Handling Element Operations

- There are 3 major operations we perform on elements
  - Entering text : on text boxes
  - Select Data : On dropdowns, display list
  - Click : On every element
- For Entering text there is sendKeys method for webelement
  - SendKeys will type the text on that control
  - If there is text already it will append
  - We can use clear method before sendkeys
- For clicking there is a click method for webelement
- For selecting data there is no method for webelement type. But there is a class called Select which will provide data selection/deselection operations
- syntax:
  - **For entering text:**
  - driver.findElement(locator).clear();
  - driver.findElement(locator).sendKeys("textToEnter");
  - **When to use clear?**
  - sendkeys will append text to the existing text box
  - if there is no text then the new text will be entered
  - if there is some text the new text will be appended
  - if we use clear before sendkeys it will remove the existing text and keeps the new text
  - **For click:**
  - driver.findElement(locator).click();
  - **For selecting data:**
  - WebElement elm = driver.findElement(locator);
  - Select lst = new Select(elm);
  - lst.selectByVisibleText("India");
- **How to handle checkbox?**
  - Perform click operation to select or deselect a checkbox.
  - webelement is having a method isSelected which returns boolean value
  - Using if condition to select checkbox when the checkbox is not selected so that we can click on it for selection
- **How to handle radio button?**
  - Perform click operation to select radio button from radio group
  - When it comes to radio button identification is the key.
  - If we have used name to findelement it always returns first element of that radio group. Because all radio buttons of that group will have the same name.
  - If we are using name to find element then use value attribute also with that. Because name attribute will be repeated for multiple radio button of that group.
  - If there is id use that to find element uniquely
  - we can also use findelements with name locator and click on radio buttons using index if value attribute is not available
- **How to handle List Box?**

- What are the operations we perform on list box?
    - selecting item
    - Get Options from list
    - Verify Option Existence in list

Sample List box html code

```
<select id="country">
<option value="select">Select One</option>
<option value="ind">India</option>
<option value="us">USA</option>
<option value="chn">China</option>
</select>
```

Select Item

- We can select items using 3 methods
    - select by visible text
    - select by value attribute
    - select by index
- Visible text is the text between Option open tag and close tag
    - WebElement elmCountry = driver.findElement(By.id("country"));
    - Select lstCountry = new Select(elmCountry);
    - lstCountry.selectByVisibleText("India");
- Value attribute will be given for option tags in case if the application is available for multiple languages.
    - lstCountry.selectByValue("ind");
- If we have to select a random value from list we can use select by index Or if value attribute is not specified for options tag and list showing in different languages then we can use index.
- We can use the same code to select multiple items if the list is multi select.
- Note: Select class will work only for the lists which has select tags
- If list is having options as checkboxes then mostly that is not created using select tag. We can handle those lists using click operation.
- **Getting options from list**
- we can use getOptions method to get list options
- This returns a list of web elements
- using for loop we can get one by one element
- We can use getText() method to get visible text from option

```
// get all options from list

List<WebElement> lstOptions = lstCountry.getOptions();

for (WebElement opt : lstOptions) {

	// get text will be used to get visible text from option
```

```
        System.out.println(opt.getText());
}
```

- **Verify Option Exist in List**
- We dont have special method to verify option existence. We have to get all options and using if condition verify the option is exist or not

Write a method to find a list option is exist or not?

```
public static boolean isListOptionExist(Select lst,String OptToCheck) {

        // get all options from list

        List<WebElement> lstOptions = lst.getOptions();

        boolean isExist = false;

        for (WebElement opt : lstOptions) {

                // get text will be used to get visible text from option

                if(opt.getText().equalsIgnoreCase(OptToCheck)) {

                        isExist = true;

                        break;

                }

        }

        return isExist;

}
```

Handling Duplicate Elements

- **What is a duplicate element?**
- Elements that are having same attributes or no attributes becomes duplicate to automation tools.
- In that case we can increase the number of attributes to find the elements
- if that is not working then we can get list of elements which are having similar attributes/locator values
- WebDriver has a method findElements which return list of webelements based on given locator.
- We can get elements from the list using index.
- syntax:
    - List<WebElement> lstElements = driver.findElements(By.xpath("elementxpath"));
    - lstElements.get(0).click();

- **What is the difference between findElement and findElements?**
- findElement return single element that identified first in application source code
- findElements return list of elements having that locator in that parent
- **Is there any other use for findElements method?**
- We can use it for getting multiple elements. For example to find how many search results displayed or how many links available in page...etc
- In most of the "how many" situations we can use findElements method

Working with HTML Tables

- **Why HTML tables in application?**
- To segregate controls in row and column manner. The alignment will be good in application if they developer use tables
- To display data in tabler format
- What are the most common operations you perform on tables?
    - Getting Data from table
    - Handle element in table cell based on other cell data

Working with Calendars

- Calendars are different from application to application
- From HTML 5 there is a new calendar element introduced <input type="date" />
- But most of the project are not upgraded to this calendar and everybody customized calendar using CSS
- A calendar is combination of multiple elements
- We need to find the logic for selecting the date
- We have to find how to select year, month and date
- In most of the cases everywhere in application we see the same calendar
- We can write a method to select date from calendar
- We can get selected date by getting data from text box
- Within method we can convert the date to the specific calendar format
- We can use SimpleDateFormat class to get the date in specific format
- we have to specify a date pattern
- You can see more pattern examples here https://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html
- We can split the output of formatted date and use the data in date selection
- Note: make sure to follow mmddyyyy format while passing the date