**How to Identify Web Elements Using Selenium Xpath and Other Locators**

**What is Locator?**

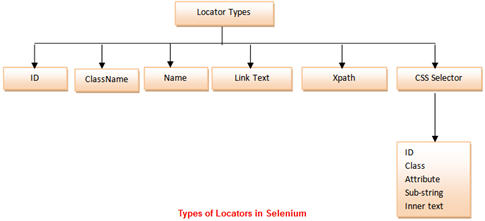
Locator can be termed as an address that identifies a web element uniquely within the webpage. Locators are the HTML properties of a web element which tells the Selenium about the web element it need to perform action on.

There is a diverse range of web elements. **The most common amongst them are:**

* Text box
* Button
* Drop Down
* Hyperlink
* Check Box
* Radio Button

### ****Types of Locators****

Identifying these elements has always been a very tricky subject and thus it requires an accurate and effective approach. Thereby, we can assert that more effective the locator, more stable will be the automation script. Essentially every Selenium command requires locators to find the web elements. Thus, to identify these web elements accurately and precisely we have different types of locators.

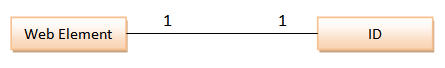
[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Types-of-Locators-in-Selenium-1.jpg)

**Now let’s understand further by exercising each of them independently.**

Before we start with the locators, let me take a moment to introduce the application under test. We would be using “https://accounts.google.com/” for locating different types of web elements using different locator types.

### ****Using ID as a Locator****

The best and the most popular method to identify web element is to use ID. The ID of an each element is alleged to be unique.

[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Types-of-Locators-in-Selenium-2.jpg)

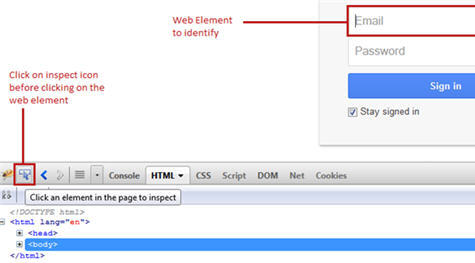
In this sample, we would access “Email” text box present in the login form at gmail.com.

**Finding an ID of a web element** **using Firebug**

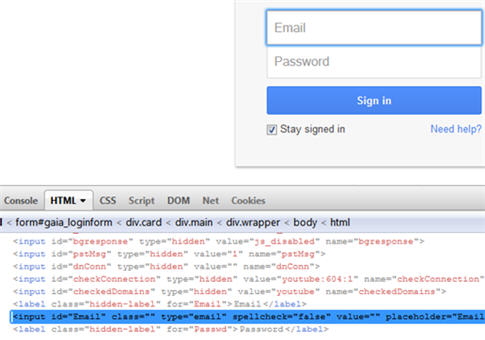
**Step 1**: Launch the web browser (Firefox) and navigate to “https://accounts.google.com/”.

**Step 2**: Open firebug (either by pressing F12 or via tools).

**Step 3**: Click on the inspect icon to identify the web element.

[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-13.jpg)

**Step 4**: Hover on the web element (Email textbox in our case) on which we desire to perform some action. In the firebug section, one can see the corresponding html tags being highlighted.

[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-2.jpg)

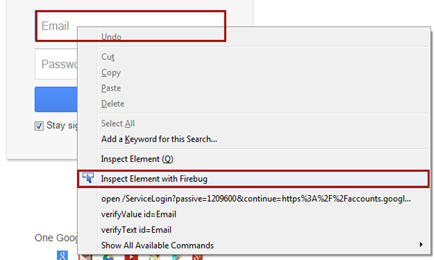
**Step 5**: Be cognizant about the ID attribute and take a note of it. Now we need to verify if the ID indentified is able to find the element uniquely and flawlessly.

**Syntax**: id = id of the element

In our case, the id is “Email”.

Alternative approach:

Instead of following step 2 to 4, we can directly locate / inspect the web element by right clicking on the web element (Email Textbox) whose locator value we need to inspect and clicking on the option “Inspect Element with Firebug”. Thus, this click event triggers the expansion of firebug section and the corresponding html tag would be highlighted.

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-3.jpg)

**Verify the locator value**

Assuming that the browser is open and is re-directed to “https://accounts.google.com/”.

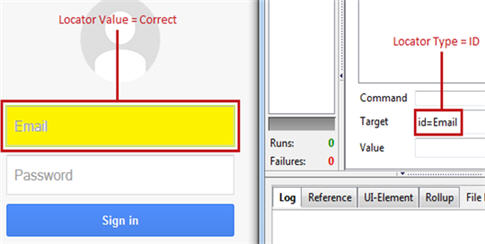
**Step 1**: Launch Selenium IDE.

**Step 2**: Click on the first row in the editor section.

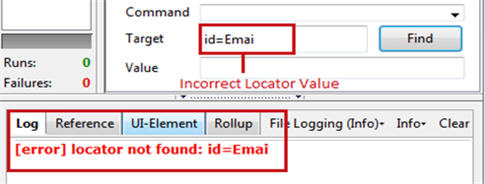
**Step 3**: Type “id=Email” i.e. the locator value in the target box.

**Step 4**: Click on the Find Button. If the provided locator value is legitimate then the Email textbox will be highlighted with yellow color with a florescent green border around the field. If the locator value provided is incorrect, an error message would be printed in the log pane at the bottom of Selenium IDE.

**Case 1** – Locator Value = Correct

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-4.jpg)

**Case 2** – Locator Value = Incorrect

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-5.jpg)

**Step 5**: In order to verify further, user can also execute “type” command against the given target by providing some value in the “Value” field. If the execution of the command enters the specified value in the Email text box that means the identified locator type is correct and accessible.

### ****Using ClassName as a Locator****

There is only a subtle difference between using ID as a locator and using classname as a locator.

In this sample, we would access “Need Help?” hyperlink present at the bottom of the login form at gmail.com.

**Finding a classname of a web element** **using Firebug**

**Step 1**: Locate / inspect the web element (“Need help?” link in our case) by right clicking on the web element whose locator value we need to inspect and clicking on the option “Inspect Element with Firebug”.

**Step 2**: Be cognizant about the classname attribute and take a note of it. Now we need to verify if the classname indentified is able to find the element uniquely and accurately.

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-6.jpg)

**Syntax:** class = classname of the element

------------

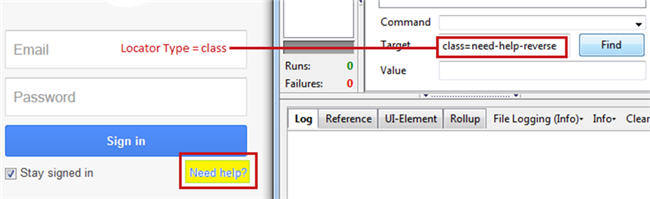
In our case, the classname is “need-help-reverse”

**Verify the locator value**

**Step 1**: Type “class= need-help-reverse” in the target box in the Selenium IDE.

**Step 2**: Click on the Find Button. Notice that the hyperlink will be highlighted with yellow color with a florescent green border around the field.

***(Click to view enlarged image)***

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-7.jpg)

### ****Using name as a Locator****

Locating a web element using name is very much analogous to previous two locator types. The only difference lies in the syntax.

In this sample, we would access “Password” text box present in the login form at gmail.com.

**Syntax:** name = name of the element

In our case, the name is “Passwd”.

**Verify the locator value**

**Step 1**: Type “name= Passwd” in the target box and click on the Find Button. Notice that the “Password” textbox would be highlighted.

### ****Using Link Text as a Locator****

All the hyperlinks on a web page can be indentified using Link Text. The links on a web page can be determined with the help of anchor tag (<a>). The anchor tag is used to create the hyperlinks on a web page and the text between opening and closing of anchor tags constitutes the link text (<a>Some Text</a>).

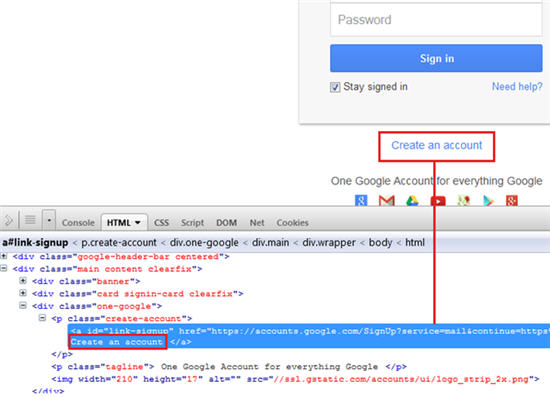
In this sample, we would access “Create an account” link present at the bottom of the login form at gmail.com.

**Finding a link text of a web element using Firebug**

**Step 1**: Locate / inspect the web element (“Create an account” link in our case) by right clicking on the web element whose locator value we need to inspect and clicking on the option “Inspect Element with Firebug”.

**Step 2**: Be cognizant about the text present within the <a> </a> tags and take a note of it. Hence this text will be used to identify the link on a web page uniquely.

***(Click to view enlarged image)***

[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-8.jpg)

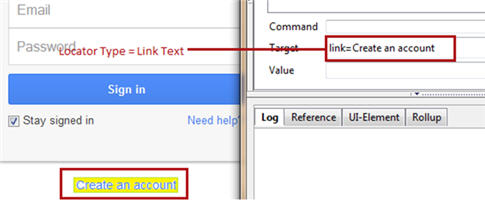
**Syntax:** link = link text of the element

In our case, the link text is “Create an account”.

**Verify the locator value**

**Step 1**: Type “link=Create an account” i.e. the locator value in the target box in Selenium IDE.

**Step 2**: Click on the Find Button. Notice that the link would be highlighted with yellow color with a florescent green border around the field.

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-9.jpg)

### ****Using Xpath as a Locator****

Xpath is used to locate a web element based on its XML path. XML stands for Extensible Markup Language and is used to store, organize and transport arbitrary data. It stores data in a key-value pair which is very much similar to HTML tags. Both being mark up languages and since they fall under the same umbrella, xpath can be used to locate HTML elements.

The fundamental behind locating elements using Xpath is the traversing between various elements across the entire page and thus enabling a user to find an element with the reference of another element.

**Xpath can be created in two ways:**

**Relative Xpath**

Relative Xpath begins from the current location and is prefixed with a “//”.

For example: //span[@class=’Email’]

**Absolute Xpath**

Absolute Xpath begins with a root path and is prefixed with a “/”.

For example: /html/body/div/div[@id=’Email’]

**Key Points:**

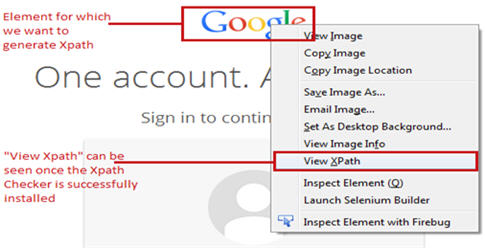
* The success rate of finding an element using Xpath is too high. Along with the previous statement, Xpath can find relatively all the elements within a web page. Thus, Xpaths can be used to locate elements having no id, class or name.
* Creating a valid Xpath is a tricky and complex process. There are plug-ins available to generate Xpath but most of the times, the generated Xpaths fails to identify the web element correctly.
* While creating xpath, user should be aware of the various nomenclatures and protocols.

**Selenium Xpath Examples**

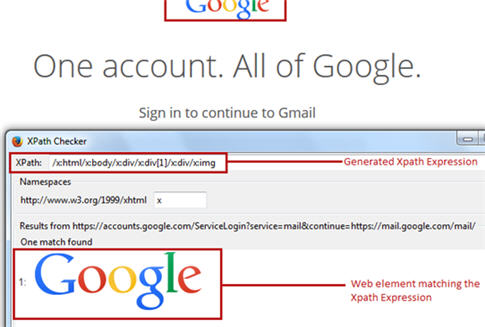
**Xpath Checker**

Creating Xpath becomes a little simpler by using Xpath Checker. Xpath Checker is a firefox add-on to automatically generate Xpath for a web element. The add-on can be downloaded and installed like any other plug-in. The plug-in can be downloaded from “https://addons.mozilla.org/en-US/firefox/addon/xpath-checker/”.

As soon as the plug-in is installed, it can be seen in the context menu by right clicking any element whose xpath we want to generate.

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-10.jpg)

Click on the “View Xpath” to see the Xpath expression of the element. An editor window would appear with the generated Xpath expression. Now user has the liberty to edit and modify the generated Xpath expression. The corresponding results would be updated cumulatively.

[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-11.jpg)

Note that the Xpath Checker is available for other browsers as well.

But re-iterating the fact, that most of the times, the generated Xpaths fails to identify the web element rightly. Thus, it is recommended to create our own Xpath following the pre defined rules and protocols.

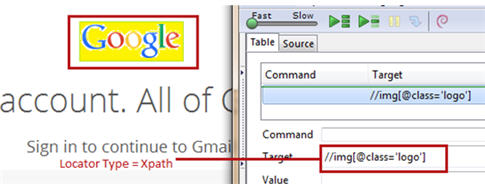
In this sample, we would access “Google” image present at the top of the login form at gmail.com.

**Creating a Xpath of a web element**

**Step 1**: Type “//img[@class=’logo’]” i.e. the locator value in the target box within the Selenium IDE.

**Syntax:** Xpath of the element

**Step 2**: Click on the Find Button. Notice that the image would be highlighted with yellow color with a florescent green border around the field.

[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/10/Selenium-Locators-12.jpg)

### ****Conclusion****

Here are the cruxes of this article.

* Locators are the HTML properties of a web element which tells the Selenium about the web element on which it needs to perform actions.
* There is a wide range of web elements that a user may have to interact with on a regular basis. Some of them are: Text box, Button, Drop Down, Hyperlink, Check Box, and Radio Button.
* With the varied range of web elements comes a vast province of strategies/approaches to locate these web elements.
* Some of the extensively used locator types are: ID, ClassName, Link Text, Xpath, CSS Selectors and Name.

Note: Owing to the fact that creating CSS Selector and Xpath requires a lot of efforts and practice, thus the process is only exercised by more sophisticated and trained users.