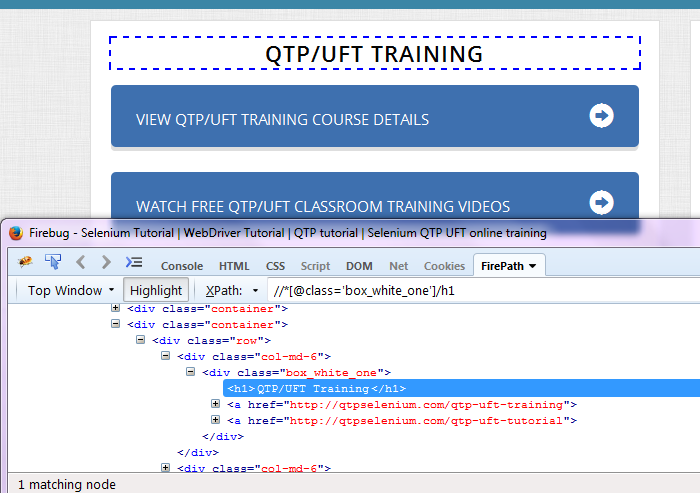
**Relative x-path:**

For Relative X-path 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 at the webpage.

You can starts from the middle of the HTML DOM structure and no need to write long x-path.

Below is the example of a relative X-Path expression of the same element shown in the below screen. This is the common format used to find element through a relative X-Path.

**Relative x-path**: //\*[@class='box\_white\_one']/h1



**CSS selectors:-**

When we don't have an option to choose Id or Name, we should prefer using CSS locators as the best alternative.

CSS is "Cascading Style Sheets" and it is defined to display HTML in structured and colorful styles are applied to webpage.

Selectors are patterns that match against elements in a tree, and as such form 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 than the X-path.
* In IE X-path works very slow, where as Css works faster when compared to X-path.

**Syntax:**

tagName[attributename=attributeValue]

Example 1: input[id=email]

Example 2: input[name=email][type=text]

Example 3:X-path= //div[@id='container']/div/header/div[2]/div/ul/li[2]/a

Example 4:CSS selector=div[id='container'] div header div:nth-child(2) div ul li:nth-child(2) a

Using CSS locators, we can also locate elements with sub-strings. Which are really help full when there are dynamically generated ids in webpage

There are there important special characters:

1. '^' symbol, represents the starting text in a string.

Example : X-path= .//div[@id='email']

Css selector ‘^’ Start-with= div[id^='ema']

Start-with= //div[starts-with@(id,'ema')]

1. '$' symbol represents the ending text in a string.

Css selector ‘$’ Start-with= div[id$='ail']

Ends-with=//div[ends-with@(id,'ail')]

1. '\*' symbol represents contains text in a string.

Contains= ‘\*’ contains= //\*[contains(@id,'mai']

**X-Path Contains, Sibling, Ancestor Functions in Selenium WebDriver**

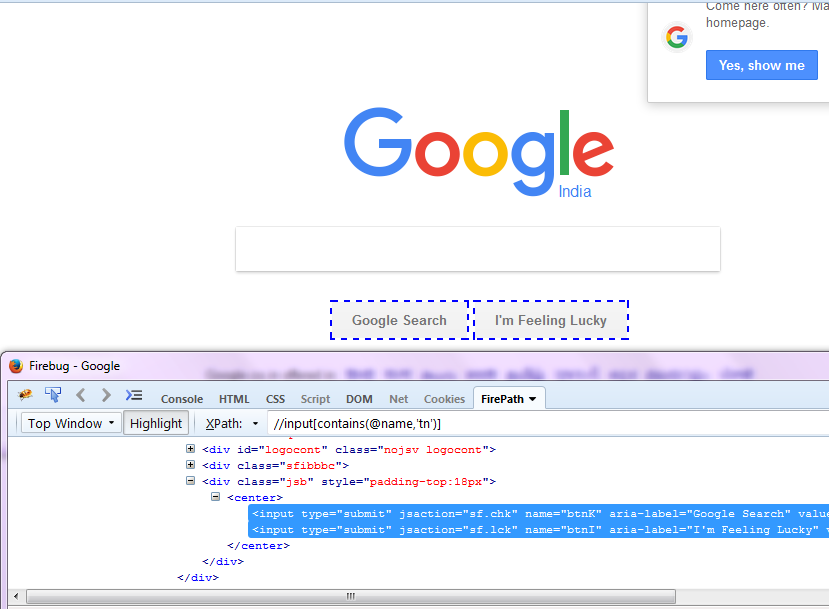
If a simple X-Path is not able to find a complicated web element for our test script, we need to use the functions from X-Path 1.0 library. With the combination of these functions, we can create more specific X-Path.

1. Contains
2. Sibiling
3. Following

**Contains:**

By using 'contains' function in X-Path, we can extract all the elements which matches a particular text value.

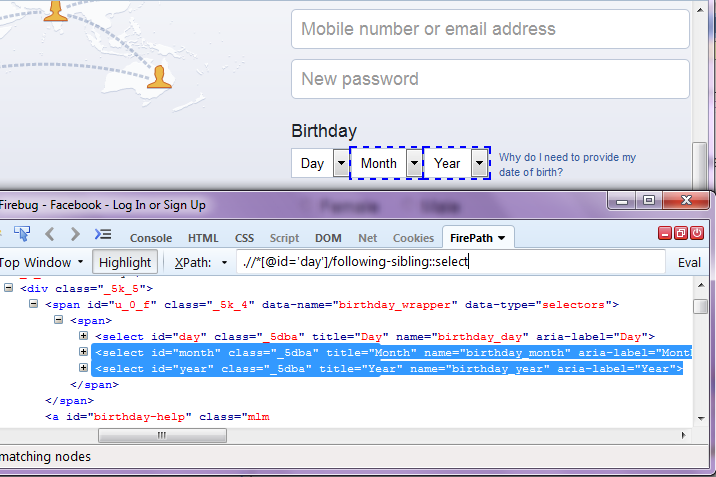
Example : //input[contains(@name,'tn')]

****

**Sibling:**

Using sibling keyword, we can fetch a web element on the which is related to some other element.

Example : .//\*[@id='day']/following-sibling::select



**Following:-**

Using following keyword, we can fetch a child element and same tag name element on the which is related to some other element.

**Example:-** **.//\*[@id='u\_0\_1']//following::div**

