NOTES: DAY 7
- Today's schedule:
    - Review
    - Task#1
    - XPath review + additional functionality
    - Webtables

    - Utility methods
    - Bunch of other tasks

- How do you handle dropdowns?
- First I will inspect and see what type of dropdown it is.
- If it is HTML dropdown (non-select), I will just locate with any locator and click.
- If it is <select> dropdown, I will use Select class coming from Selenium library.

- How do we get all of the options from a Select dropdown?
- .getOptions method

- What is the return type?
- List<WebElement>

- How do we get currently selected option?
- .getFirstSelectedOption();
- this method can be used to get the default value
- also can be used to get the value after selecting something.

- What is the return type?
- Single WebElement

- How do we select options from a Select dropdown?
- We can use 3 methods provided by Selenium library
    - selectByIndex
    - selectByValue
    - selectByVisibleText

- Syntax of how do we handle Select dropdown?

#1- We create object of Select class
#2- We need to locate the dropdown <select> tag itself and pass it into the Select constructor.

```
Select dropdownName = new Select(driver.findElement(By.LOCATOR));
```

#3- We can use the object for handling the dropdowns.

```
    dropdownName.selectByVisibleText();
```

- What is an alert?
- Some pop-ups happening on the page.
- Developers use alerts to get users attention to deliver some content.

- How many types of alerts do we have?
- We have 2 types of Alerts.

#1- HTML Alert (Non-JS Alert)
    - This will be present inside of the <html> page as some web element.
    - There is nothing special about this alert, we just locate and click

#2- Javascript Alert
    - We have 3 types of alerts

    #1- Information/Warning Alert:
        - User can ONLY accept this alert.

    #2- Confirmation Alert:
        - User can accept and decline.

    #3- Prompt Alert:
        - User can accept, decline, and also sendKeys.

- How do we handle Alerts using Selenium?
- I will check the type of the alert.
- After making sure it is a JS alert, I will use Alert from Selenium library to switch drivers focus to the Alert itself.

```
    Alert alert = driver.switchTo().alert();
    alert.accept();
    alert.dismiss();
    alert.sendKeys("string");
```

- What is an iframe?
- An iframe is HTML within another HTML page.
- Most of the time it is used to create independent sections in a page for advertisements, maps, video content or anything independent.

- How do we handle iframes?
- Selenium WebDriver can focus on one thing at a time.
- Therefore we must switch its focus from the default <html> to iframe <html> before being able to do any action inside of it.

- What happens if we do not switch to inner frame?
- If we try to locate any web element before switching, we will get NoSuchElementException

- How many ways we have to switch to inner iframe?
   #1- by index:
      - starts from 0
      - we count

   syntax: driver.switchTo().frame(index);

   #2- by id-name value
      - if there is id or name attribute we can just pass their values as a string

   syntax: driver.switchTo().frame("id-name value");

   #3- by locating it as a web element
      - we locate the iframe tag itself as a web element and pass it into the method.

   syntax: driver.switchTo().frame(WebElement);
   syntax: driver.switchTo().frame(driver.findElement(By.locator));

- How do we go switch driver's focus to default iframe?
   driver.switchTo().parentFrame();
   driver.switchTo().defaultContent();

```
1   <html>

2       <html>
3           <html> --> driver.switchTo().parentFrame(); --> will focus to 2
          </html> --> driver.switchTo().defaultContent(); --> will focus to #1
      </html>

4       <html>  --> driver.switchTo().parentFrame(); --> will focus to #1
      </html> --> driver.switchTo().defaultContent(); --> will focus to #1
   <html>
```
- The only time parentFrame() method and defaultContent() method will make any difference in use is if we have iframe inside of another iframe.
- Otherwise if we have 1 layer of iframe they will both switch back to default html
----
- WINDOWS
- Is there any difference for selenium if it is a tab or window?
- No. Regardless if it is a window or a tab it is all a WINDOW for Selenium.

- How does Selenium knows which window is different from which?
- WindowHandles

- What is a window handle?
- A randomly generated unique alphanumeric ID for each window.

- Every window has their own window handle randomly generated by selenium.
- We don't have to do anything special for it.

- How do we get the current window's handle?
- .getWindowHandle();

- What is the return type?
- String
- It will capture and return the current windows handle as a String.

- How do we get all of the currently opened windows' handles?
- .getWindowHandles();

- What is the return type?
- Set <String>

- Selenium will only return the window handles from the currently executed session.

- Syntax: driver.switchTo().window(windowHandle);

- XPATH:
- How many types of xpath do we have?
- 2 types

#1- Absolute xpath:
  - Starts with "/" single slash
  - Starts from the root element -> "html" tag
  - And it goes down 1 by 1
  - Therefore it is not dependable
  - If there is any change happens in the structure of the html page it will break easily

#2- Relative xpath:
  - Starts with "//" double slash
  - "//" means jump to any given point/jump to first matching result

  syntax: //tagName[@attribute='value']

- Can we use "//" more than once in an xpath locator?
- Yes we can. We are not limited to just once. We can use as many time as we want.

- How do we go from parent to direct child?
- "/" will take us to direct child

```
    //tagName[@attribute='value']/directChildTag
```

- How do we go from parent to any child?
- "//" will take us to any child.
- It doesn't have to be directly under

- How do we go from child to parent?
- "/.."
- We do not have to provide tag name here. It will just take to the parent.

- How do we handle dynamic web elements?

#1- I can locate a parent/child that is not dynamic and move from there
#2- I will use the methods provided by xpath: contains, starts-with, ends-with

```
    //tagName[contains(@attribute, 'value')]
    //tagName[starts-with(@attribute, 'value')]
    //tagName[ends-with(@attribute, 'value')]
```

- How do we go from sibling to sibling?
- There are two methods for going from sibling to sibling
- "/preceding-sibling::" will go to the sibling that comes before
- "/following-sibling::" will go to the sibling that comes after

- The web element we locate will become the starting point (point 0).
- We decide whether we want to go preceding or following sibling.
- We count and we pass the index number.

```
    //tagName[@attribute='value']

ex:    //option[@value='3']

    <select>
2        <option value="1"> Orange 1</option>
1        <option value="2"> Orange 2</option>
0        <option value="3"> Orange 3</option>
1        <option value="4"> Orange 4</option>
2        <option value="5"> Orange 5</option>
    </select>
```

//option[@value='3']/preceding-sibling::option[1] --> this will point to Orange 1

//option[@value='3']/following-sibling::option[2] --> this will point to Orange 5
----
ex #1: locate month May using its value
    (//option[@value='4'])[1]

ex #2: locate month of March using May locator as a base and use sibling method

    (//option[@value='4'])[1]/preceding-sibling::option[2]

ex #3: locate month of October using May locator as a base and use sibling method

    (//option[@value='4'])[1]/following-sibling::option[5]

- HOW DO WE HANDLE WEBTABLES USING SELENIUM?
- We write custom locators using xpath or cssSelector and get the data we want to get.

- How do we create HTML web tables?
- <table> tag creates html web tables.

#1- We create the table tag
#2- We create rows first
#3- We create cells inside of the rows

td: table data -> used to create cells inside of a table
th: table header-> used to create cells, but it will make content bolded and centered
tr: table row -> used to create rows inside of table

P1_ WRITE A LOCATOR THAT RETURNS THE TABLE 1 ITSELF

    //table[@id='table1']

P2_ WRITE A LOCATOR THAT RETURNS ALL OF THE ROWS INSIDE OF TABLE 1

    //table[@id='table1']//tr

P3_ WRITE A LOCATOR THAT RETURNS ALL OF THE ROWS INSIDE OF BODY - TABLE 1

    //table[@id='table1']/tbody//tr

P4_ WRITE A LOCATOR THAT RETURNS ONLY 3RD ROW IN THE BODY

    //table[@id='table1']/tbody//tr[3]

P5_ WRITE A LOCATOR THAT RETURNS ALL OF THE CELLS IN ALL OF THE ROW (IN BODY)

//table[@id='table1']/tbody//tr//td

//table[@id='table1']//tr//td -> if there is no <td> in <thead> we can use this too

P6_ WRITE A LOCATOR THAT RETURNS ALL OF THE FIRST NAMES FROM TABLE

//table[@id='table1']/tbody//tr//td[2]

P7_ WRITE A LOCATOR THAT RETURNS FRANKS CELL SPECIFICALLY

//table[@id='table1']/tbody//tr[2]//td[2]

We are saying:

//tr[2] : get me the second row in the <tbody>
//td[2] : get me the second cell in the 2nd row.

P8_ WRITE A LOCATOR THAT RETURNS FRANKS CELL SPECIFICALLY
USE FRANKS TEXT

//table[@id='table1']//td[.='Frank']

- If you can create locators with text, it will be more dependent against the dynamism of the table

P9_ WRITE A LOCATOR THAT RETURNS $100 CELL SPECIFICALLY
USE FRANKS TEXT

//table[@id='table1']/tbody//tr[3]//td[4]

//table[@id='table1']//td[.='$100.00']

P10_ WRITE A LOCATOR THAT RETURNS JASONS DUE AMOUNT BASED ON JASONS NAME

1- We can locate Jason, go to parent row, and come back in the same row to get "Due" cell
//table[@id='table1']//tbody//td[.='Jason']/../td[4]

2- We can use the sibling method from xpath
//table[@id='table1']//tbody//td[.='Tim']/following-sibling::td[2]

P11_ WRITE A LOCATOR THAT RETURNS Tim's last name BASED ON tim's NAME

```
//table[@id='table1']/tbody//td[.='Tim']/../td[1]
//table[@id='table1']/tbody//td[.='Tim']/preceding-sibling::td
```

P12_ WRITE A LOCATOR THAT RETURNS ALL EMAILS THAT HAS $50 DUE DATE

```
//table[@id='table1']/tbody//td[.='$50.00']/preceding-sibling::td[1]
```