Selenium WebDriver - Guide for Beginners

1. Navigation Methods

- driver.get(String url) : Opens the specified URL in the browser.
- driver.navigate().to(String url) : Navigates to the specified URL.
- driver.navigate().back() : Navigates back in the browser history.
- driver.navigate().forward() : Navigates forward in the browser history.
- driver.navigate().refresh() : Refreshes the current page.

2. Browser Management

- driver.getTitle() : Returns the title of the current page.
- driver.getCurrentUrl() : Returns the URL of the current page.
- driver.close() : Closes the current browser window.
- driver.quit() : Closes all browser windows and ends the WebDriver session.

3. Window and Frame Handling

- driver.manage().window().maximize() : Maximizes the browser window.
- driver.switchTo().frame(WebElement frameElement) : Switches to a frame using a WebElement.
- driver.switchTo().defaultContent() : Switches back to the main document from a frame.
- driver.switchTo().window(String windowHandle) : Switches to a specific window or tab.
- driver.getWindowHandle() : Returns the handle of the current window.
- driver.getWindowHandles() : Returns a set of all window handles.

4. Element Interaction

- driver.findElement(By locator) : Finds a single WebElement using the specified locator.
- driver.findElements(By locator) : Finds multiple WebElements using the specified locator.
- element.click() : Clicks on the WebElement.
- element.sendKeys(CharSequence keys) : Sends text to the WebElement.
- element.getText() : Retrieves the visible text of the WebElement.
- element.getDomAttribute(String attributeName) : Retrieves the value of the specified attribute
- element.isDisplayed() : Checks if the WebElement is visible.
- element.isEnabled() : Checks if the WebElement is enabled.
- element.isSelected() : Checks if the WebElement is selected (e.g., checkbox or radio button).

5. Actions Class Methods

Mouse Actions

- Actions actions = new Actions(driver); : Creates an Actions object.
- actions.click(WebElement element); : Clicks on a specific WebElement.
- actions.doubleClick(WebElement element); : Double-clicks on a specific WebElement.
- actions.contextClick(WebElement element); : Right-clicks on a specific WebElement.
- actions.dragAndDrop(WebElement source, WebElement target); : Drags and drops an element.
- actions.moveToElement(WebElement element); : Moves the mouse pointer to a specific WebElement.

Keyboard Actions

- actions.sendKeys(CharSequence keys); : Sends keys to the active element.
- actions.keyDown(Keys key); : Presses a key.
- actions.keyUp(Keys key); : Releases a key.
- actions.build(); : Builds the action sequence.
- actions.perform(); : Executes the built action sequence.

6. Select Class (Dropdown Handling)

- Select select = new Select(WebElement element); : Creates a Select object.
- select.selectByVisibleText(String text); : Selects an option by visible text.
- select.selectByValue(String value); : Selects an option by value attribute.
- select.selectByIndex(int index); : Selects an option by index.
- select.deselectByVisibleText(String text); : Deselects an option by visible text.
- select.deselectByValue(String value); : Deselects an option by value attribute.
- select.deselectByIndex(int index); : Deselects an option by index.

7. Selenium Waits

- **Implicit Wait**: Applies to all elements globally. Waits a set time for elements before throwing NoSuchElementException.

  driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));

- **Explicit Wait:** Waits for a specific condition before proceeding. Applies only to the targeted element or condition.

```
WebDriverWait wait = new WebDriverWait(driver,
Duration.ofSeconds(10));
wait.until(ExpectedConditions.visibilityOfElementLocated(By locator));
```

8. Handling Alerts

- Alert alert = driver.switchTo().alert(); : Switches to the alert.
- alert.getText(); : Retrieves the alert text.
- alert.sendKeys(String text); : Sends text to the alert (if applicable).
- alert.accept(); : Accepts the alert.
- alert.dismiss(); : Dismisses the alert.

9. Taking Screenshots

```
TakesScreenshot screenshot = (TakesScreenshot) driver;

File src = screenshot.getScreenshotAs(OutputType.FILE);

FileUtils.copyFile(src, new File("screenshot.png"));
```

Captures a screenshot of the current page and saves it as a file.

10. Common Exceptions

- NoSuchElementException : Thrown when an element is not found.
- ElementNotInteractableException : Thrown when an element is found but not interactable.
- TimeoutException : Thrown when a wait condition times out.
- NoSuchWindowException : Thrown when a window or tab is not found.
- StaleElementReferenceException : Thrown when an element reference becomes stale (no longer valid).