Ankush Vankore

Software Testing:

Process of checking the **correctness**, **completeness**, **security** and **quality** of developed software application.

Manual Testing:

Using your hand – eye – brain co-ordination.

1. Entering data in text box
2. Clicking on some links
3. Clicking on button
4. Selecting single option from multiple options (List box, drop down list, check box, radio button)

What is mean by Automation?

Automation is nothing but performing all above action with some automation testing tool

Java Basics for revision

1. Basics (Variables, operators,)
2. Conditions (if, if-else, switch case, nested if else)
3. Loops (for & enhanced for loop)
4. Wiring and calling methods
5. Creating object of class and calling the methods via class
6. Creating and using static members and methods
7. Collections
   1. List
   2. Set
   3. Map
8. Exception Handling
   1. throws keyword

Automation Testing

Advantages

* Save time
* It saves money
* It saves human errors
* Reduces human efforts
* Reusable
* Full coverage
* More productivity
* More accuracy
* Efficiency
* Consistency
* Get the report
* Cost saving

Need of automation:

* Faster execution
* Saves time
* Full coverage
* Accuracy
* Increases performance
* Good quality of testing
* Reporting
* Reduces work load
* More innovative
* Compability testing

When to Automate:

* Stable build
* Regression testing
* Repeated test scenario
* Large number of test cases
* Large amount of data to be tested
* Cross browser testing
* Performance testing
* Parallel testing

Types of automation testing

* Unit Testing
* API Testing (Application Programming Interface)
* GUI Testing
* Mobile Testing
* Performance Testing
* Security Testing

Tools

* JUnit / NUnit
* Selenium
* Postman
* Cucumber
* Tosca
* Appium
* REST Assured
* Load Runner
* HP Web Scrab
* Maven
* QTP
* Winium

Process of Automation:

1. Planning
2. Selection of Tool
   * Technology
   * Cost of tool
   * Human Resources availability
   * Support availability
   * Market presence
3. Create test script
4. Execute the script
5. Generate Report
6. Maintance

**Selenium**

**Suite / Bundle of Test Automation Tools used to test *Web / Browser Based Applications*** (Web Sites)

**Components of Selenium**

* Selenium IDE – Record and playback
* Selenium Grid – Parallel Execution
* Selenium RC (Remote Control) – Deprecated
* Selenium WebDriver – In replacement of RC

Installing Selenium IDE

Open selenium.dev 🡪 Downloads tab 🡪 scroll to Selenium IDE 🡪 Choose your browser and click on that browser name 🡪 Click on Add Extention

**Selenium WebDriver**

* A test automation tool to test web based applications (Web Sites)
* WebDriver is an **interface** in java.
* It is an API

**Pre – Requisite**

1. Windows OS
   1. Minimum 10
2. Java
   1. Minimum version: 11
3. Editor
   1. Eclipse / Intellij
4. Any updated browser
   1. Chrome / Firefox / Edge / Opera / Safari
5. Latest Selenium Jar file (API)

**Configuration Part**

1. Create 2 folders (mostly except C: )
   1. YourName\_SeleniumDemos
   2. WebDriver Jar Files
2. Open <https://www/selenium.dev>
3. Click on **Downloads**
4. Click on Latest stable version [4.24.0](https://github.com/SeleniumHQ/selenium/releases/download/selenium-4.24.0/selenium-server-4.24.0.jar) (Version may vary) **(DON’T OPEN THIS FILE)**
5. Store this file in WebDriver Jar Files
6. Open Eclipse
   1. Select the workspace (The folder where you are going to store your project) – Select the folder created in 1st step YourName\_SeleniumDemos
7. Click on Launch
8. File 🡪 New 🡪 Java Project
9. Give Name to project
10. Select Java version as Java-SE-11
11. Uncheck **create module-info.java file** checkbox
12. Create a new package inside this project
13. Create a class inside this package
14. Right click on project (in Package Explorer)
15. Click on Build Path
16. Click on Configure Build Path
17. Click on Libraries Tab
18. Click on Class Path
19. Click on Add External Jars…
20. Browse and select the File downloaded (Selenium-Server-xxxx.jar)
21. Click on Open
22. Click on Apply & Close

Selenium WebDriver

1. Launching the browser: Create the object of WebDriver interface. It will launch blank browser page.
2. get(): Launch the specified url
3. driver.manage().window().maximize(): Maximize the browser window
4. close() : Close the current browser window
5. getTitle() : Read the title of page (String)
6. getCurrentUrl() : Read the URL of page (String)
7. getPageSource() :
8. findElement() : Reads the WebElement / Control on the page using any one locator. **Always locates first occurrence.** (WebElement)

**WebElement**

* Interface
* This refers any control on the page
* Methods
  + sendKeys() : Used to enter some text in the textbox (Will append the text)
  + click() : used to click on any control
  + getText() : Used to read the text on the control (String)

Common Exception in Selenium WebDriver

1. InvalidArgumentException: Your url is wrong (URL should always start by https means it should be ABSOLUTE URL)
2. SessionNotCreatedException: Your browser and WebDriver version are mismatching.
3. NoSuchElementException: Selenium is unable to locate this control because of some reason.
   1. Value of locator is wrong
   2. Value of locator may be dynamic. (Every time it is changing.)
4. InvalidSelectorException: The value of locator is in wrong format.
5. ElementNotInteractableException: You are not able to interact with this control.

**Locators**

These are the ways to find / locate any WebElement on the page.

1. Id
2. Name
3. className
4. CssSelector
5. Xpath
6. LinkText
7. PartialLinkText
8. TagName
9. RelativeLocator

<input type="text" class="inputtext \_55r1 \_6luy" name="email" id="email" data-testid="royal\_email" placeholder="Email address or phone number" autofocus="1" aria-label="Email address or phone number">

TagName

Attribute

Value

CssSelector:

Can be used to read the control by using any one of the attribute of control.

1. Using single attribute  
   tagName[attribute=”value”]  
   input[type="text"]
2. Using multiple attributes  
   tagName[attribute1=”value”] [attribute2=”value”]  
   input[placeholder="Password"][aria-label="Password"]
3. Using special characters
   1. ^ - Starts with  
      tagName[attribute^=”value”]
   2. $ - Ends with  
      tagName[attribute$=”value”]
   3. \* - Contains   
      tagName[attribute\*=”value”]
   4. # - id  
      tagName#id value
   5. . – className  
      tagName.class – specify single class only