

Selenium With Java 2

Locators



Accessing Forms Elements

- Input Box:

```
driver.findElement(By.id("username")).sendKeys("asasya");
```

- Radio button

```
Driver.findElement(By.cssSelector("input[value='male']")).click();
```

- Check box

```
WebElement checkbox =  
driver.findElement(By.id("persist_box"));
```

```
For(int i=0;i<2;i++){
```


```
Checkbox.click();
```

```
System.out.println(checkbox.isSelected());
```

Accessing Forms Elements

- Links

```
driver.findElement(By.linkText("register  
here")).click();
```



Or Partial
link text



Locators

- ID
- Name
- Link Text

CSS Selector

- Tag and ID
- Tag and class
- Tag and attribute
- Tag, class, and attribute
- Inner text

DOM (Document Object Model)

- getElementById
- getElementsByName
- dom:name
- dom: index
- XPath

Locators

- **ID**

Id = email

- **Name**

Name = username

- Locating by Name using Filters(**Filters are additional attributes used to distinguish elements with the same name.**)

Target Format: name=*name_of_the_element*
filter=*value_of_filter*

Ex- name=tripType value=oneway

- **Link Text**

Target Format: link=*link_text*

Ex- link = register

Locators

CSS Selector

- Tag and ID

Syntax	Description
<i>css=tag#id</i>	<ul style="list-style-type: none">•tag = the HTML tag of the element being accessed•# = the hash sign. This should always be present when using a CSS Selector with ID•id = the ID of the element being accessed

Ex- "css=input#email"

take note that the HTML tag is "input" and its ID is "email".

Note: Keep in mind that the ID is always preceded by a hash sign (#).

Locators

CSS Selector

- Tag and class

Syntax	Description
<i>css=tag.class</i>	<ul style="list-style-type: none">•tag = the HTML tag of the element being accessed•. = the dot sign. This should always be present when using a CSS Selector with class•class = the class of the element being accessed

Ex - `css=input.inputtext`

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

Locators

CSS Selector

- Tag and attribute

Syntax	Description
<code>css=tag[attribute=value]</code>	<ul style="list-style-type: none">•tag = the HTML tag of the element being accessed•[and] = square brackets within which a specific attribute and its corresponding value will be placed•attribute = the attribute to be used. It is advisable to use an attribute that is unique to the element such as a name or ID.•value = the corresponding value of the chosen attribute.

Ex - `css=input[name=lastName]`

When multiple elements have the same HTML tag and attribute, only the first one will be recognized.

Locators

CSS Selector

Tag, class, and attribute

Syntax	Description
<p><code>css=tag.class[attribute=value]</code> Ex- <code>css=input.inputtext[tabindex=1]</code></p>	<ul style="list-style-type: none">•tag = the HTML tag of the element being accessed•. = the dot sign. This should always be present when using a CSS Selector with class•class = the class of the element being accessed•[and] = square brackets within which a specific attribute and its corresponding value will be placed•attribute = the attribute to be used. It is advisable to use an attribute that is unique to the element such as a name or ID.•value = the corresponding value of the chosen attribute.

Locators

CSS Selector

Inner text

Syntax	Description
<pre>css=tag:contains("inner text")</pre> <p>Ex</p> <pre>Ex: css=font:contains("Password:")</pre>	<ul style="list-style-type: none">•tag = the HTML tag of the element being accessed•inner text = the inner text of the element

Locators

Locating by DOM (Document Object Model)

getElementById

Syntax	Description
<pre>document.getElementById("id of the element")</pre> <p>Ex:</p> <pre>document.getElementById("persist_b ox")</pre>	<p>id of the element = this is the value of the ID attribute of the element to be accessed. This value should always be enclosed in a pair of parentheses ("").</p>

Locators

Locating by DOM (Document Object Model)

getElementsByName

Syntax	Description
<pre>document.getElementsByName("name") [index] Ex: document.getElementsByName ("servClass")[0]</pre>	<ul style="list-style-type: none">•name = name of the element as defined by its 'name' attribute•index = an integer that indicates which element within getElementsByName's array will be used.

Locators

Locating by DOM (Document Object Model)

dom:name

Syntax	Description
<code>document.forms["<i>name of the form</i>"].elements["<i>name of the element</i>"]</code>	<ul style="list-style-type: none">•name of the form = the value of the name attribute of the form tag that contains the element you want to access•name of the element = the value of the name attribute of the element you wish to access

Ex: `document.forms["home"].elements["userName"]`

Locators

Locating by DOM (Document Object Model)

dom:index

Syntax	Description
<code>document.forms[<i>index of the form</i>].elements[<i>index of the element</i>]</code>	<ul style="list-style-type: none">•index of the form = the index number (starting at 0) of the form with respect to the whole page•index of the element = the index number (starting at 0) of the element with respect to the whole form that contains it

Ex: `document.forms[0].elements[3]`

`document.forms[0].elements["phone"]`

Xpath

Xpath

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

Xpath locators

Id, classname, Name, Link text, Xpath, CSSpath

Types of X-path

There are two types of XPath:

- 1) Absolute XPath

`html/body/div[1]/section/div[1]/div/div/div/div[1]/div/div/div/div/div[3]/div[1]/div/h4[1]/b`

- 2) Relative XPath

`Xpath=//*[@id='rt-feature']//parent::div[1]`

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

Reference :

<http://www.guru99.com/xpath-selenium.html>

Core java

- `//` indicates a single line comment
- `/*` begins a comment that must be terminated with `*/`
- `/**` Documentation comments begin with `/**` and end with `*/` . Documentation comments are used to insert documentation into code. These comments are then used to produce documentation by javadoc

Rule

- Package names are written in small letters.
- **e.g.: java.io, java.lang, java.awt etc**
- · Each word of class name and interface name starts with a capital
- **e.g.: Sample, AddTwoNumbers**
- · Method names start with small letters then each word start with a capital
- **e.g.: sum (), sumTwoNumbers (), minValue ()**
- · Variable names also follow the same above method rule
- **e.g.: sum, count, totalCount**
- · Constants should be written using all capital letters
- **e.g.: PI, COUNT**
- · Keywords are reserved words and are