Selenium

Class 4

Agenda

WebElement Commands

Radio Buttons and CheckBoxes

WebElement Commands

WebDriver either returns something or return void(means return nothing). The same way **findElement** command of **WebDriver** returns **WebElement**. So, to get the WebElement object write the below statement:

WebElement element = driver.findElement(By.id("UserName"));

- And now if you type **element dot**, Eclipse will populate the complete list of actions
- One more thing to notice that WebElement can be of any type, like it can be a **Link, Radio Button, Drop Down, Web Table** or any HTML element.

WebElement Commands

- **sendKeys(CharSequence... keysToSend) : void** - This simulate typing into an element, which may set its value. This method accepts CharSequence as a parameter and returns nothing.
- **Command - element.sendKeys(“text”);**
- This method works fine with text entry elements like **INPUT** and **TEXTAREA** elements.

```
WebElement element = driver.findElement(By.id("UserName"));
element.sendKeys("SyntaxTech");
//Or can be written as
driver.findElement(By.id("UserName")).sendKeys("SyntaxTech");
```

WebElement Commands

- **clear( ) : void** - If this element is a text entry element, this will clear the value. This method accepts nothing as a parameter and returns nothing.
- **Command** - **element.clear();**
- This method has no effect on other elements. Text entry elements are **INPUT** and **TEXTAREA** elements.

```
WebElement element =
driver.findElement(By.name("UserName"));
element.clear();
//Or can be written as
driver.findElement(By.name("UserName")).clear();
```

WebElement Commands

- **click( ) : void** - This simulates the clicking of any element. Accepts nothing as a parameter and returns nothing.
- **Command** - **element.click();**
- Clicking is perhaps the most common way of interacting with web elements like text elements, links, radio boxes and many more.

```
WebElement element = driver.findElement(By.linkText("Pricing"));
element.click();
//Or can be written as
driver.findElement(By.linkText("Pricing")).click();
```

WebElement Commands

- **submit( ) : void**- This method works well/better than the click() if the current element is a form, or an element within a form. This accepts nothing as a parameter and returns nothing.
- **Command - element.submit();**
- If this causes the current page to change, then this method will wait until the new page is loaded.

```
WebElement element =
driver.findElement(By.xpath("//*[@value=”Login”]"));
element.submit();

//Or can be written as

driver.findElement(By.xpath("//*[@value=”Login”]")).submit();
```

WebElement Commands

- **isDisplayed( ) : boolean** - This method determines if an element is currently being displayed or not. This accepts nothing as a parameter but returns boolean value(true/false).
- **Command** - **element.isDisplayed();**

- **isEnabled( ) : boolean** - This determines if the element currently is **Enabled or not**? This accepts nothing as a parameter but returns boolean value(true/false).
- **Command** - **element.isEnabled();**

- **isSelected( ) : boolean** - Determine whether or not this element is selected or not. This accepts nothing as a parameter but returns boolean value(true/false).
- **Command** - **element.isSelected();**
- This operation only applies to input elements such as **Checkboxes**, **Select Options** and **Radio Buttons**. Th

WebElement Commands

- **getText( ) : String**- This method will fetch the visible (i.e. not hidden by CSS) innerText of the element. This accepts nothing as a parameter but returns a String value.
- **Command - element.getText();**
- This returns an innerText of the element, including sub-elements, without any leading or trailing whitespace.

```
WebElement element =
driver.findElement(By.xpath("//*[@value=”Login”]"));
String elementString = element.getText();

//Or can be written as

String elementString =
driver.findElement(By.xpath("//*[@value=”Login”]")).getText();
```

WebElement Commands

- **getAttribute(String Name) : String**- This method gets the value of the given attribute of the element. This accepts the String as a parameter and returns a String value.
- **Command - element.getAttribute();**
- Attributes are Ids, Name, Class extra and using this method you can get the value of the attributes of any given element.

```
WebElement element = driver.findElement(By.id("SubmitButton"));
String attValue = element.getAttribute("id");
//This will return "SubmitButton"
```

Difference between FindElement & FindElements Commands

The difference between **findElement()** and **findElements()** method is the first returns a WebElement object otherwise it throws an exception and the **findElements()** returns a List of WebElements, it can return an empty list if no DOM elements match the query.

findElement()	**findElements()**
• On Zero Match : throws NoSuchElementException • On One Match : returns WebElement • On One+ Match : returns the first appearance in DOM	• On Zero Match : return an empty list • On One Match : returns list of one WebElement only • On One+ Match : returns list with all matching instance

CheckBox & Radio Button Operations

- **CheckBox & Radio Button Operations** are easy to perform and most of the times the simple **ID attributes** work fine for both of these.

- But selection and d-selection is not the only thing we want with Checkboxes and Radio Buttons. We might like to check that if the Checkbox is already checked or if the Radio Button is selected by default or anything.

- Checkboxes and Radio Button deals exactly the same way and you can perform below mentioned operations on either of them.

CheckBox & Radio Button Operations

1. Various methods to select checkbox and radio button.

   1.1- Use ID for Selecting Checkbox/Radio button.
   You can use the ID attribute to select a Radio Button or a CheckBox. We've provided the Webdriver command to click which you can apply to both types of elements.

   1.2- Use IsSelected Method to Check the State of Checkbox/Radio button.
   If you've selected/deselected a Checkbox/Radio Button and you just want to check its final state. Then, you can use the <IsSelected> command to know that the correct status of the element.

Working with Radio Button/CheckBoxes

```
// Store all the elements of the same category in the list of WebElements.
List<WebElement> list = driver.findElements(By.name("radioButton"));

 // Create a boolean variable to store true/false.
Boolean is_selected = list.get(0).isSelected();

// If 'is_selected' is true that means the first radio button is selected.
      if (is_selected == true) {
            // If the first radio button is selected by default then, select the second radio button.
            list.get(1).click();
      } else {
            // If the first radio button is not selected then, click the first radio button.
            list.get(0).click();
      }
```

CheckBox & Radio Button Operations

Note: When there is a group of Radio Buttons/Check Boxes on the page then, it is possible that their names are same, but values are different. That's why we've used the Webdriver findElements command to get the list of web elements.

1.3- Use Element Value for Selecting Checkbox/Radio button.
One of the intuitive ways to select the Radio Buttons/Check Boxes is by toggling their Values. Please follow the below code example for more clarity.

Working with Radio Button/ Checkboxes

```
// Get radio group
List<WebElement> radioBtn=driver.findElements(By.name("sex"));

// Now loop from first checkbox to last checkbox.
for(WebElement radio: radioBtn) {

//Store the checkbox name to the string variable, using 'Value' attribute
String value=radio.getAttribute("value");

//check if values is same as you want and element is enabled
      if(radio.isEnabled() && value.equals("Female")) {

            // click on that element
            radio.click();
            // This statement will get you out of the for loop.
            break;
      }
```