Handling Different Controls on Web Page – Advanced

---

1. Handling Alert Box

Why?

Sometimes web apps show alert/prompt/confirmation dialogs. Selenium must handle these using `Alert` interface.

Syntax:

```
Alert alert = driver.switchTo().alert();
alert.accept();  // Click OK
alert.dismiss(); // Click Cancel
alert.getText(); // Get alert text
alert.sendKeys("Some text"); // For prompt alerts
```

Example Scenario:

Imagine a banking app where deleting a transaction triggers a confirmation alert.

```
// Click delete button
driver.findElement(By.id("deleteTransaction")).click();

// Switch to alert
Alert alert = driver.switchTo().alert();
System.out.println("Alert says: " + alert.getText());

// Accept alert
alert.accept();
```

---

2. Handling Datepicker

Why?

Most datepickers are custom (not standard `<input type="date">`). We must either:

1. Send keys if input is editable.
2. Select from the calendar UI.

Example 1: Direct Input

```
driver.findElement(By.id("dob")).sendKeys("12/10/2025");
```

Example 2: Selecting from UI

```
// Open datepicker
driver.findElement(By.id("calendar")).click();

// Select month
driver.findElement(By.xpath("//select[@class='ui-datepicker-
month']")).sendKeys("Dec");

// Select year
driver.findElement(By.xpath("//select[@class='ui-datepicker-
year']")).sendKeys("2025");

// Select day
driver.findElement(By.xpath("//a[text()='10']")).click();
```

---

3. Handling Multiple Windows/Tabs

Why?

When clicking a link/button opens a new tab/window, Selenium needs to **switch** context.

Syntax:

```
String parentWindow = driver.getWindowHandle();
Set<String> windows = driver.getWindowHandles();

for (String window : windows) {
    driver.switchTo().window(window);
    System.out.println(driver.getTitle());
}
driver.switchTo().window(parentWindow); // back to main
```

□ Example Scenario:

Clicking **“Privacy Policy”** link opens a new tab.

```
driver.findElement(By.linkText("Privacy Policy")).click();

Set<String> allWindows = driver.getWindowHandles();
for (String win : allWindows) {
    driver.switchTo().window(win);
    if (driver.getTitle().contains("Privacy Policy")) {
        System.out.println("Switched to Privacy Policy tab");
        break;
    }
}
```

---

4. Working with Drag and Drop

Why?

Useful in file uploads, kanban boards (Trello, Jira), etc.

Syntax:

```
Actions actions = new Actions(driver);
actions.dragAndDrop(source, target).perform();
```

Example:

```
WebElement source = driver.findElement(By.id("draggable"));
WebElement target = driver.findElement(By.id("droppable"));

Actions act = new Actions(driver);
act.dragAndDrop(source, target).perform();
```

---

5. Handling Iframes

Why?

Web pages may embed content inside **iframes**. Selenium must switch inside/outside them.

Syntax:

```
driver.switchTo().frame("frameName");   // by name/id
driver.switchTo().frame(0);             // by index
driver.switchTo().frame(WebElement);    // by WebElement
driver.switchTo().defaultContent();     // back to main page
```

Example Scenario:

Filling a form inside Google Ads iframe:

```
driver.switchTo().frame("adFrame");
driver.findElement(By.id("email")).sendKeys("test@mail.com");
driver.switchTo().defaultContent();
```

---

6. Handling Dynamic Objects

Why?

Sometimes IDs/names keep changing dynamically (`id=user_123`, `id=user_456`). Use **XPath/CSS with contains() or starts-with()**.

Example:

```
driver.findElement(By.xpath("//input[contains(@id,'user_')]")).sendKeys("test
User");
driver.findElement(By.cssSelector("input[id^='user_']")).sendKeys("testUser")
;
```

---

7. Working with WebTable

Why?

Used in employee lists, financial reports, etc.

Example: Print all values from table

```
WebElement table = driver.findElement(By.id("empTable"));
List<WebElement> rows = table.findElements(By.tagName("tr"));

for (WebElement row : rows) {
    List<WebElement> cols = row.findElements(By.tagName("td"));
    for (WebElement col : cols) {
        System.out.print(col.getText() + " | ");
    }
    System.out.println();
}
```

Synchronization in Selenium

1. Why Synchronization?

- Web elements load at different times.
- Without waits, Selenium throws `NoSuchElementException`.
- Synchronization ensures Selenium waits for elements.

---

2. Thread.sleep()

Hard wait (not recommended, blocks thread).

```
Thread.sleep(3000); // wait 3 sec
```

---

3. Implicit Wait

Global wait for all elements.

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

---

4. Explicit Wait

Wait for specific element/condition.

```
WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(15));
WebElement element =
wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("username")));
element.sendKeys("admin");
```

---

5. Fluent Wait

Like Explicit Wait, but polls at intervals.

```
Wait<WebDriver> wait = new FluentWait<>(driver)
    .withTimeout(Duration.ofSeconds(20))
    .pollingEvery(Duration.ofSeconds(2))
    .ignoring(NoSuchElementException.class);

WebElement element = wait.until(driver ->
driver.findElement(By.id("username")));
```

Exception Handling

Why?

Selenium tests may fail due to missing elements, timeouts, etc. Use **try-catch-finally**.

Example:

```
try {
    driver.findElement(By.id("loginBtn")).click();
} catch (NoSuchElementException e) {
    System.out.println("Login button not found: " + e.getMessage());
} finally {
    driver.quit();
}
```

Perfect
Let me create a **single Selenium Java project file** (with `main()` method) that demonstrates **all the scenarios step by step**:

Selenium Java Project – Advanced Controls, Synchronization & Exception Handling

```
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.*;

import java.time.Duration;
import java.util.List;
import java.util.Set;

public class SeleniumAdvancedDemo {
    public static void main(String[] args) {
        // Set path to chromedriver if not set in system environment
        // System.setProperty("webdriver.chrome.driver",
"path/to/chromedriver");

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();

        try {
            // ============================
            // 1. Handling Alert Box
            // ============================
            driver.get("https://the-
internet.herokuapp.com/javascript_alerts");
```

```
            driver.findElement(By.xpath("//button[text()='Click for JS
Alert']")).click();
            Alert alert = driver.switchTo().alert();
            System.out.println("Alert text: " + alert.getText());
            alert.accept();

            // ============================
            // 2. Handling Datepicker
            // ============================
            driver.get("https://jqueryui.com/datepicker/");
            driver.switchTo().frame(0); // datepicker is inside iframe
            WebElement dateBox = driver.findElement(By.id("datepicker"));
            dateBox.click();
            driver.findElement(By.xpath("//a[text()='15']")).click();
            driver.switchTo().defaultContent();

            // ============================
            // 3. Handling Multiple Windows/Tabs
            // ============================
            driver.get("https://the-internet.herokuapp.com/windows");
            String parent = driver.getWindowHandle();
            driver.findElement(By.linkText("Click Here")).click();

            Set<String> allWindows = driver.getWindowHandles();
            for (String win : allWindows) {
                driver.switchTo().window(win);
                if (!win.equals(parent)) {
                    System.out.println("Switched to new tab: " +
driver.getTitle());
                    driver.close();
                }
            }
            driver.switchTo().window(parent);

            // ============================
            // 4. Drag and Drop
            // ============================
            driver.get("https://the-internet.herokuapp.com/drag_and_drop");
            WebElement source = driver.findElement(By.id("column-a"));
            WebElement target = driver.findElement(By.id("column-b"));
            Actions act = new Actions(driver);
            act.dragAndDrop(source, target).perform();

            // ============================
            // 5. Handling Iframes
            // ============================
            driver.get("https://the-internet.herokuapp.com/iframe");
            driver.switchTo().frame("mce_0_ifr");
            WebElement editor = driver.findElement(By.id("tinymce"));
            editor.clear();
            editor.sendKeys("Hello, inside iframe!");
            driver.switchTo().defaultContent();

            // ============================
            // 6. Handling Dynamic Objects
            // ============================
```

```
            driver.get("https://the-
internet.herokuapp.com/dynamic_loading/1");
            driver.findElement(By.cssSelector("#start button")).click();
            WebDriverWait wait = new WebDriverWait(driver,
Duration.ofSeconds(15));
            WebElement helloText = wait.until(

ExpectedConditions.visibilityOfElementLocated(By.id("finish"))
            );
            System.out.println("Dynamic text: " + helloText.getText());

            // ============================
            // 7. Working with WebTable
            // ============================
            driver.get("https://the-internet.herokuapp.com/tables");
            WebElement table = driver.findElement(By.id("table1"));
            List<WebElement> rows = table.findElements(By.tagName("tr"));
            for (WebElement row : rows) {
                List<WebElement> cols = row.findElements(By.tagName("td"));
                for (WebElement col : cols) {
                    System.out.print(col.getText() + " | ");
                }
                System.out.println();
            }

            // ============================
            // 8. Synchronization Examples
            // ============================
            driver.get("https://the-
internet.herokuapp.com/dynamic_controls");

            // Thread.sleep (not recommended)

driver.findElement(By.xpath("//button[text()='Remove']")).click();
            Thread.sleep(2000);

            // Implicit Wait

driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));
            driver.findElement(By.id("message"));

            // Explicit Wait
            driver.findElement(By.xpath("//button[text()='Add']")).click();

wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("message")));

            // Fluent Wait
            Wait<WebDriver> fluentWait = new FluentWait<>(driver)
                    .withTimeout(Duration.ofSeconds(15))
                    .pollingEvery(Duration.ofSeconds(2))
                    .ignoring(NoSuchElementException.class);

            WebElement msg = fluentWait.until(d ->
d.findElement(By.id("message")));
            System.out.println("Fluent Wait message: " + msg.getText());

        } catch (Exception e) {
```

```
            System.out.println("□ Exception occurred: " + e.getMessage());
        } finally {
            driver.quit();
        }
    }
}
```

Project Demonstrates:

1. **Alert Handling** → accept/dismiss alert box.
2. **Datepicker** → selecting a date.
3. **Multiple Windows/Tabs** → switching between windows.
4. **Drag and Drop** → using `Actions`.
5. **Iframes** → switch in/out of iframe and type.
6. **Dynamic Objects** → wait for hidden element.
7. **WebTable** → iterate rows & columns.
8. **Synchronization** → `Thread.sleep`, `Implicit`, `Explicit`, `Fluent`.
9. **Exception Handling** → `try-catch-finally` with `driver.quit()`.