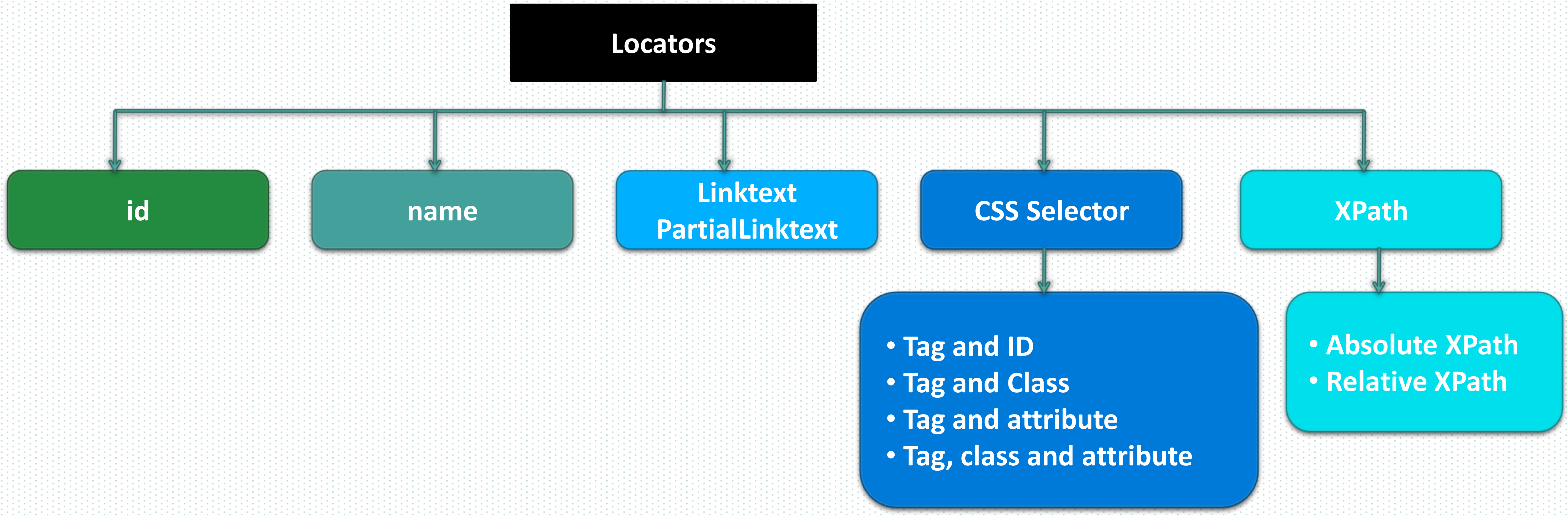
Selenium WebDriver

Agenda

- Locators in WebDriver

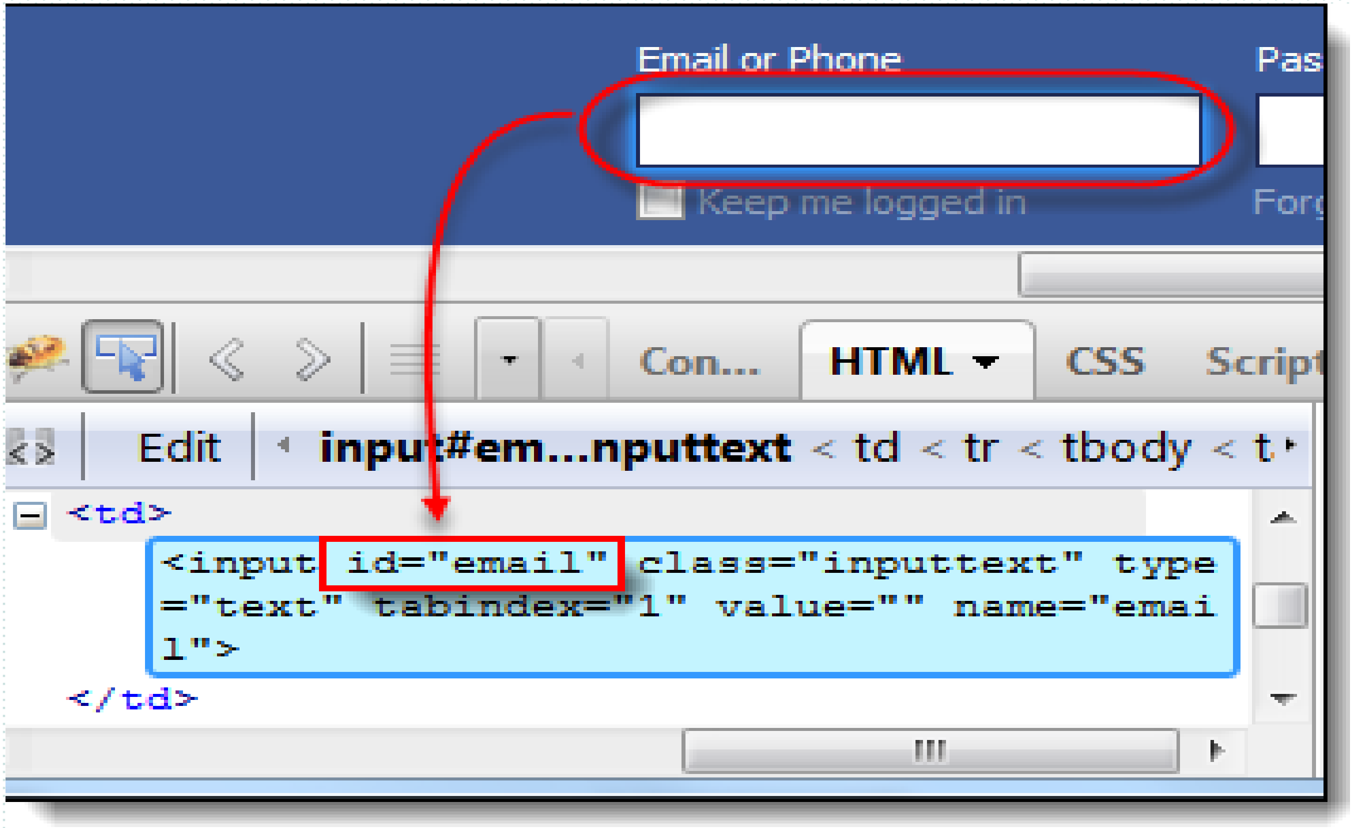
Locators

- We can identify various elements on the web using **Locators**.
- Locators are addresses that identify a web element uniquely within the page.



Locating by ID

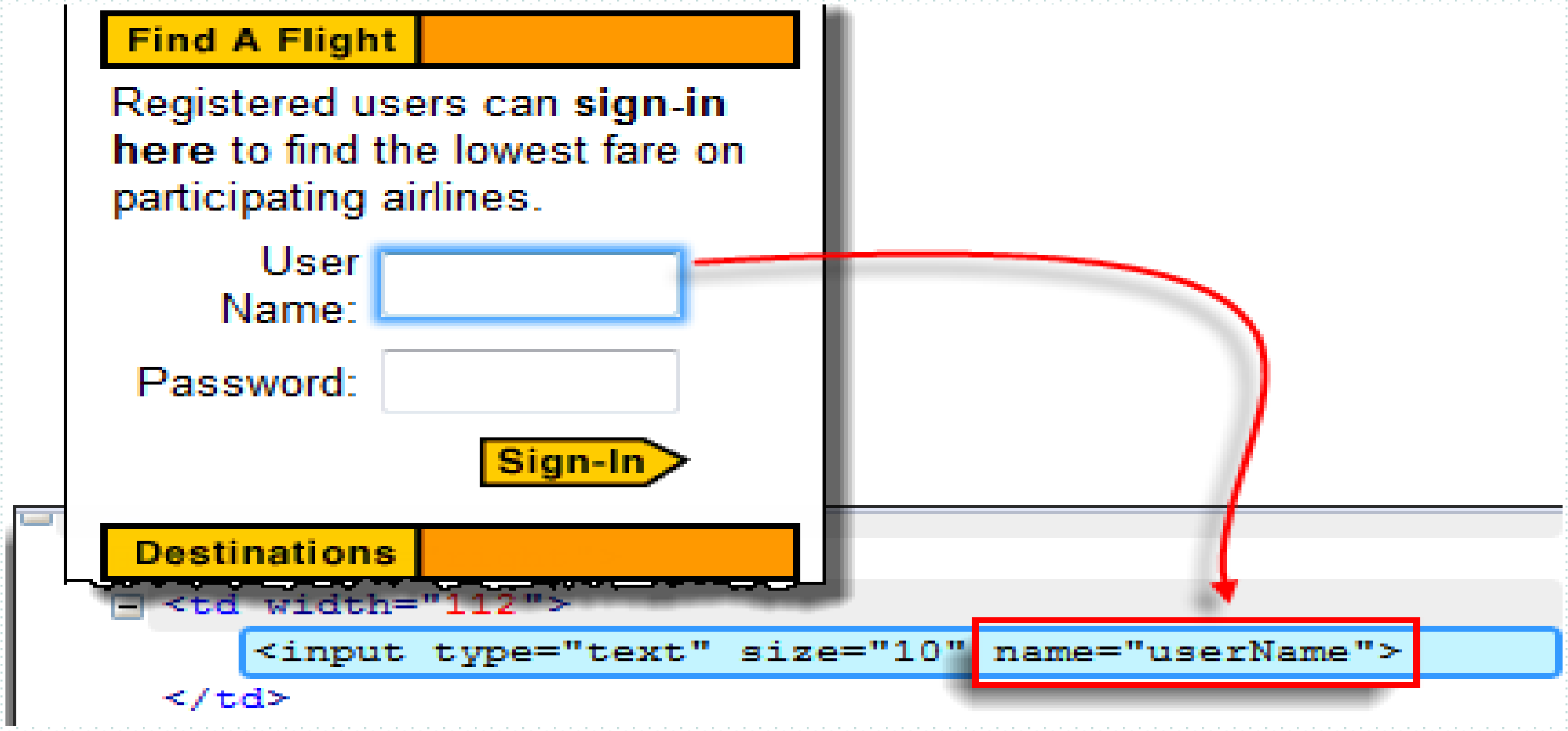
```
driver.findElement(By.id("email")).sendKeys("xxxxx@gmail.com");
```



```
<td>
    <input id="email" class="inputtext" type
    ="text" tabindex="1" value="" name="emai
    l">
</td>
```

Locating by the name

```
driver.findElement(By.name("userName")).sendKeys("mercury");
```



Registered users can **sign-in here** to find the lowest fare on participating airlines.

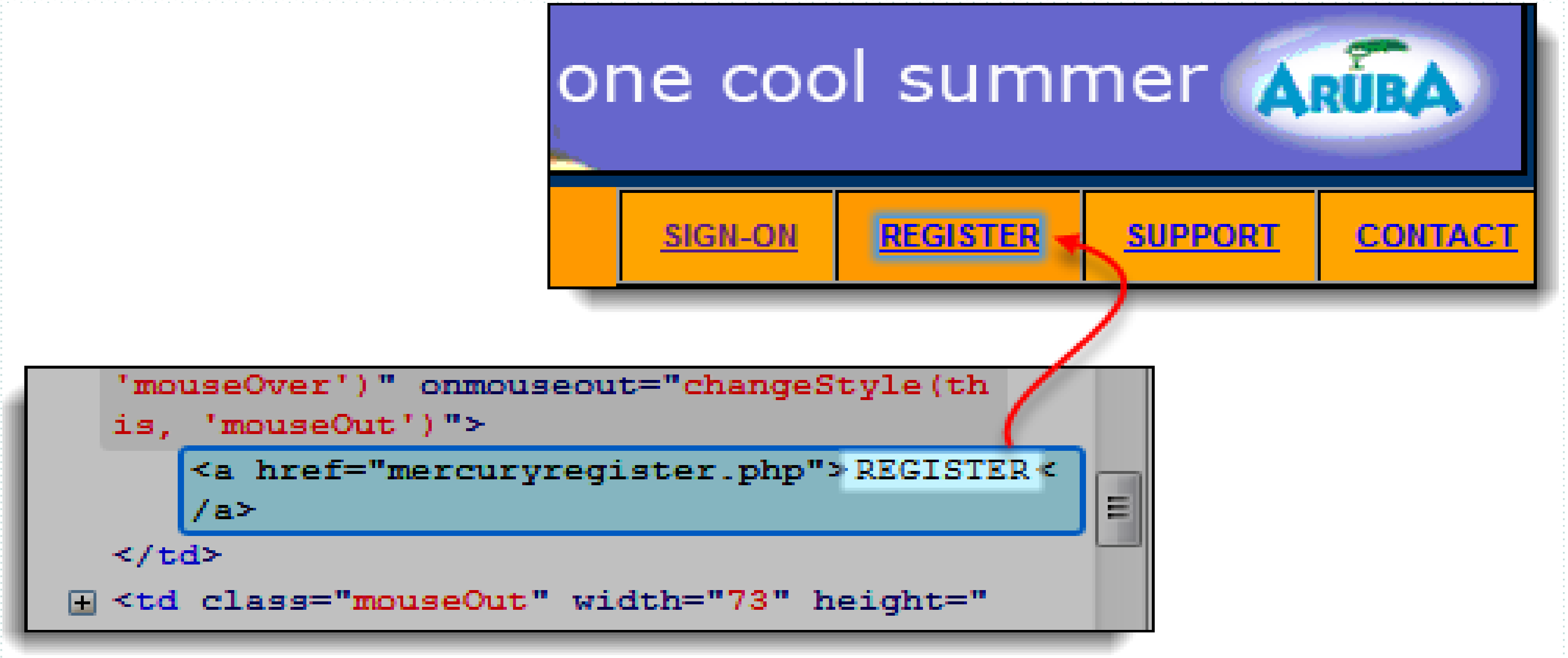
User Name:

Password:

```
<td width="112">
    <input type="text" size="10" name="userName">
</td>
```

Locating by the Link Text

```
driver.findElement(By.linkText("REGISTER")).click();
```



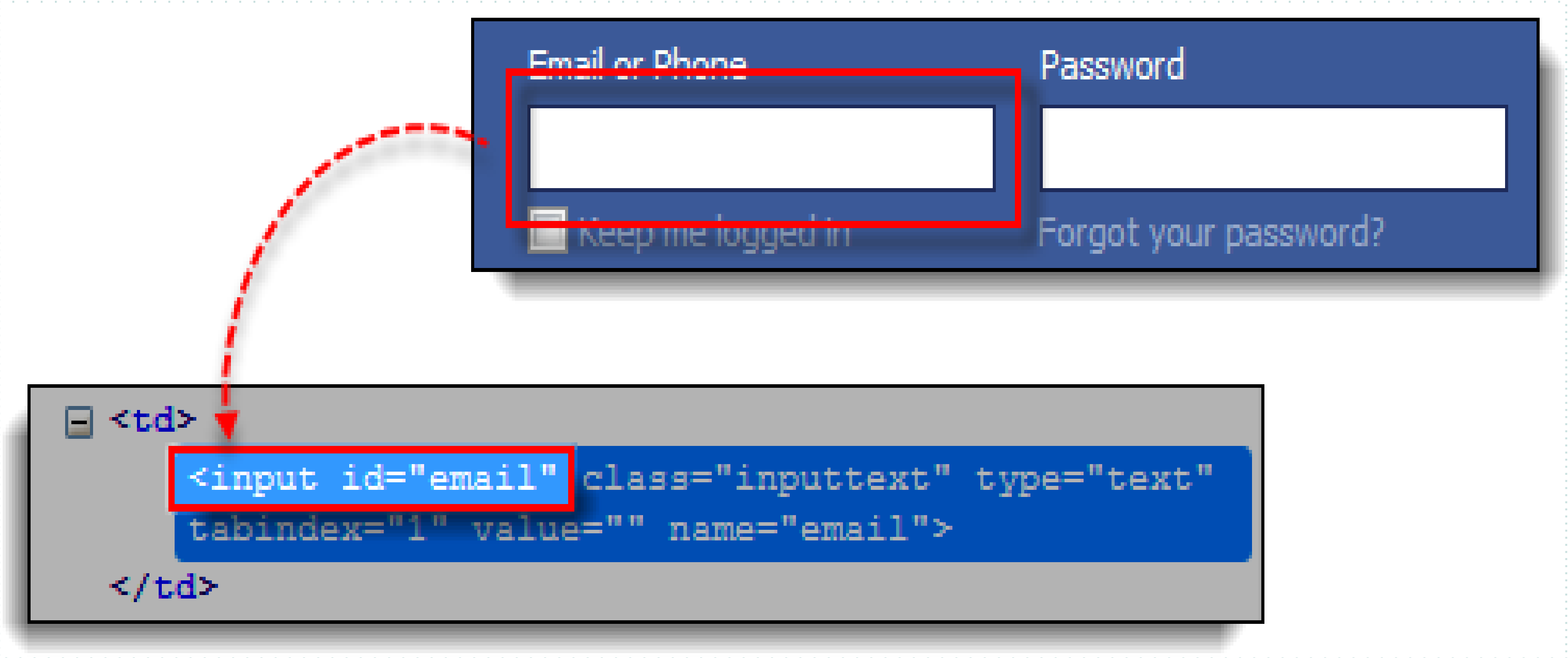
```
'mouseOver')" onmouseout="changeStyle(th
is, 'mouseOut')">
    <a href="mercuryregister.php">REGISTER<
    /a>
</td>
<td class="mouseOut" width="73" height="
```

CSS Selector - Cascading Style Sheets

- Tag and ID
- Tag and Class
- Tag and attribute
- Tag, class and attribute

CSS Selector – *Tag* and *ID*

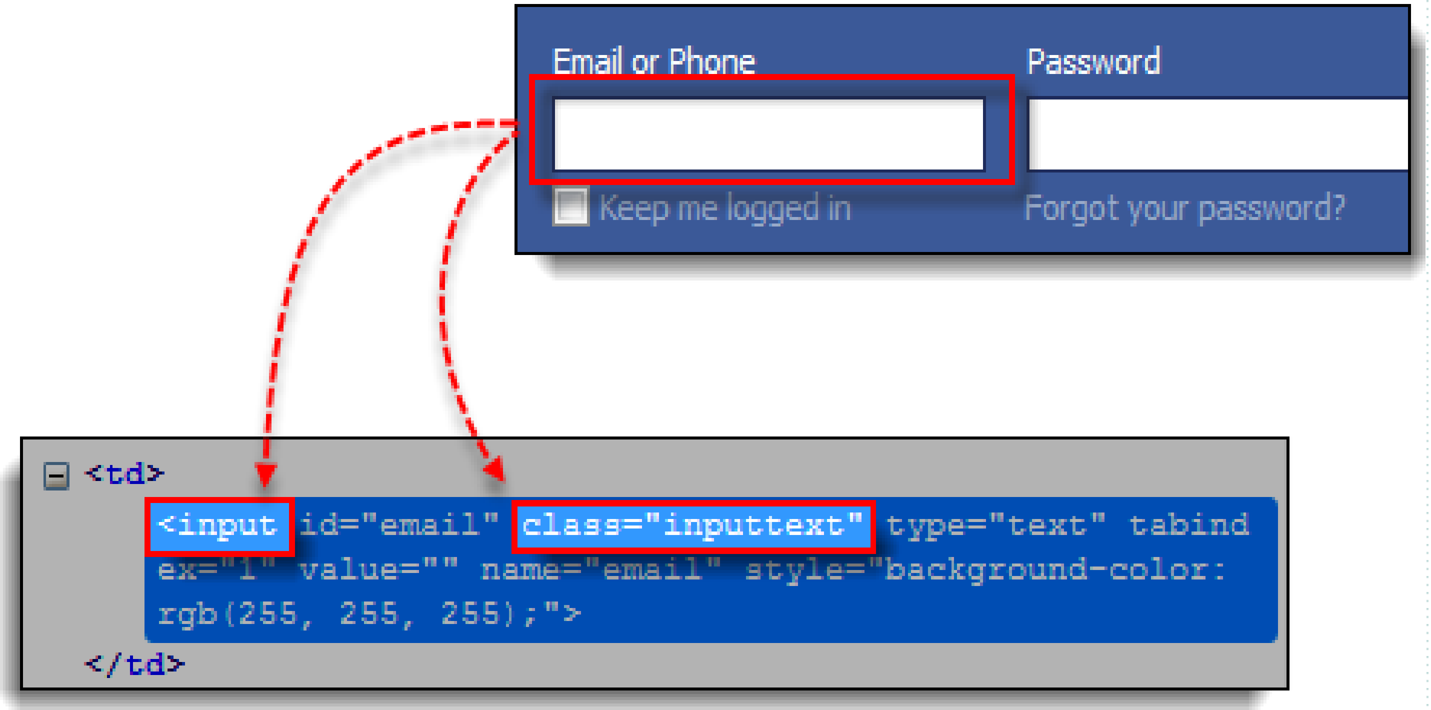
```
driver.findElement(By.cssSelector("input#email")).sendkeys("xxxxxxx@gmail.com");
```



```
<td>
    <input id="email" class="inputtext" type="text"
    tabindex="1" value="" name="email">
</td>
```

CSS Selector – *Tag* and *Class*

```
driver.findElement(By.cssSelector("input.inputtext")).sendkeys("xxxxxx@gmail.com");
```

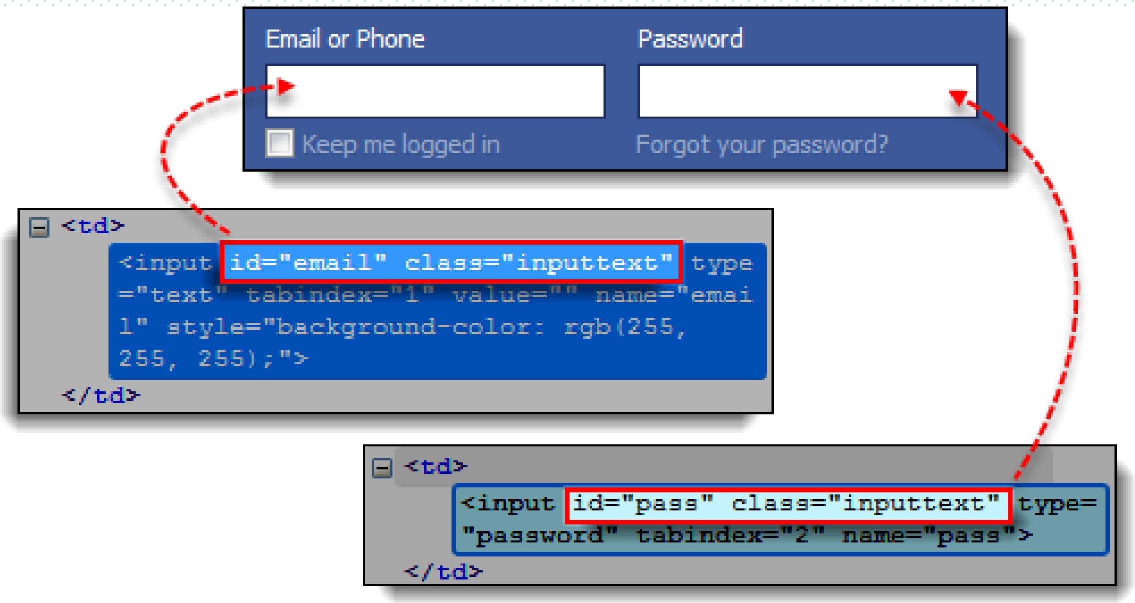


```
<td>
    <input id="email" class="inputtext" type="text" tabindex="1" value="" name="email" style="background-color: rgb(255, 255, 255);">
</td>
```

Multiple elements have the same tag and class name

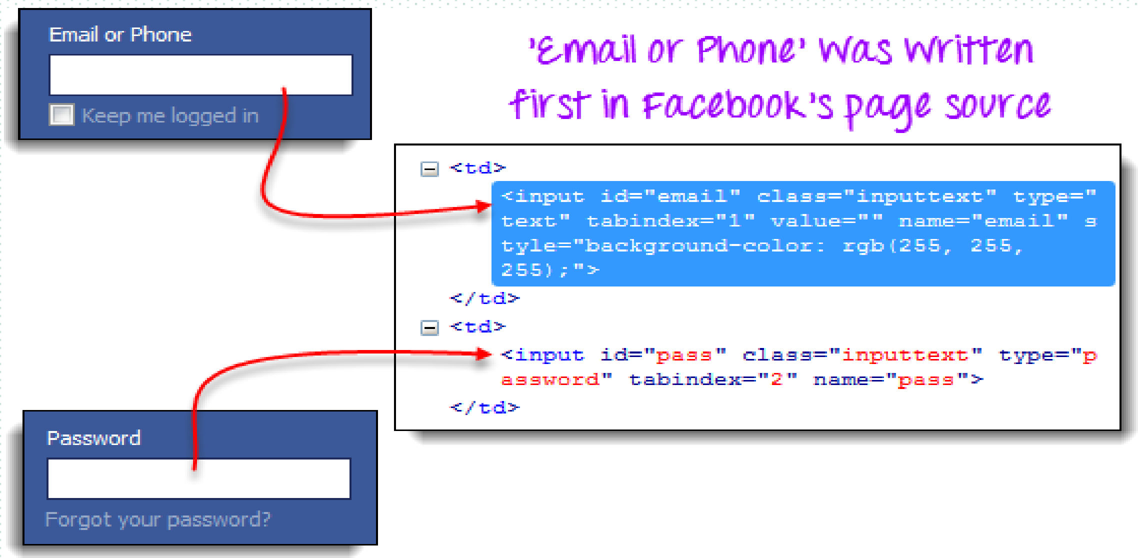
- when multiple elements have the same HTML tag and name, only the first element in source code will be recognized.

```
driver.findElement(By.cssSelector("input.inputtext")).sendkeys("xxxxxxxx@gmail.com");
```



```
<td>
    <input id="email" class="inputtext" type="text" tabindex="1" value="" name="email" style="background-color: rgb(255, 255, 255);">
</td>
```

```
<td>
    <input id="pass" class="inputtext" type="password" tabindex="2" name="pass">
</td>
```

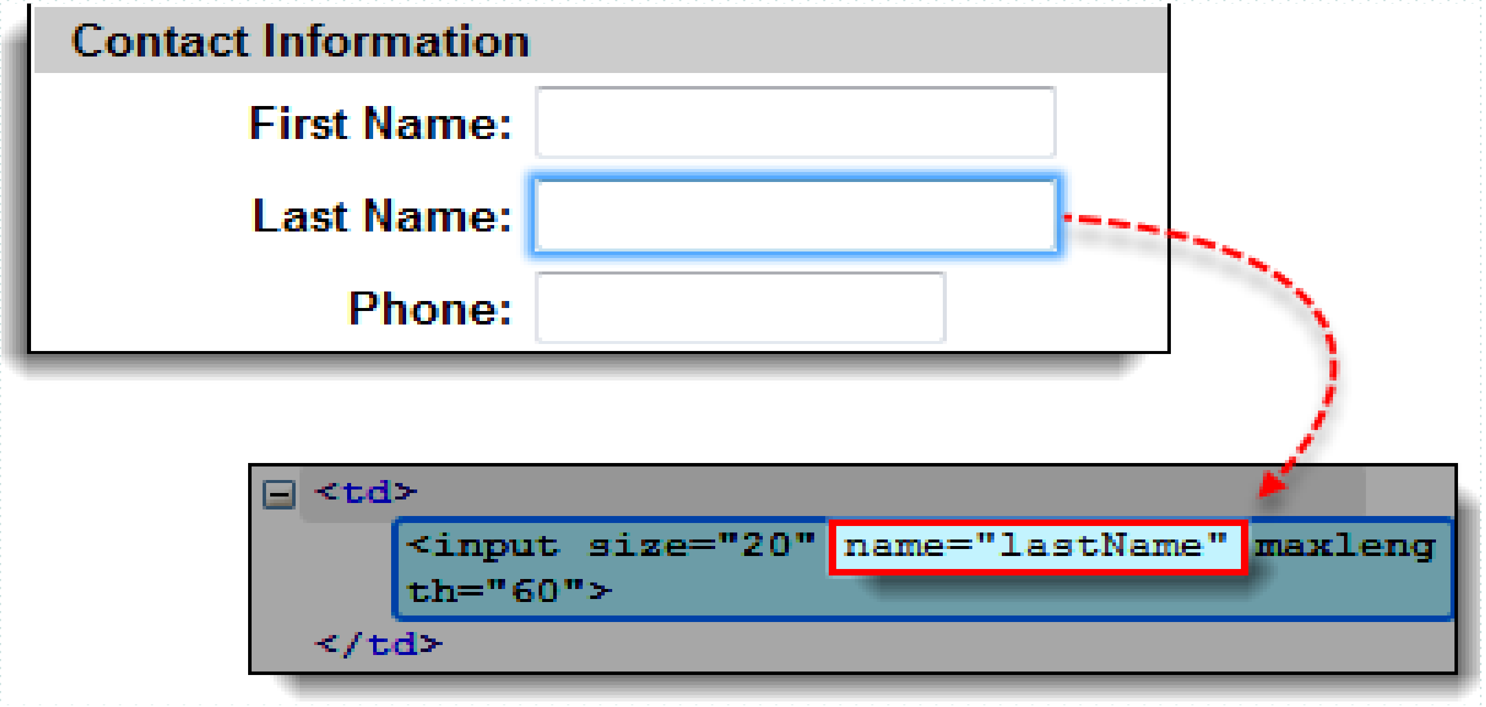


'Email or Phone' was written first in Facebook's page source

```
<td>
    <input id="email" class="inputtext" type="text" tabindex="1" value="" name="email" style="background-color: rgb(255, 255, 255);">
</td>
<td>
    <input id="pass" class="inputtext" type="password" tabindex="2" name="pass">
</td>
```

CSS Selector – *Tag* and *Attribute*

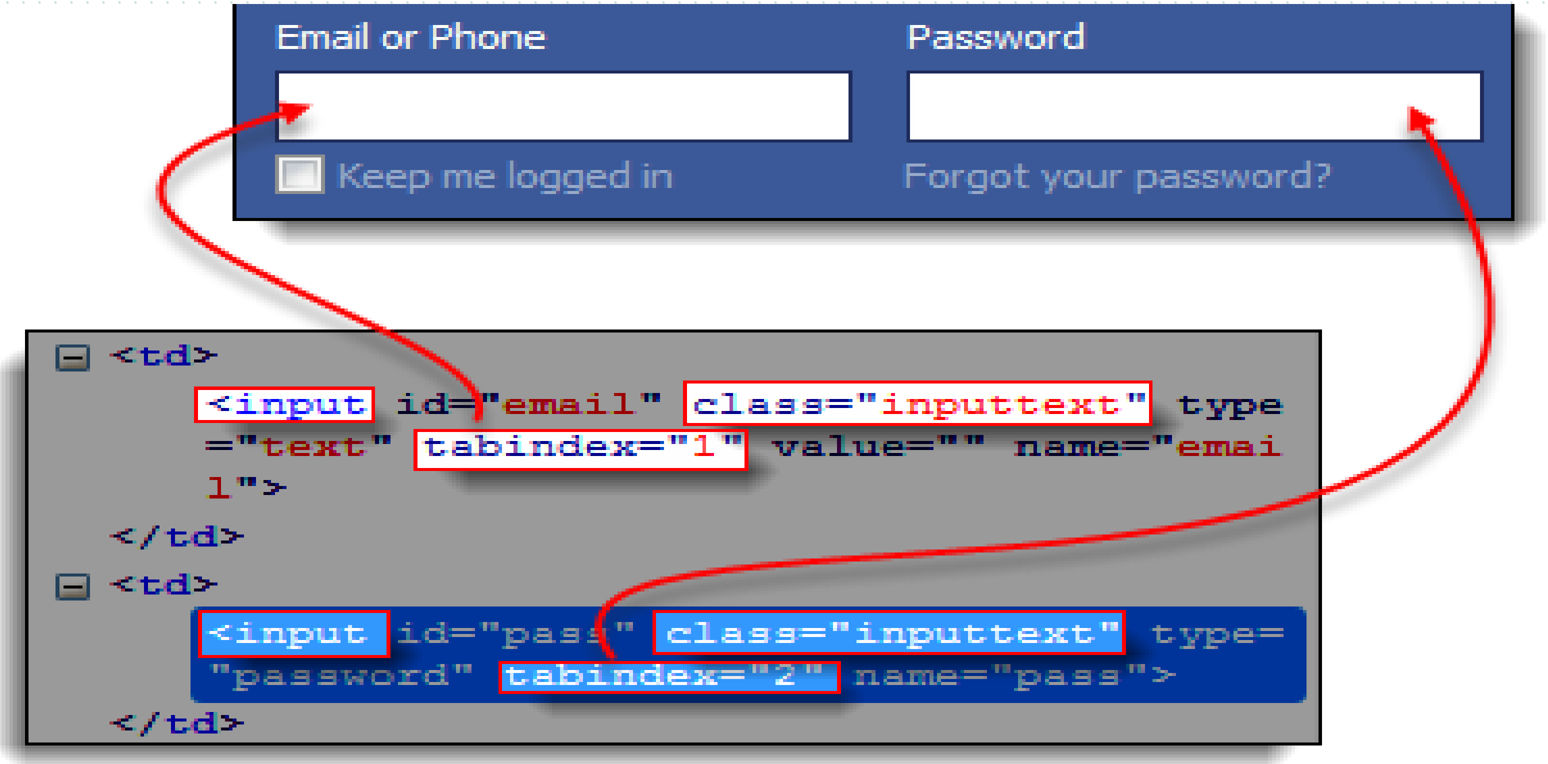
```
driver.findElement(By.cssSelector("input[name=lastName]")).sendkeys("xxxxx");
```



```
<td>
    <input size="20" name="lastName" maxlength="60">
</td>
```

CSS Selector - *Tag, class* and *attribute*

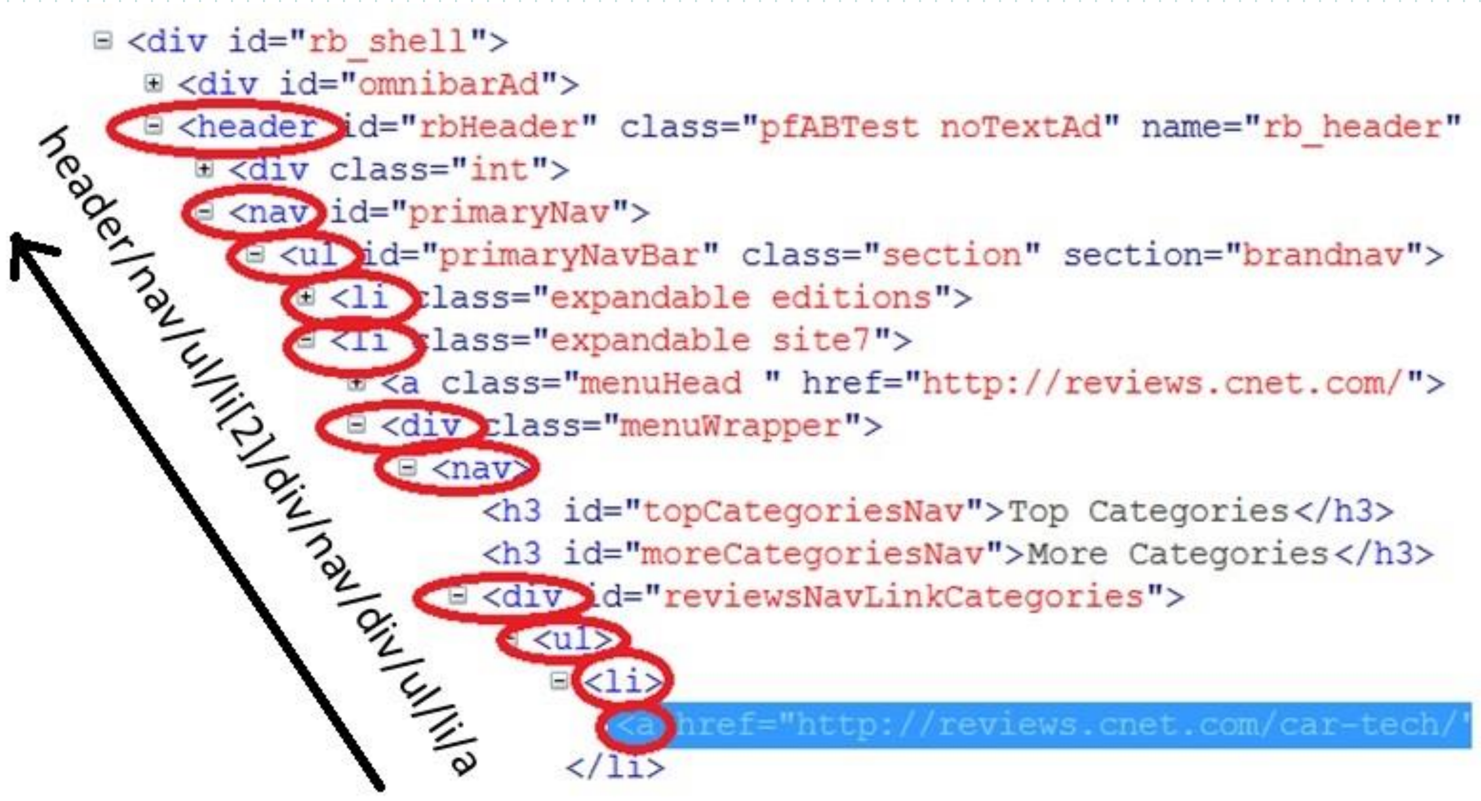
```
driver.findElement (By.cssSelector("input.inputtext[tabindex=2]").sendkeys("xxxx");
```



```
<td>
    <input id="email" class="inputtext" type
    ="text" tabindex="1" value="" name="emai
    l">
</td>
<td>
    <input id="pass" class="inputtext" type=
    "password" tabindex="2" name="pass">
</td>
```

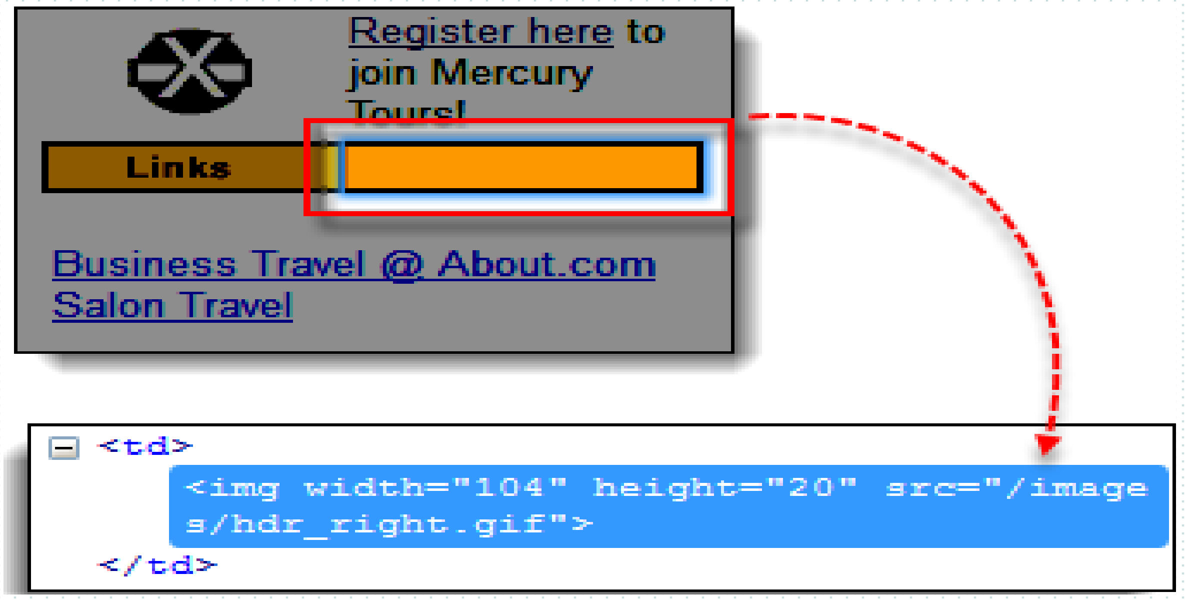
XPath

- XPath is for locating the elements or nodes in XML documents
- XML and HTML has similar syntax (HTML is a subset of XML)
- Hence XPath can be used to locate elements in HTML pages (web pages).

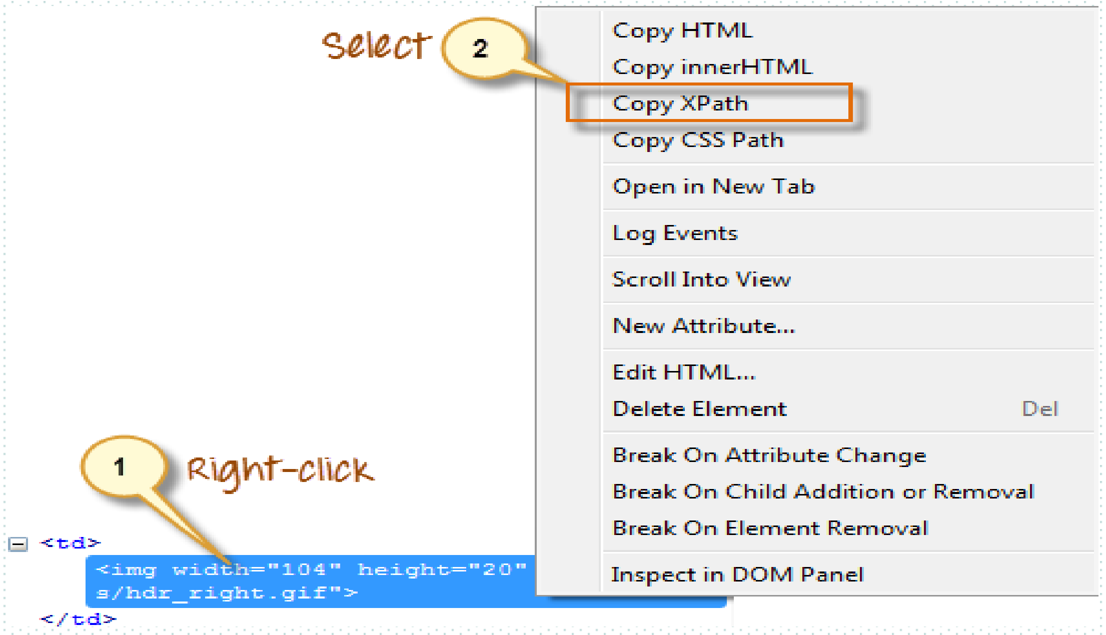


```
<div id="rb_shell">
  <div id="omnibarAd">
  <header id="rbHeader" class="pfABTest noTextAd" name="rb_header"
    <div class="int">
    <nav id="primaryNav">
      <ul id="primaryNavBar" class="section" section="brandnav">
        <li class="expandable editions">
        <li class="expandable site7">
          <a class="menuHead " href="http://reviews.cnet.com/">
          <div class="menuWrapper">
            <nav>
              <h3 id="topCategoriesNav">Top Categories</h3>
              <h3 id="moreCategoriesNav">More Categories</h3>
              <div id="reviewsNavLinkCategories">
                <ul>
                  <li>
                    <a href="http://reviews.cnet.com/car-tech/"
                  </li>
```

Capture XPath


Register here to join Mercury Tours!
Links
Business Travel @ About.com
Salon Travel

```
<td>
    <img width="104" height="20" src="/images/hdr_right.gif">
</td>
```



```
<td>
    <img width="104" height="20"
    s/hdr_right.gif">
</td>
```

Types of XPath

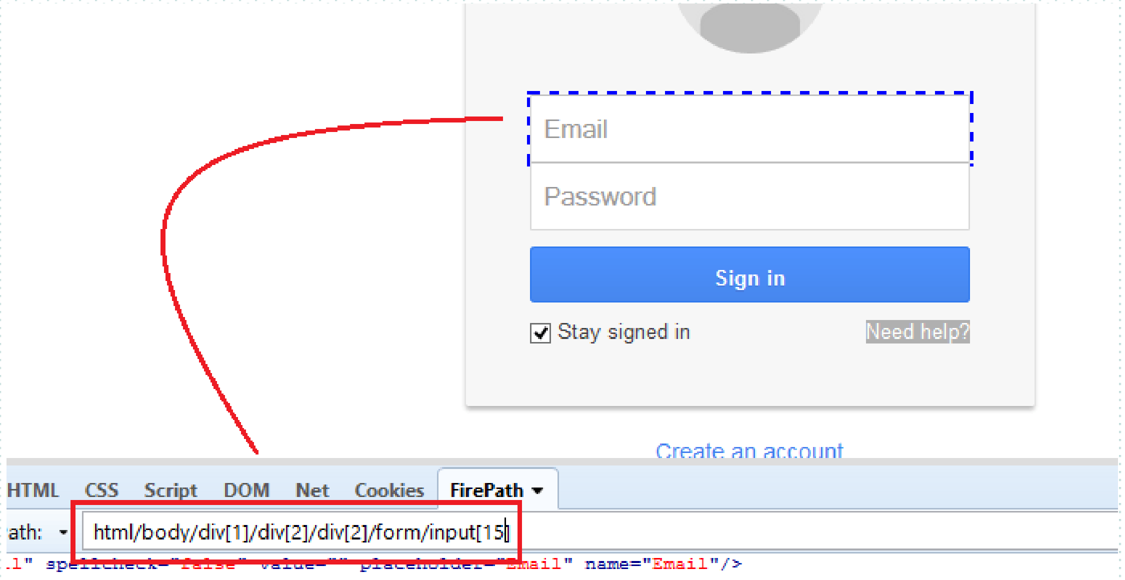
- There are two types of XPath:

1. Absolute XPath .
2. Relative XPath .

Absolute XPath

- It is the direct way to find the element, but the disadvantage of the absolute XPath is that if there are any changes made in the path of the element then that XPath gets failed.

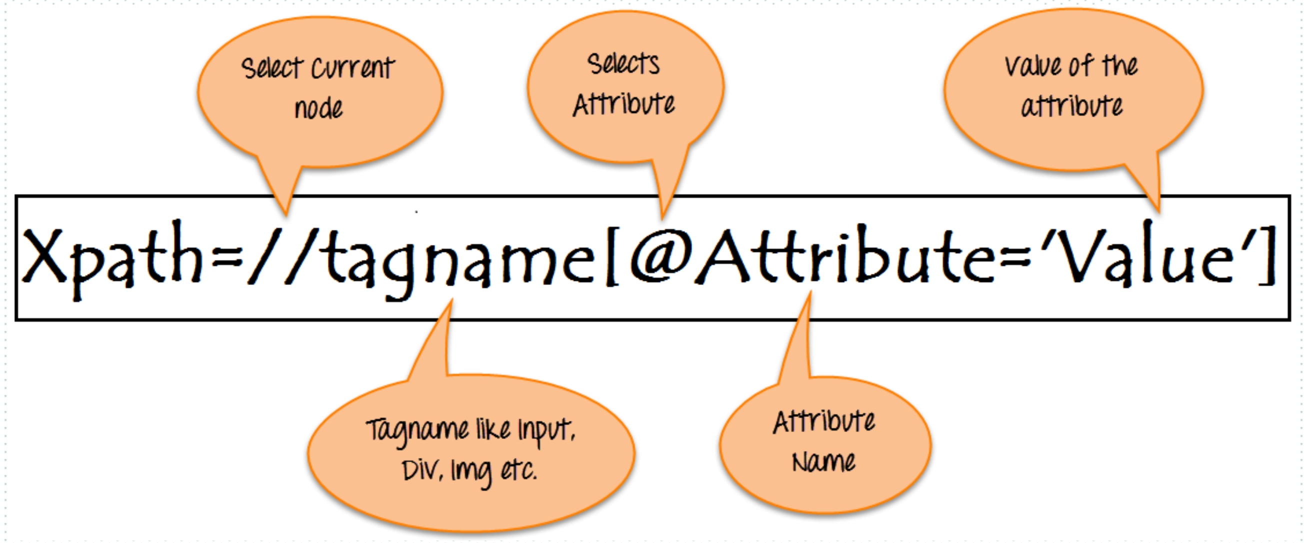
- The key characteristic of XPath is that it begins with the single forward slash(/) ,which means you can select the element from the root node.



```
html/body/div[1]/div[2]/div[2]/form/input[15]
.l" spellcheck="false" value="" placeholder="Email" name="Email"/>
```

Relative XPath

- For Relative Xpath the path starts from the middle of the HTML DOM structure. Its start with the double forward slash (//), which means it can search the element anywhere at the webpage.



Syntax for XPath

- XPath contains the path of the element situated at the web page.
- Standard syntax for creating XPath is.

```
Xpath=//tagname[@attribute='value‘]
```

// : Select current node.

Tagname: Tagname of the particular node.

@: Select attribute.

Attribute: Attribute name of the node.

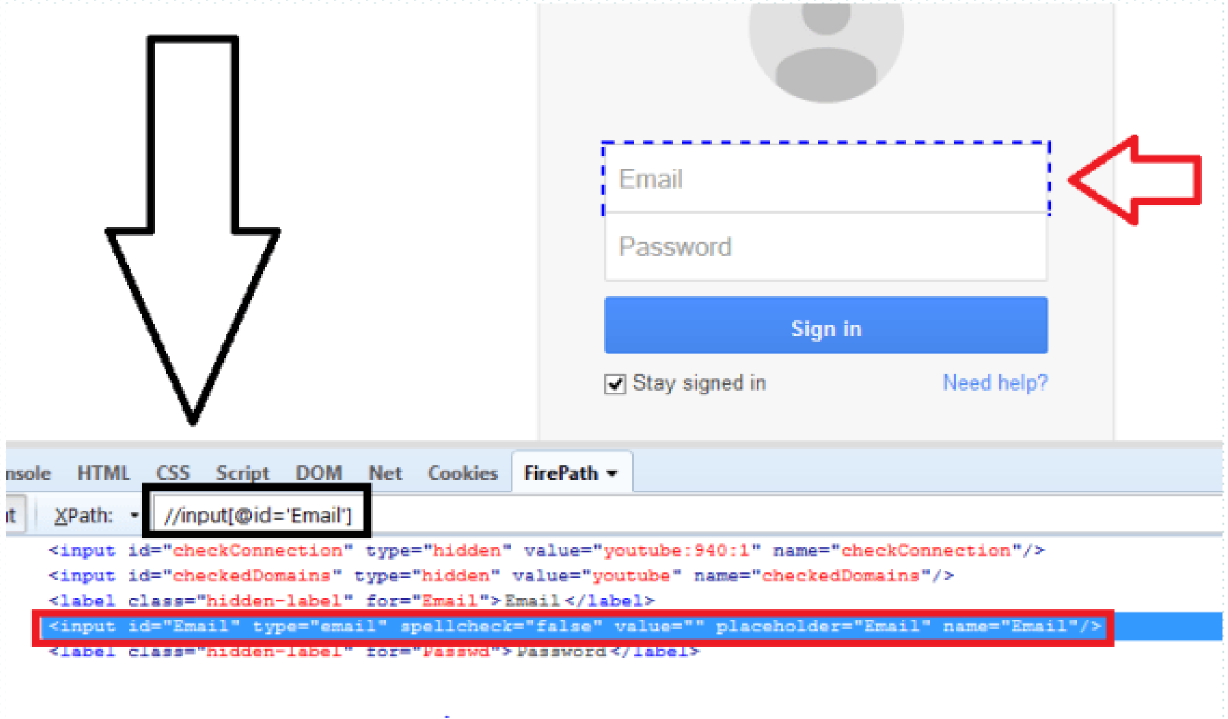
Value: Value of the attribute.

XPath with Single attribute

// tagname[@attribute-name='value1']

Examples:

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



```
//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="Passwd">Password</label>
```

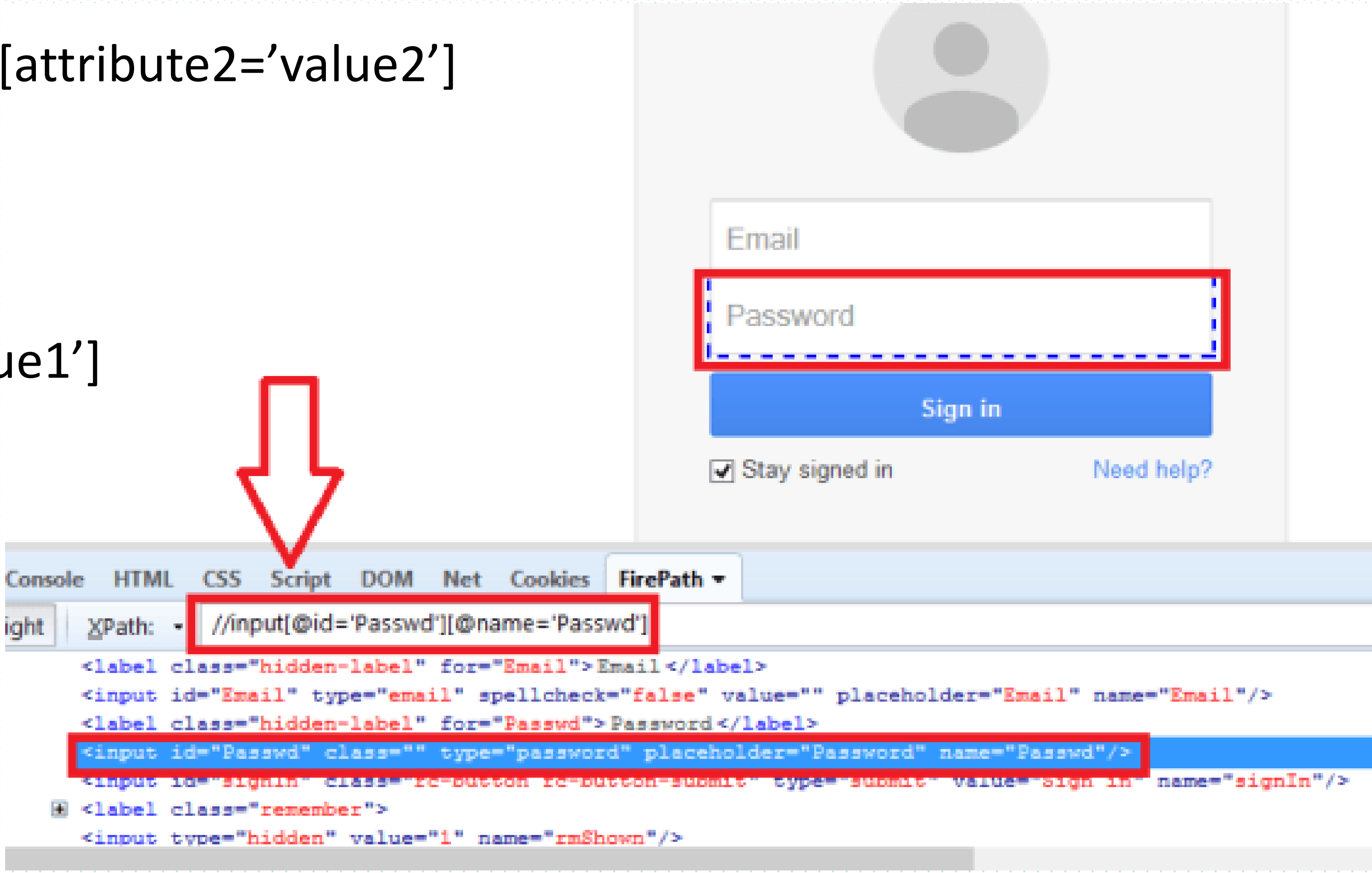
XPath with Multiple attributes

//tagname[@attribute1='value1'][attribute2='value2']

Examples:

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

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

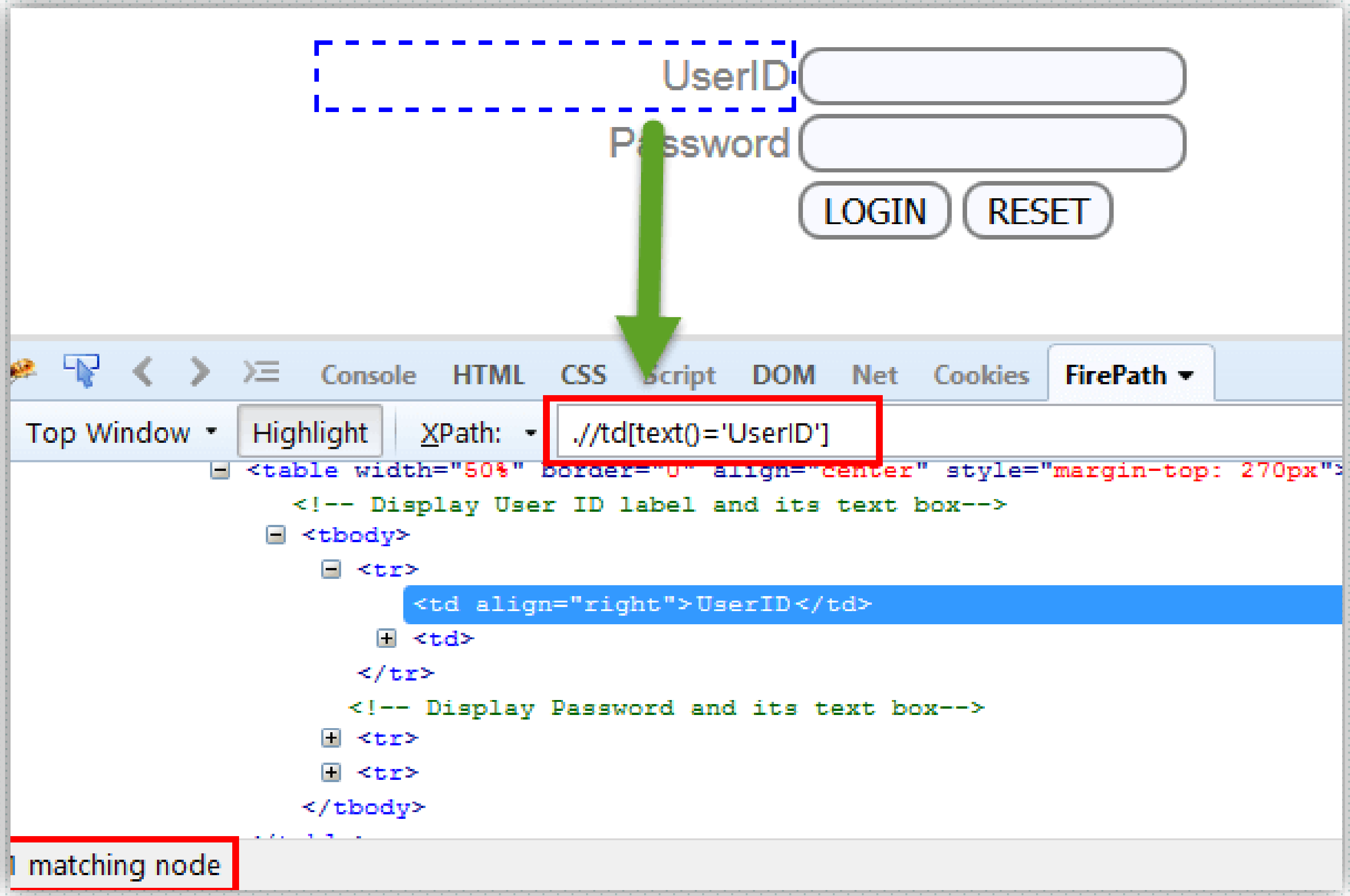


```
//input[@id='Passwd'][@name='Passwd']
```

```
<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="rc-button rc-button-submit" type="submit" value="Sign in" name="signIn"/>
<label class="remember">
<input type="hidden" value="1" name="rmShown"/>
```

XPath with text() method

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



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

```
<table width="50%" border="0" align="center" style="margin-top: 270px">
    <!-- Display User ID label and its text box-->
    <tbody>
        <tr>
            <td align="right">UserID</td>
            <td>
        </tr>
        <!-- Display Password and its text box-->
        <tr>
        <tr>
    </tbody>
```

1 matching node

XPath with Contains() method

//tagname[**contains**(@attribute,’value1’)]

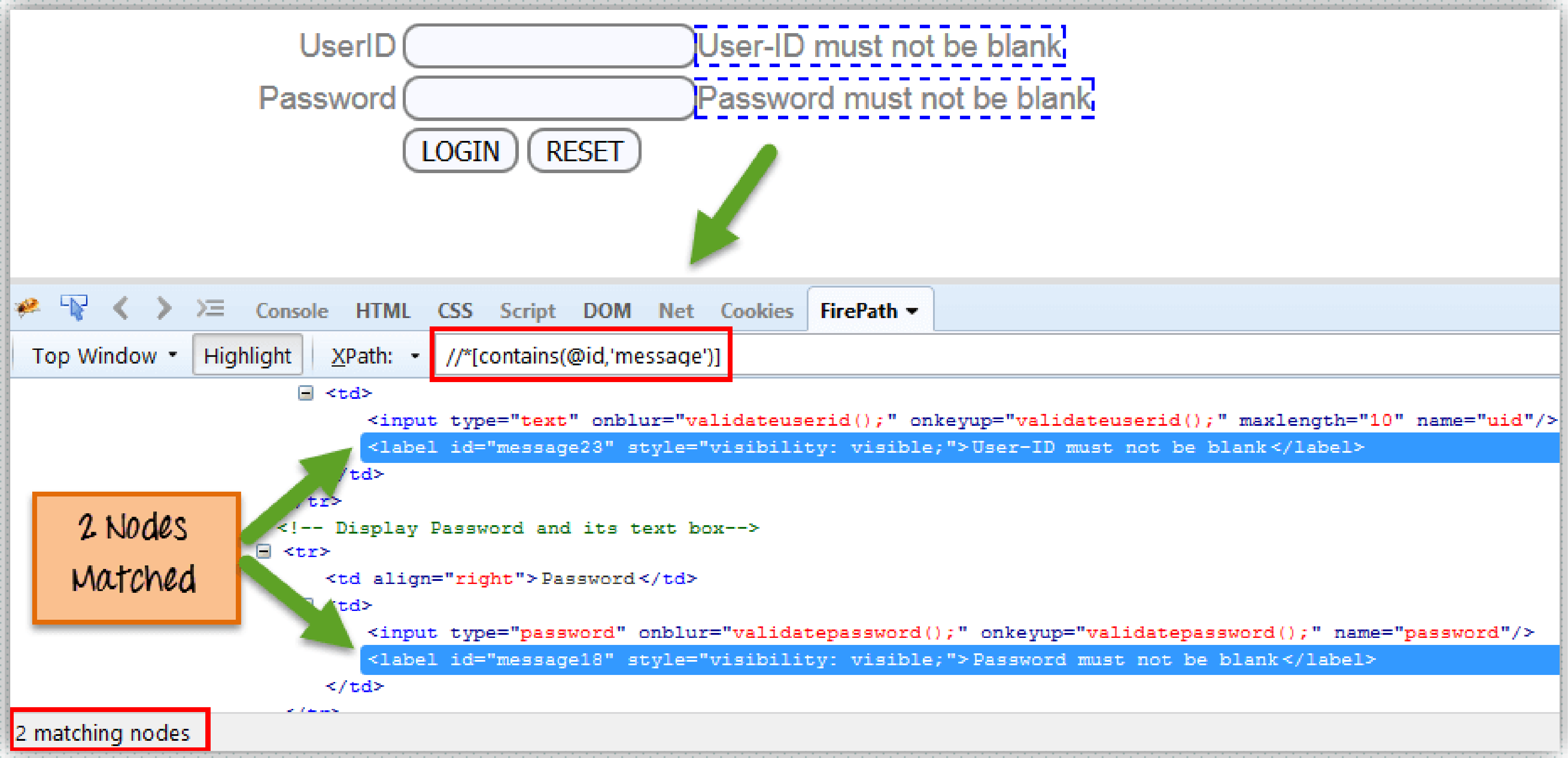
//input[contains(@id,’’)]

//input[contains(@name,’’)]

//a[contains(@href,’’)]

//img[contains(@src,’’)]

//div[contains(@id,’’)]



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

```
⊟ <td>
      <input type="text" onblur="validateuserid();" onkeyup="validateuserid();" maxlength="10" name="uid"/>
      <label id="message23" style="visibility: visible;">User-ID must not be blank</label>
    /td>
  /tr>
<!-- Display Password and its text box-->
⊟ <tr>
    <td align="right">Password</td>
    td>
      <input type="password" onblur="validatepassword();" onkeyup="validatepassword();" name="password"/>
      <label id="message18" style="visibility: visible;">Password must not be blank</label>
    </td>
```

2 matching nodes

XPath with starts-with() method

//tagname[**starts-with**(@attribute-name,'')]

Examples:

//id[starts-with(@id,'')]

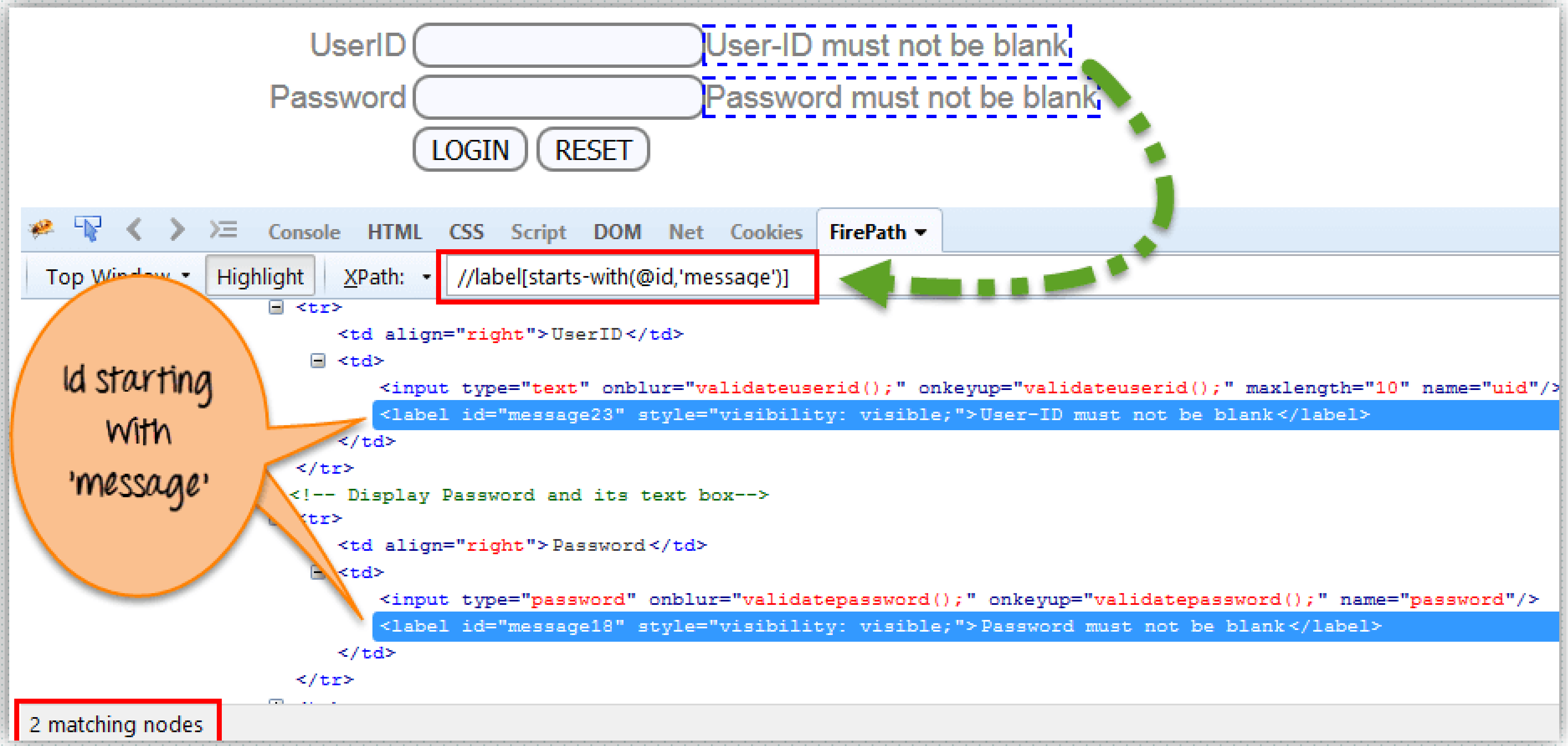
//a[starts-with(@href='')]

//img[starts-with(@src='')]

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

//input[starts-with(@id='')]

//button[starts-with(@id,'')]

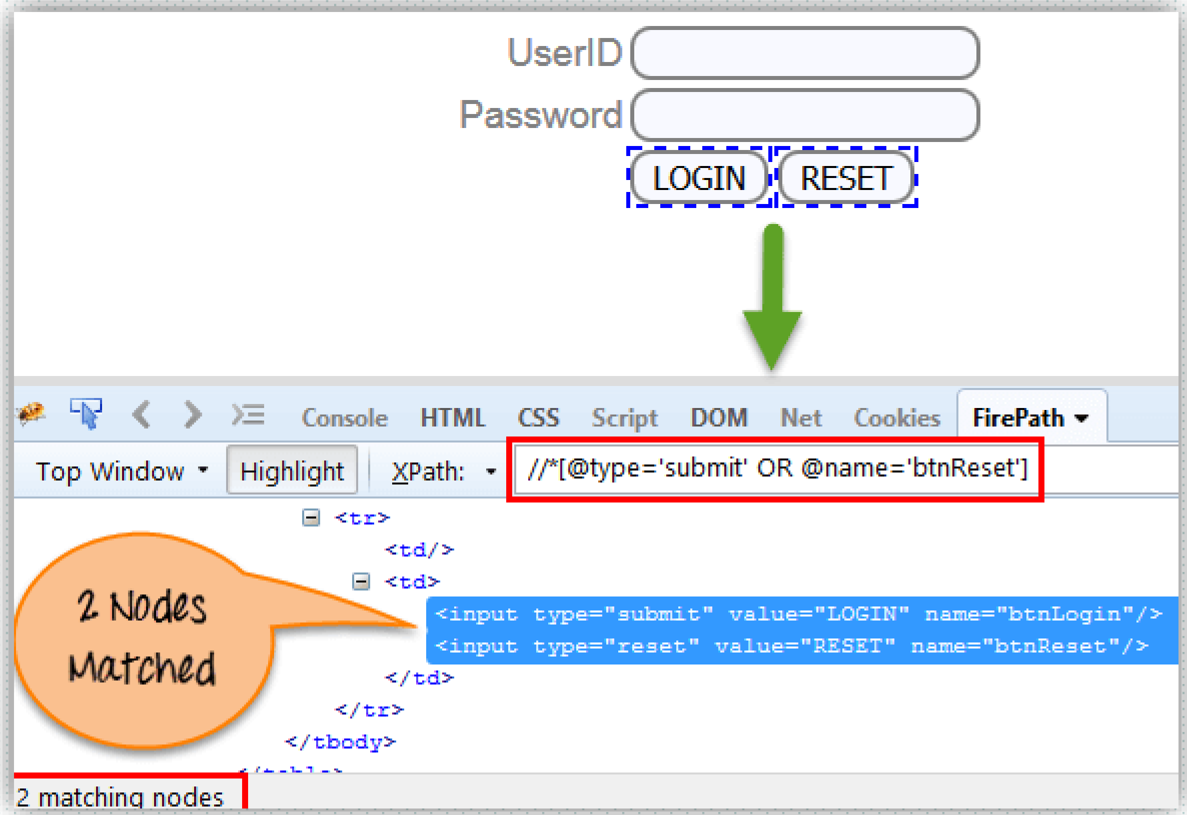


```
//label[starts-with(@id,'message')]
```

```
<tr>
    <td align="right">UserID</td>
  <td>
      <input type="text" onblur="validateuserid();" onkeyup="validateuserid();" maxlength="10" name="uid"/>
      <label id="message23" style="visibility: visible;">User-ID must not be blank</label>
    </td>
</tr>
<!-- Display Password and its text box-->
<tr>
    <td align="right">Password</td>
  <td>
      <input type="password" onblur="validatepassword();" onkeyup="validatepassword();" name="password"/>
      <label id="message18" style="visibility: visible;">Password must not be blank</label>
    </td>
</tr>
```

Using OR, AND with XPath

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

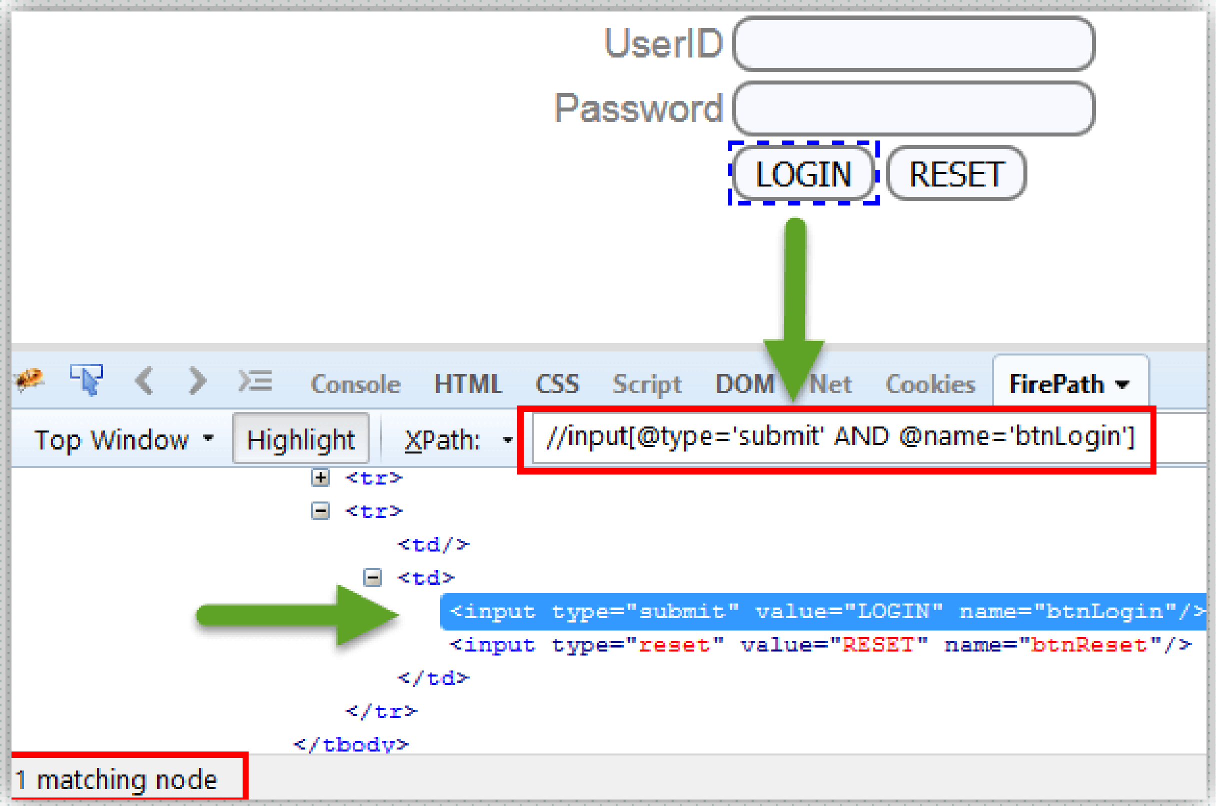


```
//*[@type='submit' OR @name='btnReset']
```

```
<tr>
    <td/>
    <td>
        <input type="submit" value="LOGIN" name="btnLogin"/>
        <input type="reset" value="RESET" name="btnReset"/>
    </td>
</tr>
</tbody>
```

2 matching nodes

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



```
//input[@type='submit' AND @name='btnLogin']
```

```
<tr>
<tr>
    <td/>
    <td>
        <input type="submit" value="LOGIN" name="btnLogin"/>
        <input type="reset" value="RESET" name="btnReset"/>
    </td>
</tr>
</tbody>
```

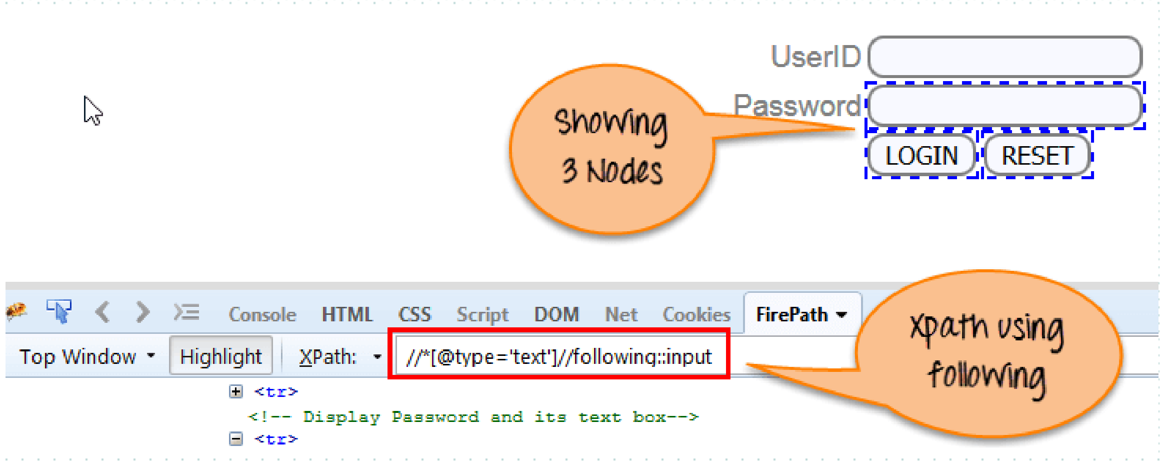
1 matching node

Following

➢ Selects all elements in the document of the current node

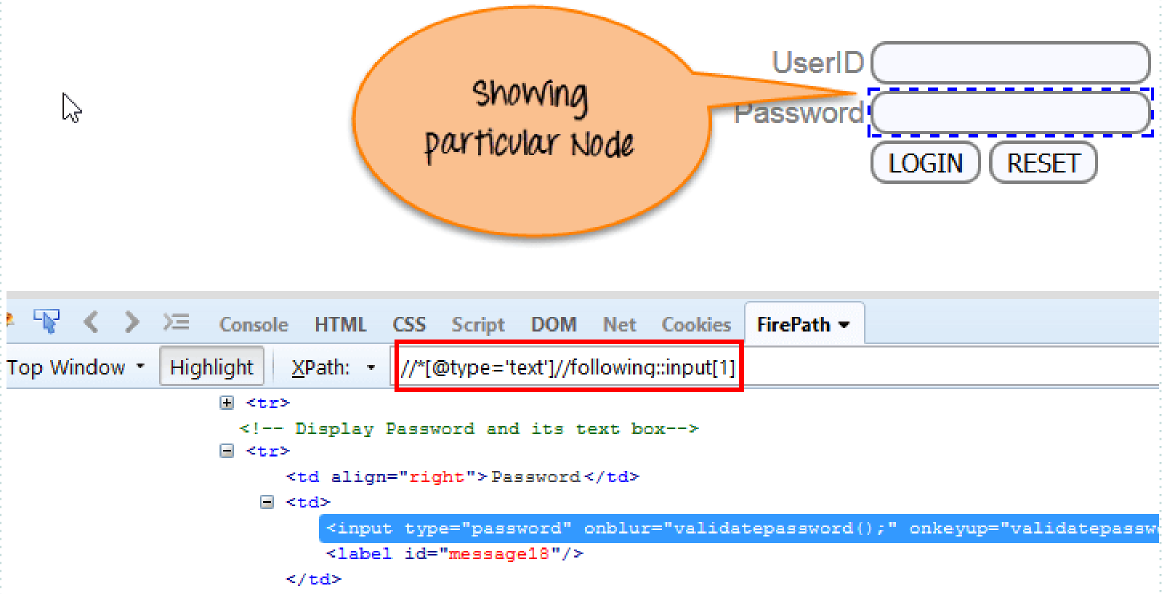
//*[@type='text']//following::input

//*[@type='text']//following::input[1]



```
//*[@type='text']//following::input
```

```
<tr>
<!-- Display Password and its text box-->
<tr>
```



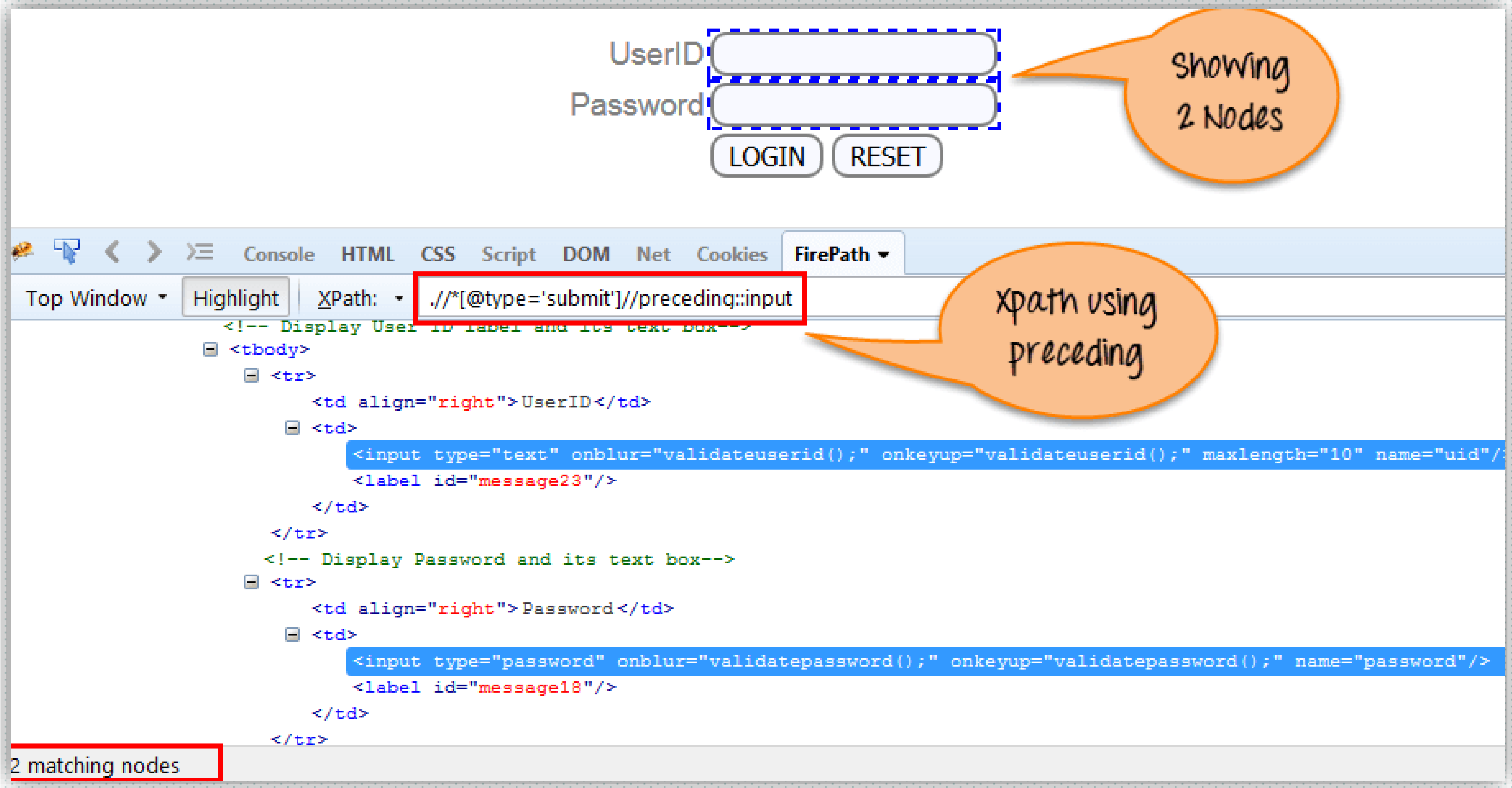
```
//*[@type='text']//following::input[1]
```

```
<tr>
<!-- Display Password and its text box-->
<tr>
  <td align="right">Password</td>
  <td>
    <input type="password" onblur="validatepassword();" onkeyup="validatepassw
    <label id="message18"/>
  </td>
```

Preceding

- Select all nodes that come before the current node

//*[@type='submit']//preceding::input



.//*[@type='submit']//preceding::input

```
<!-- Display User ID label and its text box-->
<tbody>
  <tr>
    <td align="right">UserID</td>
    <td>
      <input type="text" onblur="validateuserid();" onkeyup="validateuserid();" maxlength="10" name="uid"/>
      <label id="message23"/>
    </td>
  </tr>
  <!-- Display Password and its text box-->
  <tr>
    <td align="right">Password</td>
    <td>
      <input type="password" onblur="validatepassword();" onkeyup="validatepassword();" name="password"/>
      <label id="message18"/>
    </td>
  </tr>
```

2 matching nodes

Summary of Locators

Locators	Description	Example
By.className	finds elements based on the value of the "class" attribute	findElement(By.className("someClassName"))
By.cssSelector	finds elements based on the driver's underlying CSS Selector engine	findElement(By.cssSelector("input#email"))
By.id	locates elements by the value of their "id" attribute	findElement(By.id("someId"))
By.linkText	finds a link element by the exact text it displays	findElement(By.linkText("REGISTRATION"))
By.name	locates elements by the value of the "name" attribute	findElement(By.name("someName"))
By.partialLinkText	locates elements that contain the given link text	findElement(By.partialLinkText("REG"))
By.tagName	locates elements by their tag name	findElement(By.tagName("div"))
By.xpath	locates elements via XPath	findElement(By.xpath("//html/body/div/table/tbody/tr/td[2]/table/tbody/tr[4]/td/table/tbody/tr/td[2]/table/tbody/tr[2]/td[3]/form/table/tbody/tr[5]"))