16 Dec Automation Testing

Date - 05 Feb 24

Testing -

Client - Requirement
Product backlog
Sprint backlog

Team -

Dev - 6
QA  - 4 (2 Manual + 1 Automation + 1 Mix)
TL -  1
PO -  1
---
12 Team Members

Types

1. Manual Testing - without tools
2. Automation Testing - with tools

QA - SIT

Testcases -

expected result == actual result  - Passed
expected result != actual result  - failed

Automation Testing -

The execution of the test cases by using an automation tool or third-party tool is called automation testing.

repeated task
tendious or complex task

Adv of Auto Testing

Fast
reliable
Reuse
time-saving
accuracy improvement
reducing human-generated error ways

==========

Date - 06 Feb 24

Tools used -

1. Selenium
2. QTP
3. Playwright
3. Toska
4. WDIO
5. Cypress

List out some of automation tool?

Selenium -

1. It is Open source (free)
2. It supports Multiple languages: - Java, python, c#, ruby, JS etc..
3. It supports Multiple OS: - Windows, MacOS, Linux etc...
4. It supports Multiple browsers e.g. chrome, Firefox, safari, opera, edge etc...
5. It helps is to do Parallel Testing
6. It supports to integrate third party tools like TestNG, Cucumber, Git, Jenkins, GitHub, Maven etc.

7. It has very large community

Selenium Components

1. Selenium IDE
2. Selenium RC
3. Selenium WebDriver
4. Selenium Grid

Appium
Katalon
TestComplete
Jmeter - Performance testing
Postman  - AP Testing
LamdaTest
Robotex
SoupUI  -
Cucumber -

==========

Date - 07 Feb 24

Selenium Components

1. Selenium IDE -

simple testcases
Chrome, Firefox and Edge add-on
simple record-and-playback of interactions with the browser

Drawbacks -

Complex TestCases

2. Selenium RC - (Remote Control - Server)

Eclipse >> Server >> Browser

slow server
server shutdown

3. Selenium WebDriver -

Eclipse >> Browser

4. Selenium Grid -

several machines and manage multiple environments from a central point
Remote Automation

==========

Date - 08 Feb 24

Selenium Arch...

1. SearchContext is the super most interface, which consists of abstract methods and inherited by WebDriver.
2. WebDriver is an Interface which contain abstract methods of both interfaces i.e., search context and its own abstract methods
3. RemoteWebDriver provides implementation to all the abstract methods from both the interfaces.
4. Driver classes like ChromeDriver(), EdgeDriver(), FirefoxDriver() etc in hierarchy. These classes provides control on a browser running on the local machine.

```
ChromeDriver driver = new ChromeDriver();
FirefoxDriver driver = new FirefoxDriver();
EdgeDriver driver = new EdgeDriver();
```

```
WebDriver p = driver;

WebDriver driver = new ChromeDriver();
WebDriver driver = new FirefoxDriver();
WebDriver driver = new EdgeDriver();
```

---

How to add jar files in project

1. Visit https://www.selenium.dev/downloads/
2. Download latest version of selenium - 4.17.0
3. Unzip your folder and Copy and Paste in specfic location

1. Right click on project >> Properties >> Java Build path >> Library >
2. Click on classpath >> Add External jars >> Ctrl + A
3. Apply and close

==========

09 Feb 24

Get commands

1. Get Method
2. Get Title Method
3. Get CurrentURL Method
4. Get PageSource Method

Navigate Commands/Methods

5. navigate().to()
6. Forward command
7. Back command
8. Refresh command

Browser commands

9. quit
10. close
11. maximize
12. minimize
13. fullscreen
14. Thread.sleep(ms);

---

Date - 13 Feb 24

Website - collection of webpages
webpage - collection of webelements
webelement - such as images, videoes, textbox, dropdown, radio button, button, links etc..

HTML -

Hypertext - link
markup lang - tag

<tagname> {

</tagname> }

1. html tag -

<html>

</html>

favicon

2. head tag

<head>
</head>

3. body tag

<body>
</body>

4. heading tag

➢ <h1>This is heading 1</h1>
➢ <h2>This is heading 2</h2>
➢ <h3>This is heading 3</h3>
➢ <h4>This is heading 4</h4>
➢ <h5>This is heading 5</h5>
➢ <h6>This is heading 6</h6>

5. paragraph tag

voided tag

<tag>  or <tag/>

6. line break tag

<br>  or <br/>

7. center tag

8. Horizontal Lines

<hr> or <hr/>

9. link -

<a href = "url"> text </a>

==========

Date - 14 Feb 24

1. <title tag -

<title>
</title>

2. form tag

<form>
</form>

3. input text box -

<input type = "text" >

4. password box -

<input type = "password" >

5. checkbox

<input type = "checkbox" >

6. radio -

<input type = "radio" >

7. button -

<input type = "button" >

8. Dropdown -

```
<select>
    <option>Mumbai</option>
    <option>Pune </option>
    <option>Delhi</option>
    <option>Hyderbad</option>
</select>
```

Attribute -

key = "value"

type = "text"

id, name, class, classname, placeholder etc....

==========

Date - 15 Feb 24

Locator -

- are nothing but these are techmiques used to find webelement from the webpage.

- to find webelements we will use below method -

```
driver.findElement(locator);
driver.findElements(locator);
```

- locator - By - methods -

By.method("value");

```
driver.findElement(By.method("value"));        - single webelement
driver.findElements(By.method("value"));       - multiple webelement
```

Types -

1. id
2. name
3. classname
4. linkText
5. partialLinkText
6. Tagname
7. CSS Selector -
8. Xpath -

sendKeys("text");
click();

1. id - locates element using id attribute

```
WebElement ref_var = driver.findElement(By.id("value"));
```

2. name - locates element using name attribute

```
WebElement ref_var = driver.findElement(By.name("value"));
```

3. classname - locates element using class attribute - not supporting

WebElement ref_var = driver.findElement(By.className("value"));

4. linkText - to handle links

WebElement ref_var = driver.findElement(By.linkText("value"));

5. partialLinkText - to handle links

WebElement ref_var = driver.findElement(By.partialLinkText("value"));

- NoSuchElementException

==========

Date - 16 Feb 24

8. Xpath -

1. Absolute xpath
2. Relative xpath

1. Absolute xpath -

- path from root element to targeted element without missung any element/tag in between.
- Absolute xpath start with '/'

Eg.

/html/body/center/form/input[1]
/html[1]/body/div[1]/header/div/div[1]/div[2]/div/form/div[2]/div[1]/input

Drawbacks

1. full understanding of html coding
2. ui changes - test script failure

2. Relative xpath -

pah from target element
//

1. Basic xpath

xpath by using attribute -

xpath = //tagname[@attribute = "value"]  // id, name, class, placeholder etc....

xpath by using text function -

xpath = //tagname[text() = 'text_value']

---

Date - 19 Feb 24
org.openqa.selenium.NoSuchElementException

u_0_b_lL
u_0_b_cN
u_0_b_TO

u_0_b_

1234test
1255test

54test544
53test526

2. xpath by contains keyword/function

xpath by using attribute -

xpath = //tagname[contains(@attribute , 'value')]

xpath by using text -

xpath = //tagname[contains(text(),'text_value')]

(//div[contains(@class,'_4rR01T')][1]

Relative Locator -

Selenium 4 - Relative Locators

The findElement method accepts a new method with(By) which returns a RelativeLocator
above
below
toLeftOf
toRightOf
near

```
WebElement loginButton = driver.findElement(By.id("btnLogin"));

WebElement passwordTextBox = driver.findElement(RelativeLocator.with(By.tagName("input")).above(loginButton));
```

==========

Date - 20 Feb 24

Dropdown - web element which is available on webpage, has multiple options and user need select one of them.

How to handle dropdown -

1. store webelement ref

WebElement ref = driver.findElement(locator);

2. Create a object Select class -

Select s = new Select(ref);

3.

```
s.selectByIndex(index);            //16
s.selectByValue("value");          // "search-alias=electronics"
s.selectByVisibleText("text");     // "Electronics"
```

driver.switchTo().frame();

Iframe -

Displaying webpage as part of another webpage called as iFrame.

<iframe> </iframe>

How to identify the iframe:

1. Right click on the element, if you find the option like ‘This Frame’ then it is an iframe.
2. Inspect >> search for iframe tag in code

How to handle the iframe

We need change the focus of selenium from main page to iframe -

driver.switchTo().frame(index/id/name/webelement);

- index
- id/name
- webelement

how to back switch

driver.switchTo().defaultContent();
driver.switchTo().parentFrame();

Selenium methods -

Locators
Dropdrop
iframe
Action Class
Screenshot

Popup
Waits
Excel handling

---

Date - 22 Feb 24

Actions Class -

action - Interface

mouse actions
- right click
- double click
- mouse hover
- drag and drop

keyboard actions
- ENTER
- etc...

How to handle

1. Create object of Actions class -

Actions act = new Actions(driver);

2. methods

- right click >> act.contextClick(element);
- double click >> act.doubleClick(element);
- mouse hover >> act.moveToElement(element);
- drag and drop >> act.dragAndDrop(ele1, ele2);

build().perform();

==========

Date - 23 Feb 24

clickAndHold - src
moveToElement - des
release - des

double a = 20.2;

int b = (int)a;

How to capture Screenshot

1. typecasting -

TakesScreenshot ts = (TakesScreenshot)driver;

2. use getScreenshotAs();

File src =  ts.getScreenshotAs();

3.

File des = new File("path");

4.

FileHandler.copy(src,des);

==========

Date - 26 Feb 24

Popup -

display on my webpage with some information when we perform operation on web element

Types

Selenium support
1. Alert popup
2. window popup

Selenium don't support >> Need to use third party plugins like AutoIT, ATIOS
3. file upload
4. file download
5. authentication popup

Alert -

An Alert is a small message box which appears on screen to give the user some information or notification.

We cannot drag and drop and contains symbol or ?
To handle this popup we need to change focus of selenium from main page to alert window popup

Types -

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

info - getText();
OK - Yes - accept();
Cancel - No - dismiss();
textbox - sendKeys("text");

driver.switchTo().alert();

2. window popup / Child window popup -

- driver.switchTo().window(ID);

how toi get ID ?

1. getWindowHandle();  - only one id
2. getWindowHandles();  - all ids of open window

78A5763271D5ECA52452A0848CFEB12D
[78A5763271D5ECA52452A0848CFEB12D, 1AA073A1BB185B2EA483DC4850A3E973]
78A5763271D5ECA52452A0848CFEB12D  - 0  parent id
1AA073A1BB185B2EA483DC4850A3E973  - 1  child id

Assignment -

1. Popup handling - https://stqatools.com/demo/Alerts.php
2. window handling - https://www.myntra.com/

---

Date - 28 Feb 24

Excel Handling -

.xlsx
.xls

interfaces -

Workbook
Sheet
Row
Cell

implemented class

| .xlsx | .xls |
| --- | --- |
| XSSFWorkbook | HSSFWorkbook |
| XSSFSheet | HSSFSheet |
| XSSFRow | HSSFRow |
| XSSFCell | HSSFCell |

How to install Apache poi

1. visit https://archive.apache.org/dist/poi/release/bin/poi-bin-5.2.3-20220909.zip
2. download 5.2.3 poi in zip folder
3. unzip it and store it into specfic folder

How to add jar files -

1. Right click on project >> properties >> java build path >> classpath >> add external jars >> select all jars and apply

Step -

1. create object of File class
```
File file = new File(path);
```

2. create object of FileInputStream class -
```
FileInputStream fis = new FileInputStream(file);
```

3. create object of Workbook
```
XSSFWorkbook wb = new XSSFWorkbook(fis);
```

4. get sheet
```
XSSFSheet sheet = wb.getSheet("Sheet1");
```

5. get row
```
XSSFRow row = sheet.getRow(0);
```

6. get cell
```
XSSFCell cell = row.getCell(0);
```

7. get cell value

String data = cell.getStringCellValue();

==========

date - 01 March 24

Synchronisation/Selenium Waits:-

Matching selenium script speed with browser/application speed called as Synchronisation

Types of waits

1. Implicit wait - webpages
2. Explicit wait - webelements

1. Implicit wait - webpages

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

- It is used when we know exactly how much time will take to load page.
- Webdriver will wait for certain amount of time and if not found that web element it will throws an exception like nosuchelementexcpetion.

Notes -

➢ The implicit wait will apply globally to each web element from webpage.
➢ We know exect time and which will be provided from deveplors
➢ Here we are using compile time polymorphism

2. Explicit wait - webelements

WebDriverWait wait = new WebDriverWait(driver,Duration.ofSeconds(time));

wait.until(ExpectedConditions.methodname(locator));

Alert - alertIsPresent();
link or button - elementToBeClickable();
text - visibilityOfElementLocated();

Compulsory Assignments Question - Complete it before Monday session >>>

To complete assignment you can use any website -

1. Write selenium code to handle alerts and window popups in selenium
2. Read and Write data from excel using POI. Use Excel utils
3. Write selenium code for implicit and Explicit wait (at least 3 methods)

==========

Date - 04 March 24

Advanced Automation -

1. POM
2. Maven
3. TestNG
4. Framework - TDD
5. Cucumber
6. Framework - BDD
7. Git and Github

==========

1. POM -

Page Object Model, also known as POM,

1. For each web page in the application, there should be a corresponding Page Class.
2. These page classes contain all the web elements of that specific web page and action methods which perform operations on those WebElements.
3. We are going to use the encapsulation concept in which data members should be private and methods (member functions) as public.
4. We are going to create two packages
    - Page Layer - It contains web elements(Object Repository) and action methods
    - Test Layer - Test cases

Login testcases with valid cred

1. open browser and paste url
2. click on login link
3. enter username, enter password and click on login button
4. verify user logged in

methodnmae -

button - clickOn..Button
link - clickOn..Link
enter - enter...

```
WebElement login_link = driver.findElement(login_link);
login_link.click();
```

POM -

1. Simple pom

```
// ------------ Obj repo --------------------------

private By login_link = By.xpath("//a[@id='login2']");


// ------------------ action methods ----------------------
public void clickOnLoginLink()
{
        driver.findElement(login_link).click();
}
```

2. Page factory

```
@FindBy(xpath = "")
WebElement login_link;

public void clickOnLoginLink()
{
        login_link.click();
}
```

==========

Date - 06 March 24

Steps

1. File >> New >> Project >> Maven >> Maven Project >> Next
2. check create simple project >> Next
3. Enter groupid and artifact id(project name) >> Finish

Maven -

Maven is software project management tool and build automation tool.
It is hosted by the Apache Software Foundation in 13 July 2004. It was begin with the Jakarta Project.
Maven projects are configured using the POM (Project Object Model), which is stored in pom.xml file.
Maven dynamically downloads the Java libraries, maven plugins from one or more repositories and store them in local cache.

Adv -

To build the project
To manage the project
To define the project structure
Test management
Dependency build

Project structure

src/main/java
paqelayer - pom
src/test/java
testlayer - testcases

POM
POM statnds for Page Object Module.
Page Object Module is a design pattern, used to create object repository for storing all web elements and there action methods.
For each web page in the application, there should be a corresponding Page Class.
These page classes contain all the web elements of that specific web page and action methods which perform actions on those WebElements.
We are going to use the encapsulation concept in which data members should be private and methods (member functions) as public.
Create two packages
Page Layer - It contains web elements (Object Repository) and action methods
Test Layer - Test cases

Note -
Variable naming convention -
webelement_link
webelement_textbox
webelement_button
Method naming convention
clickOnElementButton()
clickOnElementLink()
enterTextInWebElement()

Advantages -
Reusability - reusing code - We can reuse the page class if required in different test cases which means we don't need to write code again.
Maintainability - Test case and page class are different from each other which means we can easily update the code if any new web element is
added or existing one updated.
Readability - Page code is separated from test code which helps to improve code readability.

Page Factory in Selenium
Page Factory is a class provided by Selenium WebDriver to support Page Object Design patterns.
In Page Factory, we are going to use
initElements method - to initilize web element
@FindBy annotation - to find web elements
@FindBy: An annotation used in Page Factory to locate and declare web elements using different locators.
@FindBy(locator="value")
WebElement element;
Below are locators that can be used:
Id, name, className, xpath, TagName, LinkText, PartialLinkText

initElements(): initElements is a static method in Page Factory class. Using the initElements method, one can initialize all the web elements located by @FindBy annotation.

lazy initialization: AjaxElement Locator Factory is a lazy load concept in Page Factory.
This is used to identify web elements only when they are used in any operation or activity.
The timeout of a web element can be assigned to the object class with the help of the AjaxElementLocatorFactory.

==========

Date - 08 March 24

What is testNG

- TestNG is an open source automation testing framework where NG stands for "Next Generation".

- TestNG is inspired from JUnit and NUnit
- It provides multiple rich features for testing like assertions, reporting, parallel test execution, etc.
- The creator of TestNG is Cedric Beust.

Why to use TestNG -

- HTML Reports
- Annotations made testers life easy - @
- Testng have special keyswords/Parameters
- Parallel testing is possible
- Generates Logs through listeners
- Data Parameterization is possible
- No need of main method

```
@Test
public void test()
{
        syso("Hello Wolrd");
}
```

Install testng ?

Version - 7.9.0

1. Visit https://mvnrepository.com/ then search testng
2. Copy 4 line of code and paste in pom.xml

TestNG plugin -

- click on the Help option on the menu bar.
Choose the option “Eclipse Marketplace…”
type TestNG in the search box and click the Go button or press enter key.
Now click on the install button to install TestNG.
As soon as you click on the Install button, a new window will be open for feature selection but you do not have to change anything.
Just click on the “Confirm” button.
In the next step, click on “I accept the terms of the license agreement” and then click on the “Finish” button.
After installation, restart Eclipse

Write first script in Testng -

1. Write below code

```
@Test
public void abcd()
{
        sop("This is abcd method");
}
```

2. add testng libary - by mouse hover on @Test
3. Import dependency
4. Run as TestNg Test

keyswords/Parameters -

```
@Test(keyword = value)
public void abcd()
{
        sop("This is abcd method");
}
```

1. priority

```
@Test(priority = value)
public void abcd()
{
        sop("This is abcd method");
}
```

-5000 to 5000

2. enabled = false

3. invocationCount = int

4. dependOnMethods = {"methodname"}

Annotations -

TestNG Annotation is a piece of code which controls flow of execution of test methods

"@"

types

@Beforesuite
@Beforetest
@Beforeclass
@Beforemethod
@Aftersuite
@Aftertest
@Afterclass
@Aftermethod
@Test

==========

before suite method
before test method
beforeclass method
---
before method
test1 method
after method
---
before method
test2 method
after method
---
afterclass method
after test method
after suite method

Test suite -

the collection of TestNG Tests together is called a Test Suite.
A test suite can run multiple tests at once by executing the test suite

How

1. right click on src/test/java
2. TestNg >> convert to testng
3. finish

==========

Date - 12 March 24

Listeners -

TestNG listeners are the piece of code that listens to the events occurring in the TestNG
- It is an interface that modifies the TestNG behavior

Types Of Listeners In TestNG -

1. ITestListener
2. IReporter
3. ISuiteListener
4. IInvokedMethod

start

pass
fail
skip

We are going to implement ITestListerner to track events

1. create class and Implemnts ITestListerner
2. Right click On Class >> Source >> Implement methods >> Select >> OK

config - in testng.xml

```
<listeners>
    <listener class-name="packagename.classname"/>
</listeners>
```

==========

Date - 13 March 24

1. define
2. Adv
3. Types
4. Intro
5. Maven -
    - intro
    - project structure
        - pom.xml
        - src/main/java -
            - pagelayer
            - testbase
            - utility
        - src/test/java -
            - testlayer - testng
        - folder -
            - ss
            - testoutput
            - resources
6. Testng
    - def
    - features
    - kyword
    - ano
    - listeners
8. Extent report

==========

Date - 14 march 24

www.amazon.com

com.amazon.pagelayer
com.amazon.testlayer

com.opencart.pagelayer

Extent report

1. Add dependency - 2.41.2
2. add Code
3. Config - testng.xml

Log

1. perm
2. temp

log4j

1. Add dependency - 1.2.17

2. Add Code - file.properties
3. Config - Testbase

==========

Date - 21 March 24

Working dir
Git - Local repo
Github - Remote Repo

add & Commit - Working to local
push - local to remote
clone/pull - Remote to local
import - Local to working

1. Create account on Github.com
2. Donwload and Install Git

Github
Gitlab
Bitbucket

1. pwd
2. git --version
3. cd dir
4. clear

---

1. Create repo
2. Clone - at specific folder
3. Navigate git to that folder
4. git status
5. file created
6. git add -A
7. git commit -m "message"
8. git push

9. git checkout -b branchname

---

1. Create repo on github
2. take git clone
3. after clone, change dir >> repo >> main branch
4. Change repo as workspae
   switch workspace >> select path uptop repo >> Launch
5. Create project - (maven)

Framework Implementation Steps

1. Create maven project
2. Delete default packages
3. Add dependencies
4. Create project structure -

Conflicts resolution -

==========

Date - 25 March 24

1. Conflicts resolution
2. Excel in framework
3. Jenkins
4. Cucumber
5. BDD Framework

Cucumber -

TDD >> Code - testcases
BDD >> testcases - code

BDD - Behavior Driven Development

1. Tests are written in plain descriptive English
2. Tests are explained as behavior of application

Features -

1. Collaboration between Business stakeholders, Business Analysts, QA Team and developers
2. it is easy to describe
3. Shifting from thinking in "tests" to thinking in "behavior"
4. BDD frameworks such as Cucumber or JBehave are an enabler, acting a "bridge" between Business & Technical Language

Selenium Test Script
Launch the Browser
Navigate on the LogIn page
Enter UserName and Password
Click on login button
Print a successful message
Close the Browser

Gherkin

Feature: Login Functionality

Scenario: verify login with valid cred

Given user launch the browser
When user navigate on the login page
And user enter username and password
And user click on login button
Then user get successful message
And

//*[@id="login2"]

Chropath
selector hub

Cucumber Topic -

1. TDD - Testng
2. BDD - Cucumber
3. Basic testcases execution - features, step and runner files
4. Cucumber Options
5. Gherkins
6. Data driven testing
    hardcode
    direct approach
    Scenraio Outline
    Data Tables
7. Hooks
8. Background
9. Tags

---

==========

Date - 29 March 24

3 Phases

1. Implementation

2. Execution
3. Maintainance

testng.xml

Execution

1. Run testcases with maven pom.xml
2. Run testcases with cmd
3. Run testcases with run.bat
4. Run testcases with jenkins from local machine

---

5. Run testcases with jenkins - from github

Jenkins -

1. Run testcases with maven pom.xml

Steps

1. open pom.xml
2. add plugins -
   compiler plugin - It help us to compile the code
   surefire plugin - It help us to execute the code

```
<plugin>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.8.0</version>
</plugin>
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>2.22.1</version>
  <configuration>
    <suiteXmlFiles>
      <suiteXmlFile>testng.xml</suiteXmlFile>
    </suiteXmlFiles>
  </configuration>
</plugin>
```

3. after adding plugin - maven update
4. Run as >> Maven Test

2. Run testcases with cmd without using eclipse

Maven
Eclipse Plugin
OS

How to install maven

1. Search maven download - https://dlcdn.apache.org/maven/maven-3/3.9.6/binaries/apache-maven-3.9.6-bin.zip
2. Extract
3. Add this folder to C drive in Program files
4. add env variables
   search env >> variables >> System Properties >> Env variables

   MAVEN_HOME C:\Program Files\apache-maven-3.9.5
   M2_HOME C:\Program Files\apache-maven-3.9.5

   path - C:\Program Files\apache-maven-3.9.5\bin
5. to check - execute mvn -version

ALL GOOD

Steps

1. Open cmd
2. Navigate cmd to project folder use cd foldername
3. execute - mvn test

3. Run testcases with run.bat (window - rat >> mac - sh)

1. Open project location
2. Create one text doc and then rename as run.bat >> Accept popup i.e. Yes
3. Navigate cmd to project folder use cd foldername
   cd path
   mvn test
4. on double clisk - execution will start