Alerts and Window Handling

Alerts (Popup) -

- Alerts are basically popups that take your focus away from the current browser and force you to read the alert message.
- We need to do some activities such as accept or Cancel the alert to resume your task on the browser.

Properties of Alert Popup -

1. We cannot identify alerts using inspect tools
2. We cannot write locator value for alerts
3. It is not a Window
4. We cannot handle alerts using javaScript Executor.
5. It will not allow performing any other operation on WebPage

Types of Alerts -

1. Simple Alert
2. Confirmation Alert
3. Prompt Alert

1. Simple Alert -

➔ Alert box with a message and an OK button, an alert box is often used to inform a user about a particular operation like details saved in the Database, right-click disabled, Loaded successfully such kind of messages.

2. Confirmation Alert -

➔ The confirmation box is the second kind of alert, it displays a dialog with the OK and Cancel button. Confirmation box informs the developer about user choice whether the user pressed OK or Cancel.

3. Prompt Alert -

➔ Prompt is used to get value from the user in text format. Prompt provides text bar for user input, Prompt is rarely used alert type.

Handle Alerts in WebdriverIO -

- In selenium, we need to switch the focus from main page to the alert to handle the alert. But WebdriverIO takes care of the switching part, so we can directly use the method to handle alerts.

Alerts functions in WebdriverIO

- **acceptAlert() -**
    - acceptAlert() function accepts the given alert/presses OK button, and brings back the focus to the web application.
- **dismissAlert() -**
    - dismissAlert() funtion closes the alert using (X) present in the alert, dismissAlert() never presses the cancel button
- **getAlertText() -**
    - getAlertText() fetches the text from the alert and we can use this text for required verification
- **sendAlertText(text) -**
    - We can Set the Text to the prompt using sendAlertText().

JavaScript

```
describe('Chercher tech Home Page', () => {
    it('Get title of chercher tech Home page', () => {
        browser.url('https://chercher.tech/practice/popups')
        $("//input[@name='alert']").click()
        browser.acceptAlert()
        $("//input[@name='confirmation']").click()
        browser.dismissAlert()
        $("//input[@name='confirmation']").click()
        const alertText = browser.getAlertText()
        console.log("Text is : " +alertText)
        $("//input[@name='prompt']").click()
        browser.sendAlertText("This is Chercher Tech")
    })
})
```

Multiple windows

- There is only one way you can get multiple windows via WebdriverIO, that is by clicking on a link that opens the page in a new browser window. WebdriverIO keeps a track of how many windows it opened during a session.
- Also, note that the WebDriverIO object always controls only one window at a time in the current session even though multiple windows are present.
- We can handle multiple windows in WebdriverIO using switch To methods which will allow us to switch control from one window to another window.
- The operating system assigns an alphanumeric id to each window as soon as the Tab/window is opened. This unique alphanumeric id is called GUID or window handle.
- We can use this unique id to differentiate a window and switch control among multiple windows.
- For example,
    - opening a link in a new window does not transfer control of WebDriverIO to a new window. WebDriverIO will be still controlling the old window and any operations that we perform using the WebdriverIO script will be forwarded to this old window.

Functions that will help us to handle multiple windows in webdriverio.

1. getWindowHandle
2. getWindowHandles
3. switchToWindow

Get Window Handles in WebdriverIO

WebdriverIO provides few methods to handle the multiple windows, let's see what are the methods and their uses.

1. **GU ID:**
   - ➔ GU ID abbreviation of Globally Unique Identifier, Every OS generates GU ID for application to identifying them uniquely. We will be using this GU ID to handle the multiple browsers, GU ID is a numeric string value.
   - ➔ 8589934593
2. **getWindowHandle() :**
   - ➔ getWindowHandle method in webdriverIO returns the current(active) browser's GU ID. It returns GU ID as string value.
   - ➔ browser.getWindowHandle()

3. getWindowHandles() :

- ➔ getWindowHandles method in webdriverIO returns GU ID of all the browsers present at the moment, which are opened by current browser. This method returns GU IDs as List of String.
- ➔ browser.getWindowHandles()

4. switchToWindow() :

- ➔ switchToWindow() method helps user to switch between windows. switchToWindow() method switches the control from the current browser window to the target browser window which has the specified "GU ID".
- ➔ browser.switchToWindow(guid);