Handling Dynamic XPath:--

Sometimes, we may not identify the element using the locators such as id, class, name, etc. In those cases, we use XPath to find an element on the web page.

**Types of XPath**

1. Absolute XPath—‘/’
2. Relative XPath—‘//’

At times, XPath may change dynamically and we need to handle the elements while writing scripts. Standard way of writing xpath may not work and we need to write dynamic XPath in selenium scripts. Let’s see different way of writing dynamic XPath in Selenium with examples:

1. [Using Single Slash](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#SINGLESLASHXPATH) or Absolute XPath
2. [Using Double Slash](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#DOUBLESLASHXPATH) or Relative XPath
3. [Using Single Attribute](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#SINGLEATTRIBUTEXPATH)
4. [Using Multiple Attribute](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#MULTIPLEATTRIBUTEXPATH)
5. [Using AND](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#ANDXPATH)
6. [Using OR](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#ORXPATH)
7. [Using contains()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#CONTAINSXPATH)
8. [Using starts\_with()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#STARTSWITHXPATH)
9. [Using text()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#TEXTXPATH)
10. [Using last()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#LASTXPATH)

Absolute XPath searches from the beginning of the HTML tag

Relative XPath searches from the middle of the HTML structure.

**Download ChroPath for Chrome Browser.**

<https://chrome.google.com/webstore/detail/chropath/ljngjbnaijcbncmcnjfhigebomdlkcjo?hl=fr&gl=MX>

<input id="Email" type="email" <span class="html-attribute-name">value</span>="" <span class="html-attribute-name">spellcheck</span>="<span class="html-attribute-value">false</span>" class="emailClass"

autofocus="" <span class="html-attribute-name">name</span>="<span class="html-attribute-value">Email</span>" placeholder="Enter your email"/>

**1. Using Single Slash:**

This mechanism is also known as finding elements using Absolute XPath.

Single slash is used to create XPath with absolute path i.e. the XPath would be created to start selection from the document node/start node/parent node.

Syntax:--

|  |  |
| --- | --- |
| 1 | /html/body/div[1]/div[2]/div[2]/div[1]/form/div[1]/div/div[1]/div/div/input[1] |

**2. Double Slash:**

This mechanism is also known as finding elements using Relative XPath.

Double slash is used to create XPath with relative path i.e. the XPath would be created to start selection from anywhere within the document. – Search in a whole page (DOM) for the preceding string

Syntax:--

//form/div[1]/div/div[1]/div/div/input[1]

**3. Single Attribute:**

You could write the syntax in two ways as mentioned below. Including or excluding HTML Tag. If you want to exclude HTML Tag then you need to use **\***

Syntax:--

//<HTML tag>[@attribute\_name='attribute\_value']

or

//\*[@attribute\_name='attribute\_value']

Ex:--

//input[@id='Email']

or

//\*[@id='Email']

**4. Multiple Attribute:**

**Syntax:**

//<HTML tag>[@attribute\_name1='attribute\_value1'][@attribute\_name2='attribute\_value2]

or

//\*[@attribute\_name1='attribute\_value1'][@attribute\_name2='attribute\_value2]

Ex:--

//input[@id='Email'][@name='Email']

or

//\*[@id='Email'][@name='Email']

**5. Using AND:**

**Syntax:**

//<HTML tag>[@attribute\_name1='attribute\_value1' and @attribute\_name2='attribute\_value2]

or

//\*[@attribute\_name1='attribute\_value1' and @attribute\_name2='attribute\_value2]

**6. Using OR:**

**Syntax:**

//<HTML tag>[@attribute\_name1='attribute\_value1' or @attribute\_name2='attribute\_value2]

or

//\*[@attribute\_name1='attribute\_value1' or @attribute\_name2='attribute\_value2]

**7. contains():**It is used to identify an element, when we are familiar with some part of the attributes value of an element.

**Syntax:**

//<HTML tag>[contains(@attribute\_name,'attribute\_value')]

or

//\*[contains(@attribute\_name,'attribute\_value')]

//input[contains(@id,'Email')]

or

//\*[contains(@id,'Email')]

or

//input[contains(@name,'Email')]

or

//\*input[contains(@name,'Email')]

**8. starts-with():**It is used to identify an element, when we are familiar with the attributes value (starting with the specified text) of an element.

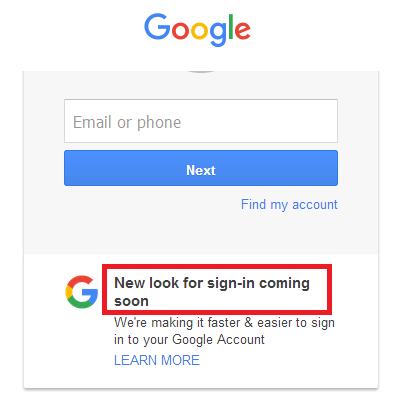
**Syntax:**

//<HTML tag>[starts-with(@attribute\_name,'attribute\_value')]

or

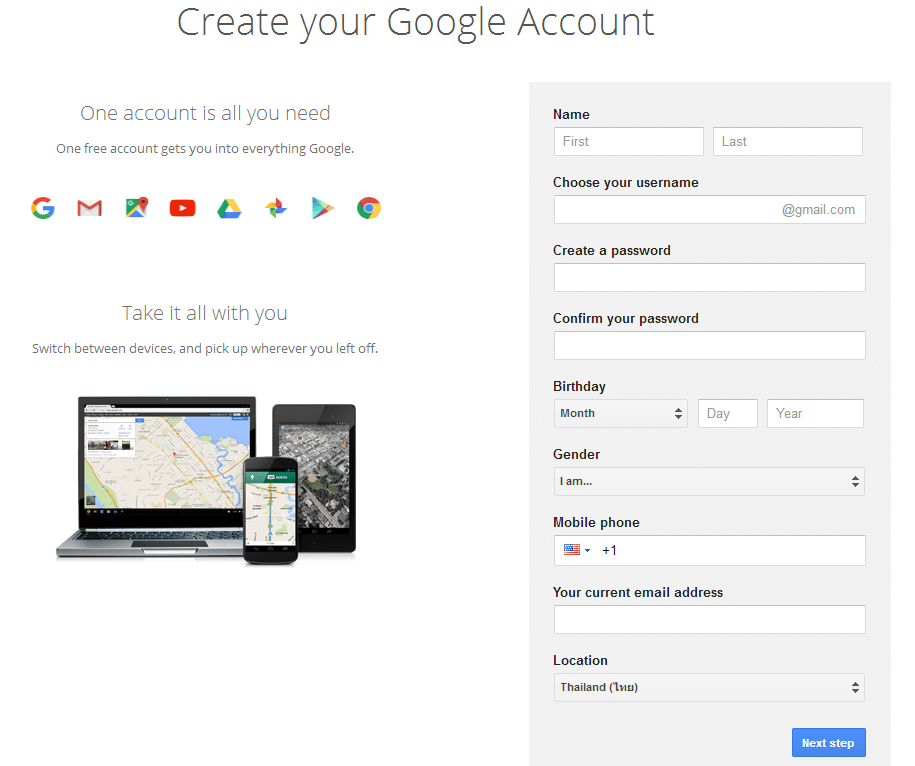
//\*[starts-with(@attribute\_name,'attribute\_value')]

**9. text():** This mechanism is used to locate an element based on the text available on a webpage



//\*[text()='New look for sign-in coming soon']

**10. last():**Selects the last element (of mentioned type) out of all input element present



To identify the element (last text field ) “Your current email address”, we could use the below xpath.

findElement(By.xpath("(//input[@type='text'])[last()]"))

To identify the element “Year”, we could use the below xpath.

**[last()-1]** – Selects the last but one element (of mentioned type) out of all input element present

//tbody/tr[last()]

//table[@id='customers']//tbody/tr[last()-1]

CSS Selector:--

CSS and XPath are used for Dynamic Search.

We use CSSSelector for its faster performance.

|  |  |
| --- | --- |
| Attribute | Symbol |
| Using ID | Use # symbol |
| Class name | . symbol |
| Attribute | Tagname[attr=’value’] |
| Multiple attributes | Tagname[attr1=’val’][attr2=’val’] |
| Contains | \* |
| Startswith | ^ |
| Endswith | $ |