Interaction with a web page requires a user to locate the web element. Find Element command is used to uniquely identify a (one) web element within the web page. Whereas, Find Elements command is used to uniquely identify the list of web elements within the web page. There are multiple ways to uniquely identify a web element within the web page such as ID, Name, Class Name, Link Text, Partial Link Text, Tag Name and XPATH.

## FindElement command syntax:

Find Element command takes in the By object as the parameter and returns an object of type WebElement. By object in turn can be used with various locator strategies such as ID, Name, Class Name, XPATH etc. Below is the syntax of FindElement command in Selenium web driver.

WebElement elementName = driver.findElement(By.LocatorStrategy("LocatorValue"));

Locator Strategy can by any of the following values.

* ID
* Name
* Class Name
* Tag Name
* Link Text
* Partial Link Text
* XPATH

Locator Value is the unique value using which a web element can be identified. It is the responsibility of developers and testers to make sure that web elements are uniquely identifiable using certain properties such as ID or name.

Example:

WebElement loginLink = driver.findElement(By.linkText("Login"));

## FindElement command syntax:

Find Element command takes in the By object as the parameter and returns an object of type WebElement. By object in turn can be used with various locator strategies such as ID, Name, Class Name, XPATH etc. Below is the syntax of FindElement command in Selenium web driver.

WebElement elementName = driver.findElement(By.LocatorStrategy("LocatorValue"));

Locator Strategy can by any of the following values.

* ID
* Name
* Class Name
* Tag Name
* Link Text
* Partial Link Text
* XPATH

Locator Value is the unique value using which a web element can be identified. It is the responsibility of developers and testers to make sure that web elements are uniquely identifiable using certain properties such as ID or name.

Example:

WebElement loginLink = driver.findElement(By.linkText("Login"));

## FindElements command syntax:

Find Elements command takes in By object as the parameter and returns a list of web elements. It returns an empty list if there are no elements found using the given locator strategy and locator value. Below is the syntax of find elements command.

List<WebElement> elementName = driver.findElements(By.LocatorStrategy("LocatorValue"));

Example:

List<WebElement> listOfElements = driver.findElements(By.xpath("//div"));

## Introduction to WebElement, findElement(), findElements()

Selenium Web Driver encapsulates a simple form element as an object of WebElement.

There are various techniques by which the WebDriver identifies the form elements based on the different properties of the Web elements like ID, Name, Class, XPath, Tagname, CSS Selectors, link Text, etc.

Web Driver provides the following two methods to find the elements.

* findElement() – finds a single web element and returns as a WebElement object.
* findElements() – returns a list of WebElement objects matching the locator criteria.

Let's see the code snippets to get a single element – Text Field in a web page as an object of WebElement using findElement() method. We shall cover the findElements() method of finding multiple elements in subsequent tutorials.

Step 1: We need to import this package to create objects of Web Elements

Step 2: We need to call the findElement() method available on the WebDriver class and get an object of WebElement.

Refer below to see how it is done.

## Input Box

Input boxes refer to either of these two types:

1. Text Fields- text boxes that accept typed values and show them as they are.
2. Password Fields- text boxes that accept typed values but mask them as a series of special characters (commonly dots and asterisks) to avoid sensitive values to be displayed.

### Locators

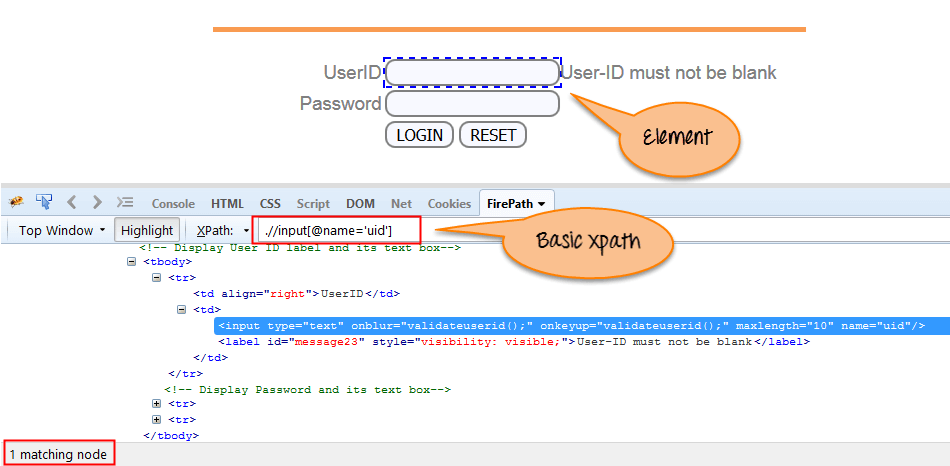
The method findElement() takes one parameter which is a locator to the element. Different locators like By.id(), By.name(), By.xpath(), By.CSSSelector() etc. locate the elements in the page using their properties like id, name or path, etc.

## Using XPath Handling complex & Dynamic elements in Selenium

### 1) Basic XPath:

XPath expression select nodes or list of nodes on the basis of attributes like ID , Name, Classname, etc. from the XML document as illustrated below.

Xpath=//input[@name='uid']



Some more basic xpath expressions:

Xpath=//input[@type='text']

Xpath= //label[@id='message23']

Xpath= //input[@value='RESET']

Xpath=//\*[@class='barone']

Xpath=//a[@href='http://demo.guru99.com/']

Xpath= //img[@src='//cdn.guru99.com/images/home/java.png']

### 2) Contains():

Contains() is a method used in XPath expression. It is used when the value of any attribute changes dynamically, for example, login information.

The contain feature has an ability to find the element with partial text as shown in below example.

In this example, we tried to identify the element by just using partial text value of the attribute. In the below XPath expression partial value 'sub' is used in place of submit button. It can be observed that the element is found successfully.

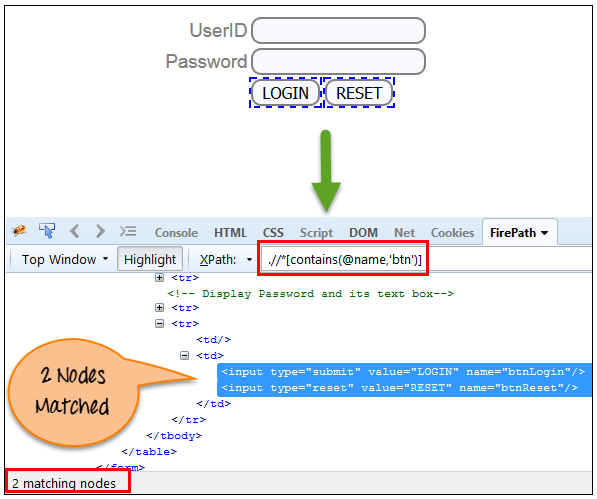
Complete value of 'Type' is 'submit' but using only partial value 'sub'.

Xpath=//\*[contains(@type,'sub')]

Complete value of 'name' is 'btnLogin' but using only partial value 'btn'.

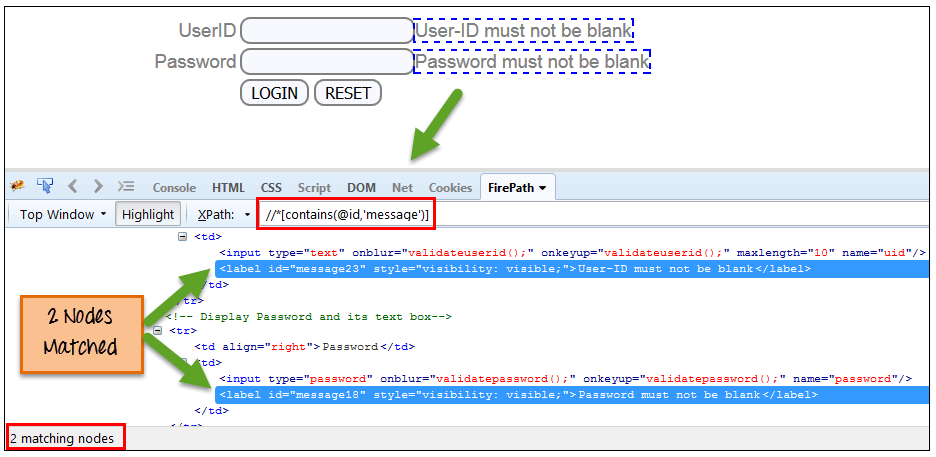
Xpath=//\*[contains(@name,'btn')]

In the above expression, we have taken the 'name' as an attribute and 'btn' as an partial value as shown in the below screenshot. This will find 2 elements (LOGIN & RESET) as their 'name' attribute begins with 'btn'.



Similarly, in the below expression, we have taken the 'id' as an attribute and 'message' as a partial value. This will find 2 elements ('User-ID must not be blank' & 'Password must not be blank') as its 'name' attribute begins with 'message'.

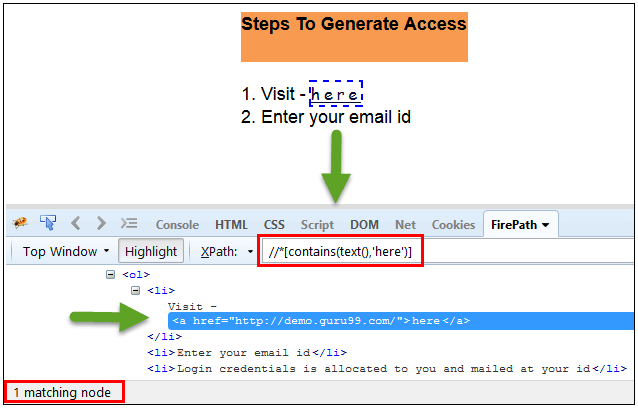
Xpath=//\*[contains(@id,'message')]



In the below expression, we have taken the "text" of the link as an attribute and 'here' as a partial value as shown in the below screenshot. This will find the link ('here') as it displays the text 'here'.

Xpath=//\*[contains(text(),'here')]

Xpath=//\*[contains(@href,'guru99.com')]



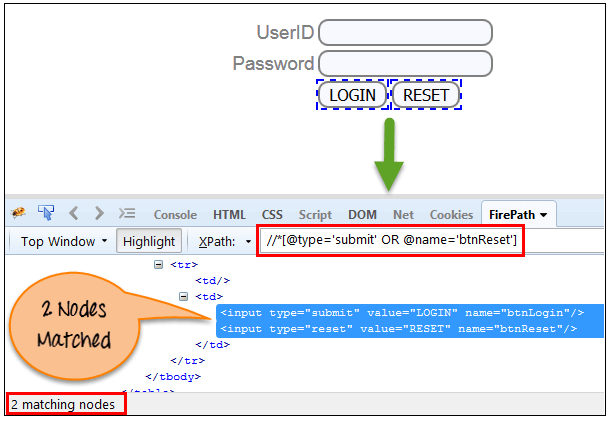
### 3) Using OR & AND:

In OR expression, two conditions are used, whether 1st condition OR 2nd condition should be true. It is also applicable if any one condition is true or maybe both. Means any one condition should be true to find the element.

In the below XPath expression, it identifies the elements whose single or both conditions are true.

Xpath=//\*[@type='submit' OR @name='btnReset']

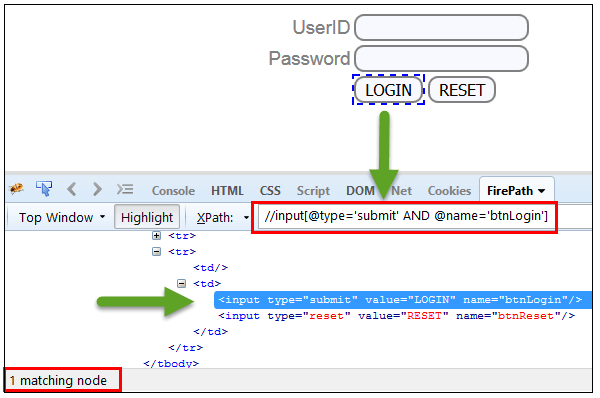
Highlighting both elements as "LOGIN " element having attribute 'type' and "RESET" element having attribute 'name'.



In AND expression, two conditions are used, both conditions should be true to find the element. It fails to find element if any one condition is false.

Xpath=//input[@type='submit' and @name='btnLogin']

In below expression, highlighting 'LOGIN' element as it having both attribute 'type' and 'name'.



### 4) Start-with function:

Start-with function finds the element whose attribute value

changes on refresh or any operation on the webpage. In this expression, match the starting text of the attribute is used to find the element whose attribute changes dynamically. You can also find the element whose attribute value is static (not changes).

For example -: Suppose the ID of particular element changes dynamically like:

Id=" message12"

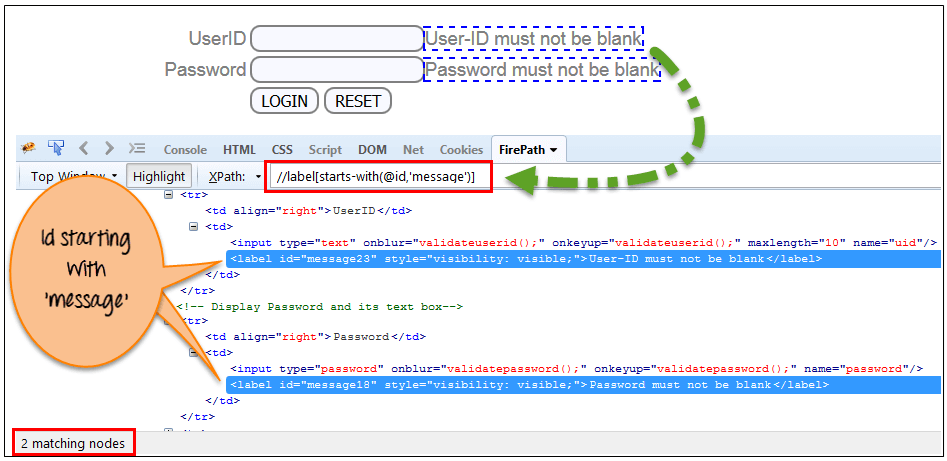
Id=" message345"

Id=" message8769"

and so on.. but the initial text is same. In this case, we use Start-with expression.

In the below expression, there are two elements with an id starting "message"(i.e., 'User-ID must not be blank' & 'Password must not be blank'). In below example, XPath finds those element whose 'ID' starting with 'message'.

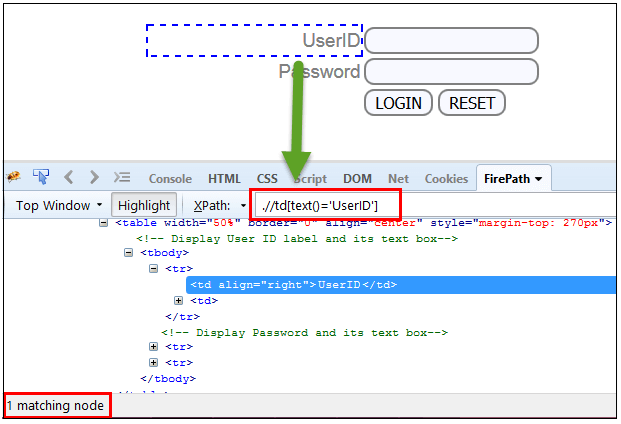
Xpath=//label[starts-with(@id,'message')]



### 5) Text():

In this expression, with text function, we find the element with exact text match as shown below. In our case, we find the element with text "UserID".

Xpath=//td[text()='UserID']



### 6) XPath axes methods:

These XPath axes methods are us