SELENIUM LOCATORS

SELENIUM DON'T SPEAK ENGLISH...!!

- A webpage looks different to a computer than to a human. Most humans don't care how computers look at web pages, but it may matter to you. That's because if you are writing a Selenium test, you have to know how to see a web page the way a computer does.

- A webpage has things on it. The text you're reading now isn't a blog post to a computer – it's a thing in a webpage. So are all the links and buttons on this page.

- Everything in the page is an element. Most of them live inside another element.

WHAT IS X-PATH?

- X-path is normally called XML (Extensible Markup Language)path, used to navigate through elements and attributes in the XML document.
- X-path is the query/language for finding the information on XML document.
- X-path contains a library of standard functions
- X-path is a major element in the XSLT (Extensible Stylesheet Language Transformations)standard
- X-path is a W3C recommendation
- Normally in every web based applications X-paths or locators are unique addresses for each and every web elements on which the selenium can perform action.
- X-paths are derived from html paths of the web objects.
- X-paths are normally constants, rarely change (The label of the text box can change but position will remain same/unchanged)
- Sometimes X-path may vary whenever u refresh the web page, the varying X=path will always have some kind of pattern. We need javascript to handle such dynamic X-path.

ABSOLUTE X-PATH

- If location path starts with root node or '/' then it is absolute x-path (Full path).
- It uses complete path from root node to desired element.
- Advantage : Identifies the element very fast
- Disadvantage: If any other tag has been added in between, then this path doesn't work
- Example: html/head/body/table/tbody/tr/th
  [If the form tag has added between body and table, then the absolute x-path will be html/head/body/form/table/tbody/tr/th (The first path will not work as 'form' tag added in between)
- eg. xpath=html/head/body/div[3]/form/fieldset/input[2]

RELATIVE X-PATH

- If path starts from the node that we have selected, its relative path (small path related with tangle and attribute value).
- It starts with '//'.
- Start by referencing the element you want and go from there.
- Preferred over absolute path.
- Syntax: //table/tbody/tr/th
- Advantage of using relative x-path is, you don't need to mention the long x-path, you can start from the middle or in between.
- Disadvantage here is, it will take more time in identifying the element as we specify the partial path not (exact path).
- If there are multiple elements for the same path, it will select the first element that is identified

DIFFERENT WAYS OF WRITING X-PATH

- **Xpath locator using @ and attribute**
- xpath=//body/div[3]/form/fieldset/input[@type='search']

  here, /input[@type=‘search’] describes the input node having attribute type=‘search'
- **Xpath using @ and attribute**

  xpath=//input[@accesskey='F']

  //input[@accesskey=‘F’], (which is root node)describes the input node having attribute @accesskey=‘F'
- **Xpath using contains keyword**

  xpath=//input[contains(@id, "searchInput")]

  used contains keyword to identify id attribute with text "searchInput"
- **Xpath using and with attributes**

  xpath=//input[contains(@id, "searchInput") and contains(@accesskey,"F")]

  two attributes in input node
- **Using starts-with keyword**

  xpath=//input[starts-with(@type, "s")]

  input node with attribute is 'type' and its value is starting with 's' (here it will get type = 'search').

DIFFERENT WAYS OF WRITING X-PATH

- Using OR (|) condition with xpath
  xpath=//input[@accesskey='F'] | //input[@id='searchInput']
  xpath=//input[@accesskey='F' or @id='searchInput']
  it will find input text box with accesskey='F' or @id='searchInput'. If any one found then it will locate it. Very useful when elements appears alternatively.

- Using wildcard * with to finding element xpath
  xpath=//*[@accesskey='F']
  This finds any element that has accessKey attribute="F"

- Using Text() to find element
  xpath=//div/a[text()="English"]
  This will find a link that has text ="English"

what is CSS?

- CSS is "Cascading Style Sheets" and it is defined to display HTML in structur and colourful styles are applied to webpage.
- Selectors are patterns that match against elements in a tree, and as such forn one of several technologies that can be used to select nodes in an XML document.
- CSS has more Advantage than X-path
- CSS is much more faster and simpler and more readable than the X-path.
- In IE X-path works very slow, where as Css works faster when compared to X path.
- In CSS there are two special characters which has important role to play.

1. dot(.) refers to class.
   Syntax: css=input.submitbtn
2. Hash(#) refers to Id
   Syntax: css=input#destination

MATCHING BY INNER TEXT

- :contains() will match elements with the desired text block: css=a:contains('Log Out') This will find the log out button on your page no matter where it's located.
- Absolute path: cssSelector =html>body>div>p>input;
- Relative path: cssSelector=input *the first instance found
- tag with attribute value:
  cssSelector=button[name=cancel]; (button tag with attribute name as cancel) Special attributes:
  id: cssSelector=button#save; (button tag with id save)
  class: cssSelector = input.username ; (tag & class attribute)

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
  $= Match a suffix
  *= Match a substring
- css=a[id^='id_prefix_']: A link with an "id" that starts with the text "id_prefix_"
- css=a[id$='_id_sufix']: A link with an "id" that ends with the text "_id_sufix"
- css=a[id*='id_pattern']: A link with an "id" that contains the text "id_pattern"