

To practice the xpath examples, use the following website:
demosite.center/wordpress/wp-login.php (Username:admin Password:demo123)

Also remember Xpath Axes (www.w3schools.com/xsl/xpath_axes.asp) <== Very helpful

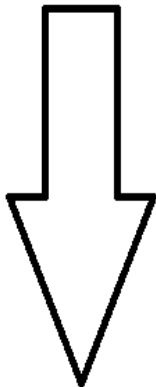
=====

Relative Xpath Method

Using single attribute Syntax: `//tagname[@attribute-name='value1']`

Examples

```
//a [@href='http://www.google.com']  
//input[@id='name']  
//input[@name='username']  
//img[@alt='sometext']
```



The screenshot shows a web login form with fields for Email and Password, a Sign in button, and a checkbox for 'Stay signed in'. A red arrow points to the Email field. Below the form, the FirePath tool is open, showing the XPath `//input[@id='Email']` selected. The DOM tree shows the HTML structure, with the Email input element highlighted in blue.

```
XPath: //input[@id='Email']  
  
<input id="checkConnection" type="hidden" value="youtube:940:1" name="checkConnection"/>  
<input id="checkedDomains" type="hidden" value="youtube" name="checkedDomains"/>  
<label class="hidden-label" for="Email">Email</label>  
<input id="Email" type="email" spellcheck="false" value="" placeholder="Email" name="Email"/>  
<label class="hidden-label" for="Password">Password</label>
```

Using multiple attributes Syntax: `//tagname[@attribute1='value1'][@attribute2='value2']`

Examples

`//a[@id='id1'][@name='namevalue1']`

`//img[@src=''][@href='']`

The screenshot illustrates the use of XPath with multiple attributes. It shows a login form with fields for Email and Password, a Sign in button, and checkboxes for 'Stay signed in' and 'Need help?'. A red dashed box highlights the Password input field. A red arrow points from this field to the FirePath extension interface. In the FirePath interface, the XPath `//input[@id='Passwd'][@name='Passwd']` is entered in the search bar. The results pane shows the corresponding HTML element: `<input id="Passwd" class="" type="password" placeholder="Password" name="Passwd"/>`, which is highlighted in blue.

```
<label class="hidden-label" for="Email">Email</label>
<input id="Email" type="email" spellcheck="false" value="" placeholder="Email" name="Email"/>
<label class="hidden-label" for="Passwd">Password</label>
<input id="Passwd" class="" type="password" placeholder="Password" name="Passwd"/>
<input id="signIn" class="kc-button kc-button-submit type="submit" value="Sign in" name="signIn"/>
<label class="remember">
  <input type="hidden" value="1" name="rmShown"/>
```

Using contains method Syntax: `//tagname[contains(@attribute,'value1')]`

Syntax:

```
//input[contains(@id,"")]
//input[contains(@name,"")]
//a[contains(@href,"")]
//img[contains(@src,"")]
//div[contains(@id,"")]
```

Using starts-with method Syntax: `//tagname[starts-with(@attribute-name,"")]`

```
//id[starts-with(@id,"")]  
//a[starts-with(@href="")]  
//img[starts-with(@src="")]  
//div[starts-with(@id="")]  
//input[starts-with(@id="")]  
//button[starts-with(@id,"")]
```

Using following method Syntax: Xpath/following::again-ur-regular-path

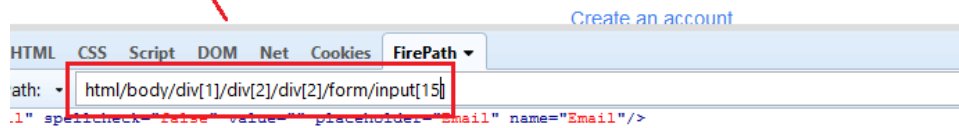
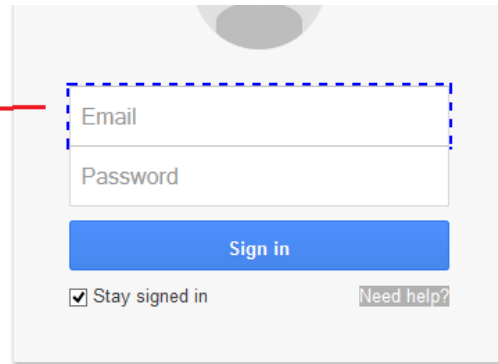
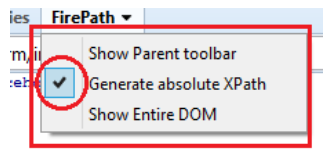
```
//input[@id='']/following::input[1]  
//a[@href='']/following::a[1]  
//img[@src='']/following::img[1]
```

Using preceding method Syntax: Xpath/preceding::again-ur-regular-path

```
//input[@id='']/preceding::input[1]  
//a[@href='']/preceding::a[1]  
//img[@src='']/preceding::img[1]
```

Absolute Xpath Method

1-/html/head/body/div/input



Relative and Absolute Xpath Method

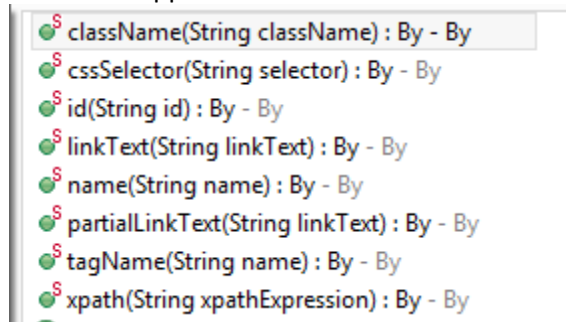
//parent-xpath/absolute xpath

//input[@id='section']/div/input

CSS Selector

Selenium supports multiple locators which will help you to identify web element and perform the operation based on your requirements.

Selenium supports 8 locators



This is one of the most frequently asked questions in [interview](#) like how many selectors are present and which one is best and where to use which locator?

In terms of performance, CSS perform well as compared to [XPATH](#) and CSS will not change based on browsers, that is it will behave same in all browsers but xpath will behave differently in IE browser.

Symbol used while writing CSS selector in Selenium Webdriver

	attribute	Symbol used
1	Using id	use # symbol
2	Using class name	use . symbol
3	Using attribute	tagname[attribute='value']
4	Using multiple attribute	tagname[attribute1='value1'] [attribute2='value2']
5	Contains	* symbol
6	Starts with	^ symbol
7	Ends with	\$ symbol

Find CSS Selector using Single Attribute

Example: `input[id='user_login']`

Syntax: `tagname[attribute='value']`

The image illustrates how to find a CSS selector for a specific element on a web page. It shows a login form with a 'Username' input field, a 'Password' input field, a 'Remember Me' checkbox, and a 'Log In' button. Below the form, there are links for 'Lost your password?' and 'Back to WordPress Demo Install'.

Below the form, the Firebug DOM view is shown. The 'CSS' tab is selected, and the 'highlight' section displays the CSS selector `input[id='user_login']`, which is highlighted with a red box. A red arrow points from this selector to the 'Username' input field in the form above. The DOM tree shows the following structure:

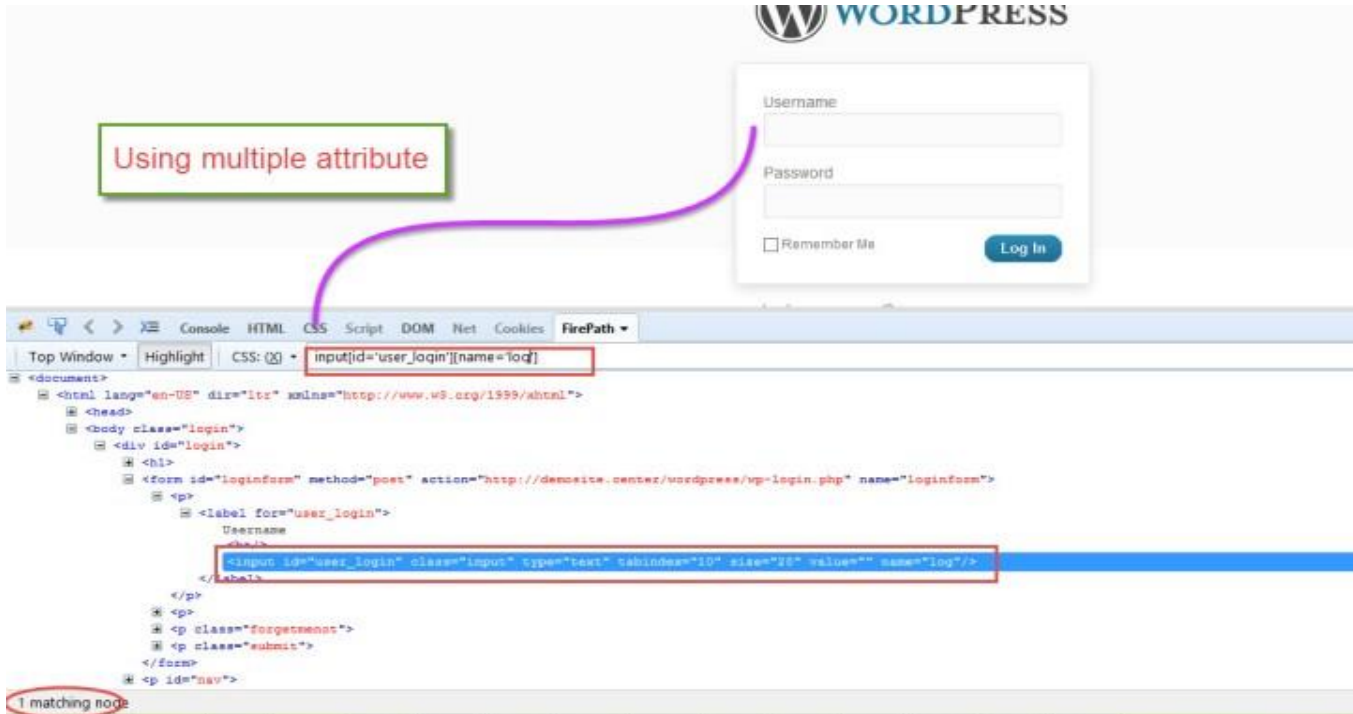
```
<form id="loginform" method="post" action="http://demosite.center/wordpress/wp-login.php" name="loginform">
  <p>
    <label for="user_login">
      Username
      <br/>
      <input id="user_login" class="input" type="text" tabindex="10" size="20" value="" name="log"/>
    </label>
  </p>
```

Numbered annotations are present: '1' points to the 'CSS' tab in the Firebug interface; '2' points to the `<input id="user_login" ...>` line in the DOM tree; and '3' points to the 'Username' input field in the form.

Find CSS using Multiple Attributes

Syntax: tagname[attribute1='value1'] [attribute2='value2']

Using multiple attribute



The screenshot shows a WordPress login form with fields for Username, Password, and a Remember Me checkbox, along with a Log In button. Below the form, the FirePath tool is open, displaying the CSS selector `input[id='user_login'] [name='log']` in the search bar. The tool's DOM tree shows the corresponding HTML element: `<input id='user_login' class='input' type='text' tabindex='10' size='25' value='' name='log' />`. The status bar at the bottom indicates "1 matching node".

Find CSS using id and Class name

Syntax: using ID: **tagname#id**

Syntax: using Classname: **tagname.classname**

Generally, class name will **NOT** be unique. so always try class name with some other attributes.



Find CSS using contains

Syntax: `tagname[attribute*='value']`

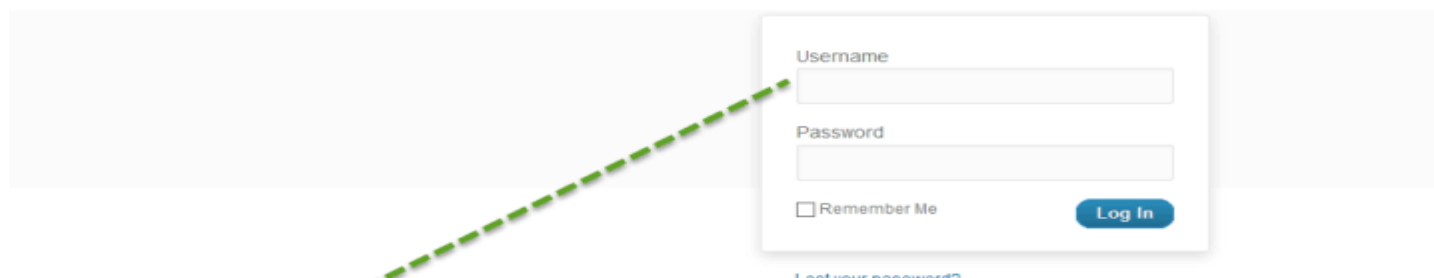
* symbol used for contains



Top Window ▾ Highlight CSS: (X) a[title*='Password Lost']

```
<document>
  <html lang="en-US" dir="ltr" xmlns="http://www.w3.org/1999/xhtml">
    <head>
      <body class="login">
        <div id="login">
          <h1>
            <form id="loginform" method="post" action="http://demo.site.center/wordpress/wp-login.php" name="loginform">
              <p id="nav">
                <a title="Password Lost and Found" href="http://demo.site.center/wordpress/wp-login.php?action=lostpassword">Lost your password?</a>
              </p>
              <script type="text/javascript" async="" defer="" src="//piwik.usinter.com/piwik.js"/>
              <script type="text/javascript"> function wp_attempt_focus() { setTimeout( function() { try { d = document.getElementById('user_login'); d.focus();
```

1 matching node



Top Window ▾ Highlight CSS: (X) input[id*='login']

```
<document>
  <html lang="en-US" dir="ltr" xmlns="http://www.w3.org/1999/xhtml">
    <head>
      <body class="login">
        <div id="login">
          <h1>
            <form id="loginform" method="post" action="http://demo.site.center/wordpress/wp-login.php" name="loginform">
              <p>
                <label for="user_login">
                  Username
                  <br/>
                  <input id="user_login" class="input" type="text" tabindex="10" size="20" value="" name="log"/>
                </label>
              </p>
```

1 matching node

Find CSS using Start-with

Syntax: `tagname[attribute^='value']`

^ symbol used for
start-with method

☐ Remember Me

Log In

[Lost your password?](#)

[← Back to WordPress Demo Install](#)

Top Window ▾ Highlight CSS: `a[title^='Are you']` FirePath ▾

```
<form id="loginform" method="post" action="http://demo.site.center/wordpress/wp-login.php" name="loginform">
<p id="nav">
<script type="text/javascript" async="" defer="" src="//piwik.usinternet.com/piwik.js"/>
<script type="text/javascript"> function wp_attempt_focus(){ setTimeout( function(){ try{ d = document.getElementById('user_login
), 200); } wp_attempt_focus(); if(typeof wpOnload=='function')wpOnload(); </script>
<p id="backtoblog">
<a title="Are you lost?" href="http://demo.site.center/wordpress/">← Back to WordPress Demo Install</a>
</p>
</div>
<div class="clear">
<script src="http://piwik.usinternet.com/scripts/demosite.js" type="text/javascript"/>
<br/>
<script src="/includes/js/bframe.jsp" type="text/javascript"/>
</div>
<script type="text/javascript" src="http://demo.site.center-3080/demo.site.com.js"></script> </div>
```

1 matching node

Find CSS using ends-with

Syntax: `tagname[attribute$='value']`

\$ Symbol will act as End with method

Username
admin

Password
●●●●●●●●

☐ Remember Me

[Lost your password?](#)

[Back to WordPress Demo Install](#)

Top Window ▾ Highlight CSS: (X) a[title\$='lost?']

```
<document>
  <html lang="en-US" dir="ltr" xmlns="http://www.w3.org/1999/xhtml">
    <head>
    <body class="login">
      <div id="login">
        <h1>
        <form id="loginform" method="post" action="http://demosite.center/wordpress/wp-login.php" name="loginform">
        <p id="msg">
          <script type="text/javascript" async="" defer="" src="//piwik.uszinter.com/piwik.js"/>
          <script type="text/javascript"> function wp_attempt_focus(){ setTimeout( function(){ try{ d = document.getElementById('
          ), 200); } wp_attempt_focus(); if(typeof wpOnload=='function')wpOnload(); </script>
        <div id="backtoblog">
          <a title="Are you lost?" href="http://demosite.center/wordpress/">Back to WordPress Demo Install</a>
        </p>
      </div>
      <div class="clear"/>
      <script src="http://piwik.uszinter.com/scripts/demosite.js" type="text/javascript"/>
      <br/>
    </body>
  </html>
```

1 matching node

Check this link from [SauceLabs for CSS Selector](#)

You can always use XPATH and CSS in your script, depends on your requirement.

If you want to run your test script in all browsers (For Cross Browser testing) then go with CSS.
CSS will generally stay the same across browsers but xpath may not.