Module Name: Selenium WebDriver

Trainer Name: Ankush Vankore

Total Sessions: 19

* Selenium WebDriver
  + Automation Concepts
  + Configuration
  + Launching Different Browsers
    - Chrome
    - Edge
    - Firefox
    - Safari
  + Basic Methods of WebDriver
    - get()
    - getTitle()
    - getCurrentUrl()
    - getPageSource()
    - close()
    - quit()
  + Locators
    - Name
    - Id
    - ClassName
    - CssSelector
    - XPath
    - LinkText
    - PartialLinkText
    - TagName
  + Handling different controls
    - Text box
    - Command button
    - Check box
    - Radio button
    - Hyperlinks
    - Drop down list
    - List box
  + Synchronization
    - Thread.sleep()
    - ImplicitWait
    - ExplicitWait
    - FluentWait
    - PageLoadTimeout
  + Handling tables
  + Handling multiple windows
  + Handling alerts
  + JavascriptExecutor interface
    - Scrolling the page
    - Clicking on controls
    - Entering value in text boxes
  + Handling FileUpload control
  + Robot Class
  + Handling Menus
  + Mouse Actions
    - Click
    - Right click
    - Double click
    - Drag and drop
    - Hovering the mouse
  + Handling iframe
  + Screenshot
* TestNG
  + Advantages
  + Configuration
  + Executing Single test
  + Executing Multiple tests
  + Setting the priority
  + Annotations
    - @Test
    - @BeforeTest
    - @AfterTest
    - @BeforeMethod
    - @AfterMethod
  + Data driven testing via @DataProvider
  + Assertions
  + Reports
    - Normal Report
    - HTML Report
  + Executing test / tests via xml file
  + Parameterization
  + Keyword Driven Framework
  + Modular Framework
  + Page Object Module
  + Data Driven Framework
    - Reading the data from Excel
    - Writing the data to Excel
  + Hybrid Framework
* Maven
  + Introduction & Advantages
  + Configuration
  + Adding dependencies
  + Executing tests
  + Extent Report
* Cucumber
  + Introduction to BDD Approach
  + Advantages
  + Configuration
  + Feature File
  + Keywords in feature file
    - Feature
    - Scenario
    - Given
    - Then
    - When
    - And
    - Background
    - Examples
    - Scenario Outline
  + Step Definition
  + Runner class
  + Tags in Cucumber
  + Hooks in Cucumber
  + Data Driven Testing in Cucumber
* Revision of Java
  + Language Basics
    - Variables
    - Operators
    - Checking conditions
    - Loops (for loop and for each loop)
    - Functions
    - Arrays
  + Class & Object
    - Creating Class
    - Creating object
    - Constructor
    - Static members and functions
  + Inheritance
    - Extends keyword
  + **Interface**
  + Exception Handling
    - **throws** keyword
  + Collections
    - **List**
    - Set
    - Map

**Software Testing**

It is the process of checking Correctness, Completeness, Security & Quality of developed software application.

**Manual Testing Process**

1. Creating Test Scenarios
2. Creating Test Cases
3. Creating Test Data
   1. ECP
   2. BVA
   3. EG
4. Creating RTM

**Operations involved in Testing**

1. Entering data in the text box
2. Clicking on buttons
3. Navigation from one page to another
4. Selecting value from Check box, Radio button, Dropdown list, list box
5. Navigation from tabs
6. Validating the expected result with actual result
7. Generating report
8. Mark the test case Pass or Fail

In manual testing tester uses their **Hand-Eye-Brain** Coordination.

**What is Automation Testing?**

* Perform all above actions to test any application via a machine.
* Machine in this context is Test Automation Tool / Automation Testing Tool
* Every tool is a software
* Every tool understand specific programming language
* You have to give the instructions using any one of the supported programming language.

**Advantages & Need of Automation**

* Repetition
* Saves time
* Faster executing
* Reduces human efforts
* Regression Testing
* Less human errors
* Reusability
* Improve the quality
* 100% Test Coverage
* Compability Testing
* Huge amount of data can be tested

**When to Automate and which tests to be automated?**

* Stable build
* Repetitive task
* Large data
* Compability Testing
* Performance Testing
* Frequent Regression testing
* Security testing

**Types of Tools**

* Unit Testing
  + JUnit
  + NUnit
* GUI Testing / Functional Testing
  + Selenium WebDriver
  + Tosca
  + QTP
  + Appium
  + Playwright
* API Testing
  + Postman
  + REST API

**Process of Automation**

1. Planning
2. Selection of Tool
   1. Type of application to be tested. (AUT / SUT)
   2. Cost of tool
   3. Support availability
   4. Tool availability / market presence
   5. Testers availability
3. Creating Test Script
4. Creating Test Data
5. Execution
6. Report
7. Maintance

**Selenium**

It is suite / bundle of Test Automation Tools to test Web Based Applications (Web Sites)

**Components of Selenium**

1. Selenium IDE – Record and Playback
2. Selenium Grid – Parallel Execution
3. Selenium RC -
4. Selenium WebDriver

**Selenium WebDriver**

* It is an automation testing tool to test web based applications
* It is an API (Application Programming Interface)
* Interface in Java

**Pre-Requisite for Selenium WebDriver**

1. Minimum Windows 10
2. Java (Minimum JDK 11)
3. Editor (IDE)
   1. Eclipse
   2. Idea Itellij
4. At least one Updated Browser
5. Selenium Jar File

**Configuration**

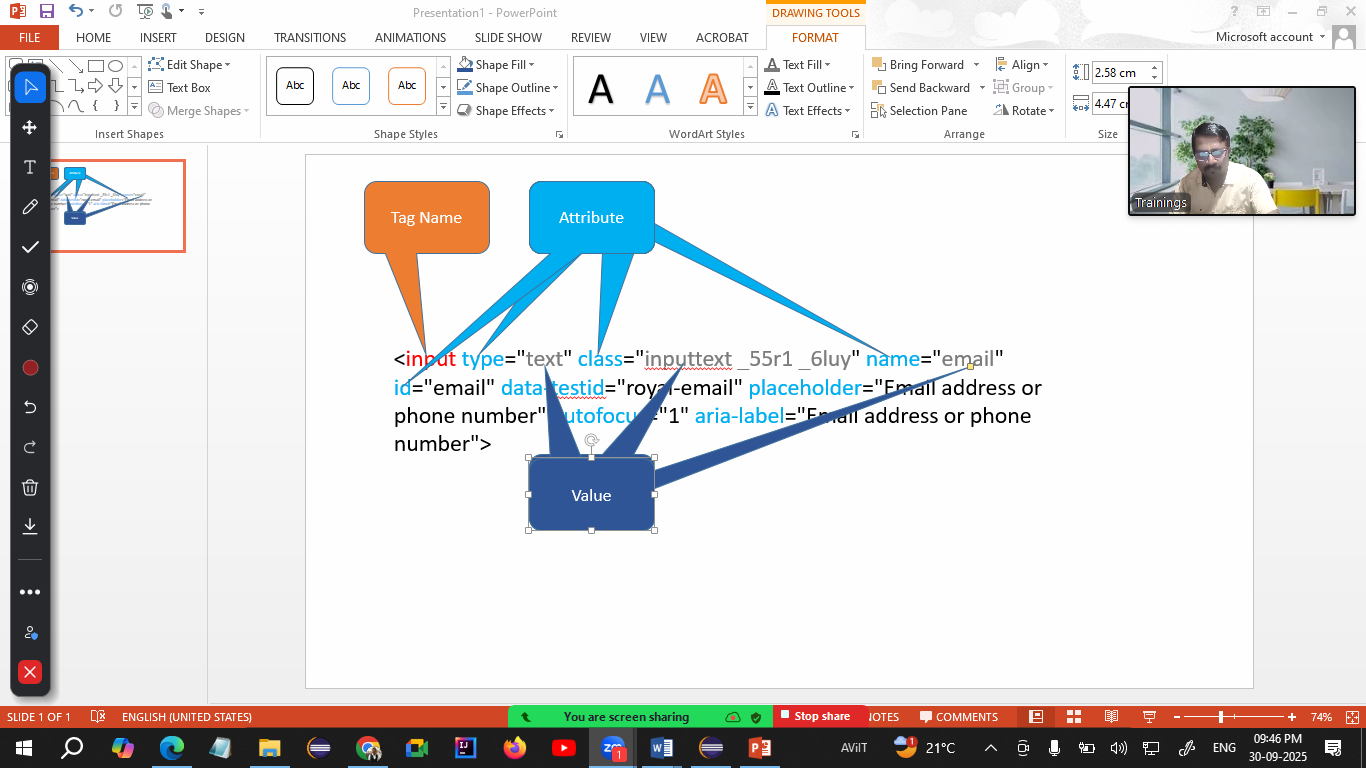
1. Create 2 folders
   1. YourName\_SeleniumDemos (Gaurav\_SeleniumDemos) 🡪 To store all selenium demos
   2. Selenium Jar Files 🡪 To store all the jar files (libraries) required for Selenium
2. Launch <https://www.selenium.dev/>
3. Click on Downloads link
4. To download the Selenium Jar file Click on Latest stable version [4.35.0](https://github.com/SeleniumHQ/selenium/releases/download/selenium-4.35.0/selenium-server-4.35.0.jar)
5. Open your Downloads folder (or where this file is downloaded)
6. Copy this file and paste in 2nd folder (Selenium Jar Files)
7. Open Eclipse
8. Click on Browse button and select the 1st folder which you have created in earlier steps.
9. Click on Launch button (Here we selected the workspace ie. Folder)
10. Click on File 🡪 New 🡪 Java Project
    1. Give the name for project (SeleniumAutomationProject)
    2. Select the Java version (only of it is showing less that 11)
    3. Uncheck **Create module-info.java file** checkbox
    4. Click on Finish
11. Create a package in this project. (com.WebDriverDemos)
12. Create a class in this package.  
    **Selenium Configuration**
13. Right click on Project 🡪 Build Path 🡪 Configure Build Path
14. Click on Libraries tab
15. Click on Classpath
16. Click on Add External JARs…
17. Select the file which you downloaded and stored in 2nd folder
18. Click on Open button
19. Click on Apply and Close button

**WebDriver Methods.**

1. Create object of WebDriver interface 🡪 Launch the blank browser window.
2. get() 🡪 Open the URL.
3. driver.manage().window().maximize() 🡪 Will maximize the browser window.
4. close() 🡪 Will close the **current** browser window.
5. getTitle() 🡪 Will return the title of web page opened by WebDriver object. (String)
6. getCurrentUrl() 🡪 Will return the URL of web page. (String)
7. getPageSource() 🡪 Will return page source (rendered HTML) of page (String)
8. findElement() 🡪 It will find and return single control on the page. (WebElement)

**Common Exceptions**

1. InvalidArgumentException 🡪 The URL is not in the correct format. URL should be absolute means should start with https / http
2. NoSuchElementException 🡪 Selenium is unable to locate / find the control due to
   1. Your locator value may be wrong.
   2. Your locator value may be dynamic.



**Locators**

Locators are the way / method to locate or identify the controls on the page.

1. Name
2. Id
3. ClassName
4. CssSelector
5. LinkText
6. PartialLinkText
7. XPath
8. TagName

**WebElement**

* Every control on the web page is treated as WebElement like Text box, command button, radio button, hyperlink, dropdown list, list box, some text on the page etc.
* WebElement is an interface in Selenium. Which can hold any control on the page.

**Methods of WebElement Interface**

1. sendKeys() 🡪 Will enter the text in text box. Will append the text in the text box.
2. click() 🡪 Will click on any control.

**CssSelector**

You can read / find any control on the page using any one of the attribute of that control.

1. Using Single Attribute  
   Syntax:   
   tagName[attribute=”value”]  
   input[placeholder="Email address or phone number"]
2. Using Multiple Attributes  
   Syntax  
   tagName[attribute1=”value”][attribute2=”value”]
3. Using Special Characters
   1. ^ 🡪 Starts With (If starting characters are fixed and ending characters are dynamic)
   2. $ 🡪 Ends With (If ending characters are fixed and starting characters are dynamic)
   3. \* 🡪 Contains text (Checks for the text containing in the attribute value)

