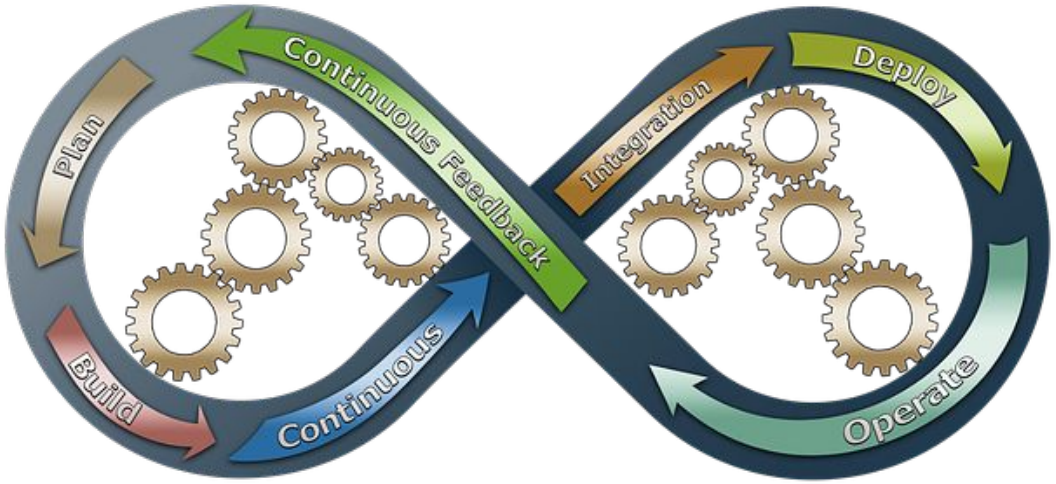



# Session 4.5

# Handling Page Elements

AN INITIATIVE BY

UNICAL ACADEMY





Email: info@unicalacademy.com | Website: www.unicalacademy.com | Phone: +91-8886978888

# Introduction

UNICAL ACADEMY

Let’s go!!!

# Handling Page Elements

UNICAL ACADEMY

- Link
- Button
- Image, Image Button, Image Link
- Edit box
- Text Area
- Check box
- Radio Button
- Drop down box
- List box
- Combo box
- Web Table / HTML Table
- Frame etc…

# Handling TextField and Text Area

UNICAL ACADEMY

- **Handle Text Area**

Capture Text Area/Capture Error Message

- **Capture Text Area:**

```
driver.get("https://www.gmail.com");
String s = driver.findElement(By.xpath("html/body/div[1]/div[2]/div[1]/h1")).getText();
System.out.println(s);
```

- **Capture Error Message:**

```
WebDriver driver = new FirefoxDriver();
driver.get("https://login.yahoo.com/");
driver.manage().window().maximize();
driver.findElement(By.xpath(".//*[@id='login-signin']")).click();
String ErrorMessage =
driver.findElement(By.id("mbr-login-error")).getText();
System.out.println(ErrorMessage);
```

UNICAL ACADEMY

- **Handle Window Popup**

```
WebDriver driver = new FirefoxDriver();
driver.get("https://mail.rediff.com/cgi-bin/login.cgi");
driver.findElement(By.name("proceed")).click();
Alert alert = driver.switchTo().alert();
String Error_Message =alert.getText(); //Returns Error message
System.out.println(Error_Message);
alert.accept();//Closes OK Button
driver.findElement(By.id("login1")).sendKeys("Inda123");
```

- **Handling Edit box**

- **Operations on Edit box**

  - Enter a value
  - Clear the Value
  - Return the Value
  - Check Displayed status
  - Check Enabled statues

# Handling Checkbox

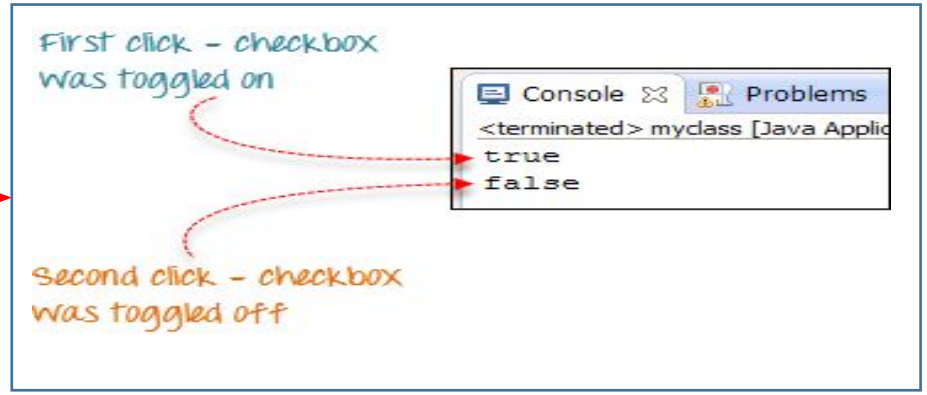
UNICAL ACADEMY

## Checkbox

- Toggling a check box on/off is also done using the **click()** method.
- The code below will click on Facebook's "Keep me logged in" check box twice and then output the result as TRUE when it is toggled on, and FALSE if it is toggled off.

```
public static void main(String[] args) {
    WebDriver driver = new FirefoxDriver();
    String baseURL = "http://www.facebook.com";

    driver.get(baseURL);
    WebElement chkFBPersist = driver.findElement(By.id("persist_box"));
    for(int i=0; i<2; i++){
        chkFBPersist.click();
        System.out.println(chkFBPersist.isSelected());
    }
    driver.quit();
}
```

UNICAL ACADEMY

## Example

```java
@Test
public void tryCheckbox(){

    WebElement option1 = driver.findElement(By.id("vfb-6-0"));  1

    //This will Toggle On the Check box
    option1.click();  2

    //Check whether the Check box is toggled on
    if(option1.isSelected()){  3
        System.out.println("Checkbox is Toggled On");

    }else{
        System.out.println("Checkbox is Toggled Off");
    }

    // This should Toggle Off the Check box
    option1.click();  4

    // Lets see whether its Toggled Off
    if(!option1.isSelected()){
        System.out.println("Checkbox is now Toggled Off !!");
    }

}
```

1. Locate the checkbox element by its ID

2. click() toggles on the checkbox

3. isSelected() gives the Toggle status of the checkbox

4. Click() again on the checkbox turns the Toggle off

# Handling Radio buttons

UNICAL ACADEMY

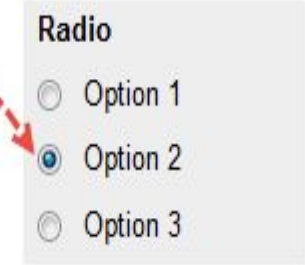
## Radio buttons

- Radio Buttons can be toggled on by using the click() method.

```
WebElement radio1 = driver.findElement(By.id("vfb-7-1"));
WebElement radio2 = driver.findElement(By.id("vfb-7-2"));

//Radio Button1 is selected
radio1.click();

//Radio Button1 is de-selected and Radio button2 is selected
radio2.click();
```


click() is the method to toggle on the radio buttons

# Handling Links

UNICAL ACADEMY

## Links

1) Accessing links using Exact Text Match: By.linkText()
2) Accessing links using Partial Text Match: By.partialLinkText()
3) How to get Multiple links with the same Link Text
4) Case-sensitivity for Link Text
5) Links Outside and Inside a Block

## Accessing links using Exact Text Match:

- **Accessing links using their exact link text is done through the By.linkText() method**. However, if there are two links that have the very same link text, this method will only access the first one.

```
<html>
    <head>
        <title>Sample</title>
    </head>
    <body>
        <a href="http://www.google.com">click here</a>
        <br>
        <a href="http://www.fb.com">click here</a>
    </body>
</html>
```

UNICAL ACADEMY

## Example

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class MyClass {

    public static void main(String[] args) {
        String baseUrl = "http://demo.guru99.com/test/link.html";
        System.setProperty("webdriver.chrome.driver","G:\\chromedriver.exe");
        WebDriver driver = new ChromeDriver();

        driver.get(baseUrl);
        driver.findElement(By.linkText("click here")).click();
        System.out.println("title of page is: " + driver.getTitle());
        driver.quit();
    }

}
```

## Here is how it works-

```
driver.findElement(By.linkText("Create a new account")).click();
```

findElement() is used to find Links in the page

click() is the method to access links

# Handling Buttons

UNICAL ACADEMY

## Buttons

The Selenium click button can be accessed using the click() method.

1.Find the button to Sign in

2.Click on the "Sign-in" Button in the login page of the site to login to the site.

```
WebElement login = driver.findElement(By.id("SubmitLogin"));
login.click();
```

UNICAL ACADEMY

## Submit Buttons

- Submit buttons are used to submit the entire form to the server.
- We can either use the click () method on the web element like a normal button as we have done above or use the submit () method on any web element in the form or on the submit button itself.
- When **submit()** is used, WebDriver will look up the DOM to know which form the element belongs to, and then trigger its submit function.

# WebElements Methods

UNICAL ACADEMY

- A WebElement is sometimes called an element.
- It symbolizes an HTML element within an HTML document.
- HTML stands for HyperText Markup Language which instructs the browser how to display content.
- The HTML element contains a start tag, end tag and content between both tags.

## Different types of WebElement Methods:

WebElements Methods are mentioned below.

- findElement
- click
- sendKeys
- getText
- getAttribute
- clear
- isDisplayed
- isEnabled

UNICAL ACADEMY

## WebElement

- The findElement() method is the most important WebElement method.
- It's important because we must first find the WebElement before performing an action on the WebElement.
- If there are multiple elements with the same locator value then findElement() locates the first WebElement. A WebElement is sometimes called an element.

**Sample:**

```
@Test
public void demoWebElementMethods () {
WebElement linkLogin =
driver.findElement(By.id("menu-item-7501"));
}
```

## Click()

- The click() method is used to click an element. However, to click an element, it must be visible with a height and width larger than zero (0).
- The Login link is visible and will be clicked after writing linkLogin.click().

**Sample:**

```
@Test
public void demoWebElementMethods () {
WebElement linkLogin =
driver.findElement(By.id("menu-item-7501"));
linkLogin.click();
```

| Modifier and Type | Method | Description |
|---|---|---|
| void | clear() | If this element is a form entry element, this will reset its value. |
| void | click() | Click this element. |
| WebElement | findElement(By by) | Find the first WebElement using the given method. |
| java.util.List<WebElement> | findElements(By by) | Find all elements within the current context using the given mechanism. |
| default java.lang.String | getAccessibleName() | Gets result of a Accessible Name and Description Computation for the Accessible Name of the element. |
| default java.lang.String | getAriaRole() | Gets result of computing the WAI-ARIA role of element. |
| java.lang.String | getAttribute(java.lang.Strin | Get the value of the given attribute of the element. |
| java.lang.String | getCssValue(java.lang.Strin | Get the value of a given CSS property. |
| default java.lang.String | getDomAttribute(java.lang | Get the value of the given attribute of the element. |
| default java.lang.String | getDomProperty(java.lang. | Get the value of the given property of the element. |
| Point | getLocation() | Where on the page is the top left-hand corner of the rendered element? |
| Rectangle | getRect() | |

UNICAL ACADEMY

| Modifier and Type | Method | Description |
|---|---|---|
| Dimension | getSize() | What is the width and height of the rendered element? |
| java.lang.String | getTagName() | Get the tag name of this element. |
| java.lang.String | getText() | Get the visible (i.e. |
| boolean | isDisplayed() | Is this element displayed or not? This method avoids the problem of having to parse an element's "style" attribute. |
| boolean | isEnabled() | Is the element currently enabled or not? This will generally return true for everything but disabled input elements. |
| boolean | isSelected() | Determine whether or not this element is selected or not. |
| void | sendKeys(java.lang.CharSequ | Use this method to simulate typing into an element, which may set its value. |
| void | submit() | If this current element is a form, or an element within a form, then this will be submitted to the remote server. |

# Handling dropdowns/listboxes

UNICAL ACADEMY

- Before proceeding with this section, let us first understand some of the basics of handling drop-downs in Selenium WebDriver.

**Select in Selenium WebDriver:**

- The 'Select' class in Selenium WebDriver is used for selecting and deselecting option in a dropdown. The objects of Select type can be initialized by passing the dropdown webElement as parameter to its constructor.

```
WebElement testDropDown = driver.findElement(By.id("testingDropdown"));
Select dropdown = new Select(testDropDown);
```

UNICAL ACADEMY

# How to select an option from drop-down menu?

- WebDriver provides three ways to select an option from the drop-down menu.

1. **selectByIndex** - It is used to select an option based on its index, beginning with 0.

```
dropdown.selectByIndex(5);
```

2. **selectByValue** - It is used to select an option based on its 'value' attribute.

```
dropdown.selectByValue("Database");
```

3. **selectByVisibleText** - It is used to select an option based on the text over the option.

```
dropdown.selectByVisibleText("Database Testing");
```

UNICAL ACADEMY

# Types of DeSelect Methods:

1. **deselectByVisibleText Method**

```
dropdown.deselectByVisibleText();
```

2. **deselectByIndex Method**

```
dropdown.deselectByIndex();
```

3. **deselectByValue Method**

```
dropdown.deselectByValue();
```

4. **deselectAll Method**

```
dropdown.deselectAll( );
```

# Select Class methods

UNICAL ACADEMY

- As Select is an ordinary class, its object is created by the keyword New and also specifies the location of the web element.

## Select Methods:

1. **selectByVisibleText**: *selectByVisibleText(String arg0): void*
   - This method is used to select one of the options in a drop-down box or an option among multiple selection boxes. It takes a parameter of String which is one of the values of Select element and it returns nothing.

```
obj.Select.selectByVisibleText("text");
```

# Select Class methods

UNICAL ACADEMY

2. **selectByIndex:** ***selectByIndex(int arg0) : void***

- This method is similar to '*selectByVisibleText*', but the difference here is that the user has to provide the index number for the option rather than text. It takes the integer parameter which is the index value of Select element and it returns nothing.

```
oSelect.selectByIndex(int);
```

3. **selectByValue:** ***selectByValue(String arg0) : void***

- This method asks for the value of the desired option rather than the option text or an index. It takes a String parameter which is one of the values of Select element and it does not return anything.

```
oSelect.selectByValue("text");
```

# Select Class methods

UNICAL ACADEMY

**4. getOptions:** ***getOptions( ) : List<WebElement>***

- This method gets all the options belonging to the Select tag. It takes no parameter and returns List<WebElements>.

```
oSelect.getOptions();
```

**5. deselectAll()**

- This method clears all the selected entries. This is only valid when the drop-down element supports multiple selections.

```
objSelect.deselectAll();
```

UNICAL ACADEMY

# Working with Dynamic Web Table Elements

- In this module, we will learn about the web tables or HTML tables in a web page, tags available in HTML and how to handle web tables dynamically.
- Web tables are basically a group of elements that are logically stored in a row and column format. It is used to organize similar information on a web page.

**Below is an example of Html table:**

| Firstname | Lastname | Points |
|---|---|---|
| Jill | Smith | 50 |
| Eve | Jackson | 94 |
| John | Doe | 80 |
| Adam | Johnson | 67 |

UNICAL ACADEMY

# Working with Dynamic Web Table Elements

**Below is the snippet of HTML structure of an HTML table:**

```
<table class="reference" style="width:60%">
  <tbody>
    <tr>
    <tr>
    <tr>
    <tr>
    <tr>
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Points</th>
    </tr>
    <tr>
      <td>Jill</td>
      <td>Smith</td>
      <td>50</td>
    </tr>
```

**Below tags are generally defined in html tables:**

1.’table’ tag defines HTML table.
2.’tbody’ tag defines a container for rows and columns.
3.’tr’ defines rows in an HTML table.
4.’td’/’th’ define the column of an HTML table.

UNICAL ACADEMY

# Find the details of a web table:

- There are many ways we can handle a web table.

**Approach #1:**

- Below is the xpath of one of the cell in html table. Let's say "firstname"

```
//div[@id='main']/table[1]/tbody/tr[1]/th[1]
```

- tr[1] defines first row and th[1] defines first column.
- If a number of rows and columns are always constant, let's say our HTML table will always have 5 rows and 3 columns.

```
for(int numberOfRows=1; numberOfRows<=5; numberOfRows++)
{
for(int numberOfCol=1; numberOfCol <=3; numberOfCol++)
{
System.out.println(driver.findElement(By.xpath
("//div[@id='main']/table[1]/tbody/tr
["+numberOfRows+"]/th["+numberOfCol+"]")));
}
}
```

- Except for row and column number, each component of XPath remains the same. So you can iterate using "for loop" for each row and column as mentioned above.

UNICAL ACADEMY

## Approach #2:

- The first approach is best suitable for the table which doesn't change its dimensions and always remains the same. Above approach will not be a perfect solution for dynamically changing web tables.

**Let's take above HTML table as an example:**

```
WebElement htmltable=driver.findElement(By.xpath("//*[@id='main']/table[1]/tbody"));
List<WebElement> rows=htmltable.findElements(By.tagName("tr"));

for(int rnum=0;rnum<rows.size();rnum++)
{
List<WebElement> columns=rows.get(rnum).findElements(By.tagName("th"));
System.out.println("Number of columns:"+columns.size());

for(int cnum=0;cnum<columns.size();cnum++)
{
System.out.println(columns.get(cnum).getText());
}
}
```

**Step 1**: First get the entire HTML table and store this in a variable 'htmltable' of type web element.
**Step 2**: Get all the rows with tag name 'tr' and store all the elements in a list of web elements. Now all the elements with tag 'tr' are stored in 'rows' list.
**Step 3**: Loop through each row and get the list of elements with tag *'th'*. *'rows.get(0)'* will give first row and *'findElements(By.tagName("th"))'* will give list of columns for the row.
**Step 4**: Iterate using *'columns.getsize()'* and get the details of each cell.
**Note**: Above approach will be best suitable if the table dimensions changes dynamically.

# Extracting data from WebTable

UNICAL ACADEMY

## Read Data In Rows To Handle Table:

- For accessing the content present in every row, to handle table in Selenium, the rows (< tr >) are variable whereas the columns (< td >) would remain constant. Hence, the rows are computed dynamically.

  - **XPath to access Row :** 1, Column : 1 – //*[@id="customers"]/tbody/tr[1]/td[1]
  - **XPath to access Row :** 2, Column : 2 – //*[@id="customers"]/tbody/tr[2]/td[2]
  - **XPath to access Row :** 3, Column : 2 – //*[@id="customers"]/tbody/tr[3]/td[2]

- A for loop is executed with rows ranging from 2..7. The column values are appended to the XPath are td[1]/td[2]/td[3] depending on the row & column that has to be accessed to handle the table in Selenium.

```
before_XPath = "//*[@id='customers']/tbody/tr["
aftertd_XPath_1 = "]/td[1]"
aftertd_XPath_2 = "]/td[2]"
aftertd_XPath_3 = "]/td[3]"
for t_row in range(2, (rows + 1)):
    FinalXPath = before_XPath + str(t_row) + aftertd_XPath_1
    cell_text = driver.find_element_by_xpath(FinalXPath).text
    print(cell_text)
```

# Extracting data from WebTable's

UNICAL ACADEMY

## Read Data In Columns To Handle Table:

- **For column-wise access to handle table in Selenium, the rows remain constant whereas the column numbers are variable i.e. the columns are computed dynamically**.

  - **XPath to access Row : 2, Column : 2** – //*[@id="customers"]/tbody/tr[2]/td[2]
  - **XPath to access Row : 2, Column : 3** – //*[@id="customers"]/tbody/tr[2]/td[3]
  - **XPath to access Row : 2, Column : 4** – //*[@id="customers"]/tbody/tr[2]/td[4]

- A for loop is executed with columns ranging from 1..4 The row values are appended to the XPath are tr[1]/tr[2]/tr[3] depending on the row & column that has to be accessed.

```
before_XPath_1 = "//*[@id='customers']/tbody/tr[1]/th["
before_XPath_2 = "//*[@id='customers']/tbody/tr[2]/td["
after_XPath = "]"
for t_col in range(1, (num_columns + 1)):
  FinalXPath = before_XPath_1 + str(t_col) + after_XPath
  cell_text = driver.find_element_by_xpath(FinalXPath).text
  print(cell_text)
```

# Select date from calendar

UNICAL ACADEMY

- A calendar control (date picker control) allows the user to select a date easily. Usually, it appears as an input field in an HTML form.
- There are two popular types of Calendar controls you'd need to automate calendar using Selenium WebDriver:

  1. **jQuery Calendar –** The jQuery calendar is a part of the open-source project at JS Foundation (previously called jQuery Foundation). There are a number of other elements like user interface interactions, widgets, effects, etc. that are also built on top of jQuery JavaScript Library.

  2. **Kendo Calendar –** Kendo Calendar is developed by Telerik.com. It is not an open-source project. Using Kendo UI, developers can build JavaScript apps faster.

- Kendo and jQuery Calendars work on all major web browsers. But there are a few exceptions- Kendo on IE works on IE7 (and above) whereas jQuery works on IE8 (and above). Both these controls are responsive and have mobile browser compatibility.

UNICAL ACADEMY

# Handling 'JQuery Calendar' in an IFRAME

- **Before performing any operation on the date picker, you have to first switch to that iFrame.**
- **Once inside the iFrame, you should perform the following operations**:

**Step 1:** Click on the Calendar Control to open the same.
**Step 2:** Find the Year drop-down control and select the required year from that drop-down.
**Step 3:** Find the Month drop-down control and select the required month from that drop-down.
**Step 4:** Once year and month is selected, locate the corresponding date by navigating through the Date table.

UNICAL ACADEMY

# Handling 'JQuery Calendar' With Dates From Multiple Months

- **Below are the test cases requirements to automate calendar using Selenium WebDriver**:

  1. Navigate to https://jqueryui.com/resources/demos/datepicker/other-months.html
  2. Locate the datepicker element and perform a click on the same
  3. Click on the 'next' button in the Calendar control till the expected year & month are found.
  4. Locate the entry in the calendar matching the 'date' and select the same to complete the test i.e. 04/01/2020

UNICAL ACADEMY

# Session Recap