System.setProperty("webdriver.chrome.driver","D:\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

driver.get("http://facebook.com");

<input type="password" class="inputtext \_55r1 inputtext \_1kbt inputtext \_1kbt" name="pass" id="pass" tabindex="0" placeholder="Password" aria-label="Password">

Input - Tagname

Class, name,id - Attributes

inputtext \_55r1 inputtext \_1kbt inputtext \_1kbt, pass,pass = value

Xpath Syntax:

//tagName[@attribute = 'Value']

//input[@id='email']

//\*[@name='email']

CSS Syntax 1:

tagName[attribute = 'Value']

CSS Syntax 2:

**tagName#id**

<input class="input r4 wide mb16 mt8 username" type="email" value="" name="username" id="username" aria-describedby="error" style="display: block;">

Ex:  input#username

CSS Syntax 3:

**tagName.class name**

<input type="email" class="inputtext login\_form\_input\_box" name="email" id="email" data-testid="royal\_email">

ex : input.inputtext

 login\_form\_input\_box

incase if  any space b/w in classname then replace space with . (dot)

Regular Expression:

//tagName[contains(@attribute,'constant\_value')]  - xpath

<input name="email123" >

Example: //input[contains(@name,'username')]

tagName[Attribute\*='value'] - Css regular expression

input[name\*='username']

Example 1:

<input id=”txtName” class=”textboxcss” tabindex=”1″ type=”text”>

css=input.textboxcss

WebElement cssele = driver.findElements(By.cssSelector(“input.textboxcss”));

Example 2:

<input id=”txtName” class=”textboxcss top” tabindex=”1″ type=”text”>

css=input.textboxcss.top

Here input is tag name followed by dot and class name textboxcss. In example 2 class name is textboxcss<space>top in that case we put dot in between textboxcss and top.

<input value=”Reading” type=”checkbox”>

css=input[type=’checkbox’]

or

css=input[value=’Reading’]

Example:

 In this case when we have common id but other attributes are different, we go with this combination. Syntax for this combination is css=tag#id[attribute=’value’].

<input id=”txtName” class=”textboxcss” tabindex=”1″ name=”taComment” type=”text”>

<input id=”txtName” class=”textboxcss” tabindex=”1″ name=”tbComment” type=”text”>

css=input#txtName[name=’taComment’]

In this case when we have id but we have class name which is common around other elements but other attributes are different, we go with this combination. Syntax for this combination is css=tag.classname[attribute=’value’].

Example:

<input class=”textboxcss” tabindex=”1″ name=”taComment” type=”text”>

<input class=”textboxcss” tabindex=”1″ name=”tbComment” type=”text”>

css=input.textboxcss [name=’taComment’]

**nth-chilld()** – In this case we have same Id or class name and other attributes for different elements, we can go with nth-child(). Syntax for this combination is css=tag:nth-child(n). Here in syntax we can use any combination discussed above. With that we need to use: nth-child(n). n represent child number.

Example:

<ul>

<li>C</li>

<li>C++</li>

<li>C#</li>

<li>Java</li>

<li>Python</li>

<li>Ruby</li>

</ul>

css= li:nth-child(n)

WebElement cssele = driver.findElements(By.cssSelector(“li:nth-child(n)”));

Here if we can see the ul li parent child structure. We have only tag names which are common to everyone. Here if we want to locate sat Java we will put n=4 in above command.

Parent- Child relationship - Xpath: Traverse from parent to Child

//div[@class='RNNXgb']/div/div[2]/div/input

//div[@id='gbw']/div[1]/div[1]/div[1]/div[2]/a[1]

Difference between Relative and Absulte xPath

Relative - doesnot depend with parent nodes and directly writing xpath.(started with double //)

Absulte - Traverse from Parent to Child.  Parent/child  (started with single /)

Relative Xpath is best.

how to move to sibiling nodes? or how to traverse to sibling element using xpath?

.//xpath of root node/following-sibling::tag name[1]    -  first sibliing

.//xpath of root node/following-sibling::tag name[2]    -  Second sibliing

example : .//\*[@id='tablist-tab1']/following-sibling::li[2]

How to traverse back to parent element from child element using xpath?

.//xpath of child node/parent::tag name

example : .//\*[@id='tablist-tab1']/parent::ul

Note : Using CSS we can't come back

Identifying object with text using xpath locator

//\*[text()=' Selenium']

 // To check Whether image is broken or not

If value of naturalWidth attribute is 0 then image is broken or else image is not broken

// Difference b/w navigate().to and Get()

driver. get() never stores history whereas driver. navigate().to() stores browser history so as to be used for other commands forward and back etc. In single page applications while navigate().to() navigates to the page by changing URL like doing forward/backward, get() refreshes page.

What will be the CSS Selector equivalent for a sub-child in xpath expression //div//a?

**Direct child**

XPath: //div/a  
  
CSS: div > a

A direct child in XPATH is defined by the use of a “/“, while on CSS, it’s defined using “>”

**Child or subchild**

XPath: //div//a []  
  
CSS: div a

If an element could be inside another or one it’s childs, it’s defined in XPATH using “//” and in CSS just by a whitespace

**Choosing a specific match**

<ul id = "recordlist">

<li>Cat</li>

<li>Dog</li>

<li>Car</li>

<li>Goat</li>

</ul>

If we want to select the fourth li element (Goat) in this list,

CSS: #recordlist li:nth-of-type(4)

**Sub-string matches**

CSS in Selenium has an interesting feature of allowing partial string matches using ^=, $=, or \*=. I’ll define them, then show an example of each:

^= Match a prefix

CSS: a[id^='id\_prefix\_']

A link with an “id” that starts with the text “id\_prefix\_”

$= Match a suffix

CSS: a[id$='\_id\_sufix']

A link with an “id” that ends with the text “\_id\_sufix”

\*= Match a substring

CSS: a[id\*='id\_pattern']

A link with an “id” that contains the text “id\_pattern”

### **Matching by inner text**

CSS: a:contains('Log Out')