Locating Elements through CSS and XPath

This guide has mainly three subsections, namely:

4.4.1 Finding the element present on the page by using CSS Selector.

4.4.2 Finding the element present on the page by using XPath.

4.4.3 Pushing the code to your GitHub repositories

Step 4.4.1: Finding the element present on the page using CSS Selector.

- Using CSS Selectors in Selenium. As we all know, CSS stands for Cascading Style Sheets. By using CSS selectors, we can find or select HTML elements on the basis of their id, class, or other attributes. CSS is faster and simpler than XPath, particularly in case of IE browser where Path works very slowly.
- Open Eclipse
- Using Path as a CSS Selector
- CSS Selector has many formats, namely:
  a. **Tag and ID**
    - Syntax: “css = tag#id”
    - Example: driver.findElement(By.cssSelector(“input#email”));

  b. **Tag and Class**
    - Syntax: “css = tag.class”
    - Example: driver.findElement(By.cssSelector(”input.inputtext”));

  C. **Tag and Attribute**
    - Syntax: “css = tag[attribute=value]”
    - Example: driver.findElement(By.cssSelector(“input[name=lastName]”));

  c. **Tag, Class, and Attribute**
    - Syntax: “tag.class[attribute=value]”
    - Example:

driver. findElement(By.cssSelector(“input.inputtext[tabindex=1]”));

d. **Inner text**
   - Syntax: “css = tag.contains(“innertext”)”
   - Example: driver.findElement(By.cssSelector(font:contains(“Boston”)));

Step 4.4.2: Finding the element present on the page using Path.

- In Selenium automation, if the elements are not found by the general locators like id, class, name, etc., then XPath is used to find an element on the web page.
- XPath contains the path of the element situated at the web page. Standard syntax for creating XPath is:

  XPath=//tagname[@attribute='value']

  - **//:** Select current node.
  - **Tagname:** Tagname of the particular node.
  - **@:** Select attribute.
  - **Attribute:** Attribute name of the node.
  - **Value:** Value of the attribute.
- Types of XPath:

There are two types of XPath:

a. **Absolute XPath**
   - It is a direct way to find the element, but the disadvantage of the absolute XPath is that if there are any changes made in the path of the element, then that XPath fails.
   - The key characteristic of XPath is that it begins with the single forward slash (/), which means you can select the element from the root node.
   - Syntax for absolute Path: html/body/div[1]/div[1]/div/h4[1]/b
   - Example: driver.findElement(By.xpath(“html/body/div[1]/div[1]/div/h4[1]/b ”));
   - Writing absolute XPath on the elements which are present in the web page will be very lengthy. To reduce the length, we use relative XPath.

b. **Relative XPath**

- For relative XPath, the path starts from the middle of the HTML DOM structure. It starts with the double forward-slash (//), which means it can search the element anywhere on the web page.
- You can start from the middle of the HTML DOM structure and you don’t need to write long XPath.

- Syntax for relativeXPath: //*[@class=’relativexapath’]
- Example:
  driver.findElement(By.xpath(“//*[@class=’relativexapath’]”))