Handling DropDown, CheckBox, and Radio Button

In this lesson, we will discuss the ways of handling dropdown and checkboxes using Selenium WebDriver.

WE'LL COVER THE FOLLOWING

- DropDown
    - Select an item from the dropdown
    - De-Select an item from the multi-select dropdown
- Radio button
- CheckBox

The below example code can be downloaded and run locally; it is included as a part of the Sample UI test framework, please refer.

DropDown

The selenium WebDriver provides a `Select` class which is used for selecting and de-selecting an option in a dropdown.

Sample HTML code for dropdown is given below:

```
<select name="fruits">
  <option value="apple">Apple</option>
  <option value="pear">Pear</option>
  <option value="banana">Banana</option>
  <option value="orange">Orange</option>
</select>
```

Select an item from the dropdown

The following code shows different ways of selecting an item from the dropdown:

```
// Create a WebDriver object
// Create a WebDriver object
WebDriver driver = DriverManager.getWebDriver();

driver.get("http://codetoautomate.com/educative-selenium-demo/");

// Create a Javascript executor object.
JavascriptExecutor jsDriver = (JavascriptExecutor) driver;

// Scroll down to the element
jsDriver.executeScript("document.getElementById('my-select').scrollIntoViewIfNeeded()");

// Identify the drop down - select element
Select select = new Select(driver.findElement(By.id("my-select")));

// select by value.
String value = "Orange";
select.selectByValue(value);

// Click on the print selected option button
WebElement button = driver.findElement(By.id("select-button"));
button.click();

// Verify the printed option == selected option
WebElement result = driver.findElement(By.id("demo1"));
Assert.assertEquals(result.getText().trim(), value);

// Select by Index
select.selectByIndex(2);

// Select by Visible text
select.selectByVisibleText("Banana");
```

De-Select an item from the multi-select dropdown

The following code shows different ways of de-selecting an item from the dropdown:

```
// Create a WebDriver object
WebDriver driver = DriverManager.getWebDriver();

driver.get("http://codetoautomate.com/educative-selenium-demo/");

// Create a Javascript executor object.
JavascriptExecutor jsDriver = (JavascriptExecutor) driver;

// Scroll down to the element
```

```
jsDriver.executeScript("document.getElementById('my-select').scrollIntoViewIfNeeded()");

// Identify the drop down - select element
Select select = new Select(driver.findElement(By.id("my-select")));

// select by value.
String value = "Orange";
select.selectByValue(value);
select.deselectByValue(value);

// Click on the print selected option button
WebElement button = driver.findElement(By.id("select-button"));
button.click();

// Verify the printed option == empty
WebElement result = driver.findElement(By.id("demo1"));
Assert.assertTrue(result.getText().trim().isEmpty());

// Select by Index
select.selectByIndex(2);
select.deselectByIndex(2);

// Select by Visible text
select.selectByVisibleText("Banana");
select.deselectByVisibleText("Banana");
```

This is applicable only for multi-select.

Radio button

We can handle the radio button by clicking the desired radio button element. The example below shows two radio buttons and the interaction between them.

```
// Create a webdriver object
WebDriver driver = DriverManager.getWebDriver();

// Open the web URL
driver.get("http://codetoautomate.com/educative-selenium-demo/");

String value = "Cricket";

// Find the radioButton WebElement with value Cricket.
WebElement source = driver
```

```
        .findElement(By.xpath(String.format("//input[@name = 'sports' and @value = '%s']", value)));
// Click on the element to select that radio button option
source.click();

// Find the submit button
WebElement button = driver.findElement(By.id("radio-button-submit"));
// Click the submit button
button.click();

// Find the displayed results text WebElement
WebElement result = driver.findElement(By.id("result"));

// Assert selected radio button and displayed are the same.
Assert.assertEquals(result.getText().trim(), String.format("Sports: %s", value));
```

CheckBox

A checkbox can be selected using its WebElement.

Following is the sample code:

```
// Create a WebDriver object
WebDriver driver = DriverManager.getWebDriver();

// open the web url
driver.get("http://codetoautomate.com/educative-selenium-demo/");

// Create a Javascript executor object.
JavascriptExecutor jsDriver = (JavascriptExecutor) driver;

// Scroll down to the element
jsDriver.executeScript("document.getElementById('checked').scrollIntoViewIfNeeded()");

// find the check-box element
WebElement checked = driver.findElement(By.id("checked"));

// Assert that if it's already selected
Assert.assertFalse(checked.isSelected());

// find the element to avoid any Stale element exception before performing
// any new operation
```

```
checked = driver.findElement(By.id("checked"));
checked.click();


// find the element to avoid any Stale element exception before performing
// any new operation
checked = driver.findElement(By.id("checked"));

// Assert the selection
Assert.assertTrue(checked.isSelected());
```

Now that you’re familiar with handling dropdown, checkbox and radio buttons. In the next lesson, you’ll learn how to add a wait on any element after a specific action.