelector("input#username")).sendKeys( ...keysToSend: "yogesh");

// 2. Find element using a combination of tag name and class name.
// In CSS, when you have multiple classes, use a dot (.) between class names.
// For example, "btn primary-btn" becomes "btn.primary-btn".
driver.findElement(By.cssSelector("button.btn.primary-btn")).click();

// 3. Find element using a tag and an attribute.
// Use square brackets to target attributes. This is useful when the element does not have an ID or class.
// Example: locate an anchor tag with href="#".
driver.findElement(By.cssSelector("a[href='#']"));

// 4. Find element using a combination of class name and attribute.
// When you have multiple elements with similar attributes,
// you can use both class and attribute to locate the element.
// Example: Locate an anchor tag with class "btn secondary-btn" and href="#".
driver.findElement(By.cssSelector("a.btn.secondary-btn[href='#']"));
```