Types of XPath in Selenium & it's usages

1. XPath also known as the XML path is a language that helps to query the XML documents.
2. XPath helps us to identify and locate elements on the web page using certain attributes, conditions, values and by HTML element tag name.
3. Syntax - *//tagname[@attribute_name= 'value']* where:

   // = current node.

   tagname = HTML tag name you want to locate.

   @attribute_name = attribute name the HTML element contains which you are referring to.

   value = value of the attribute you want to refer.
4. Types of XPath – Absolute XPath and Relative XPath.
5. Absolute XPath - starts with the root node of the page and traverses from the root node through the whole DOM to reach to the desired element. It starts with a single slash '/' and traverses from the root element to the desired element.

   Example: */html/body/div[2]/div/div/div[2]/div/div[6]/div/div[2]*
6. Relative XPath – can be started from middle of the DOM structure or any where on the page. It starts with a double slash '//' which denotes the current node. The xpath search starts from the mentioned tag name along with the attribute value defined.

   Example: *//input[@placeholder='Full Name']*

➢ **HTML DOM Elements –**

<label class="form-label" id="userName-label">Name</label>

<input required="" placeholder="First Name" type="text" id="firstName" class="form-control">

➢ **XPath for above HTML element:**

1. Using Attribute value - *//input[@placeholder='First Name']*
2. Using Conditions AND & OR with attribute value:

   AND - *//input[@placeholder='First Name' AND @type='text']*

   OR - *//input[@placeholder='First Name' OR @type='text']*
3. Using text(), contains() & starts-with() functions:

   text() – *//label[text()='Name']*

   contains() – *//label[contains(@id, 'userName')]*

   starts-with() – *//label[starts-with(@id, 'user')]*

4. Combinations of all the above functions and conditions:

   //input[@placeholder='First Name' AND contains(@id, 'Name')]

   //label[@id='userName-label' OR text()='Name']

- **Axes in XPath –**

We can use the relationship between different element on DOM to locate them on the web page, which are more suitable for complex xpaths.

There are various axes available: ancestor, descendant, parent, child, preceding, following, following-sibling.

- **HTML DOM Elements –**

```
<div class="main div node">
        <div class="col-md-3 col-sm-12">
                <label class="form-label" id="userName-label">Name</label>
                <label class="form-label" id="title-label">Title</label>
        </div>
        <div class="col-md-4 col-sm-6">
                <input required="" placeholder="First Name" type="text" id="firstName" class="form-control">
                <input required="" placeholder="Title" type="text" id="titleName" class="mr-sm-3 form-control">
        </div>
</div>
```

- **XPath for above HTML element using Axes:**

1. Using ancestor axes to select main div tag:

   //label[text()='Title']//ancestor::div

2. Using descendant axes to select 2nd input tag:

   //div[contains(@class, 'main')]//descendant::input[@id='titleName']

3. Using parent axes to select parent div tag:

   //label[@id='title-label' AND text()='Title']//parent::div[contains(@class, 'md-3')]

4. Using child axes to select 2nd label field:

   //div[contains(@class, 'md-3')]//child::label[text()='Title']

5. Using preceding axes to select 2nd label field:

   //label[text()='Title']//preceding::label

6. Using following axes to select 2nd input field:

   //input[@id= 'firstName']//following::input

7. Using following-sibling axes to select 3rd div tag:

   //div[contains(@class, 'col-md-3')]//following-sibling::div

- **Few more point to keep in mind:**

1. If you want to traverse to an element which is one level above the current node you are on, you can use the '/..' forward slash followed by two dots. Ex - *//label[text()='Female']/..//input*
2. Using '*' instead of tag name - an asterisk '*' is consider as an wildcard character in xpath, whenever you want to perform actions based on attribute value instead of element tag then we can use '*' at the start of the xpath instead of tag name.

   Example - *//*[text()='Other']*