**SELENIUM-Automation**

***DAY-1*** *(10-feb-2020)*

**When Automation?**

Automation is costly, Automation test case design takes lot more Time.. But Execution is faster

Frequency and volume- how frequent Regression?

**Selecting Tool..**

Tools can be for stand-alone app, web app, mobile app etc.

**Categories:**  
**\*Test management Tool –** Used to manage test cycle, test planning, test strategy, test cases, metrics etc. example: Jira, QA complete, RTM (IBM)

\***Functional / Regression:** Tools which perform actual testing, can be integrated to TMT. Example: microfpcus-UFT, microfocus-SilkTest, IBM- RFT (All are **commercial** and can support all type). Selenium, Appium, Jbehave, Cucumber (**Open Source,** Selenium only for web testing but third party app can help)

**\*Performance Testing Tool:** ex: load Runner, Silk Runner, Jmeter Etc.

**Life cycle of Automation Test:**

1. Tool Support for application(AUT):

Every object under test should belong to the standard class.(Ok.click(), uname.type()). **ClassName🡪Label🡪pattern class**

(Button) (ok) (password)

1. Import Libraries
2. Identify Object(spy, inspection etc.) find which object to which class
3. Design Test (Contains 3 major fields, steps, data, expected o/p)
4. Run –check whether the test script can identify proper object and execute
5. Synchronization: time sync between entering login screen and loading of login fields. **WAIT UNTIL SOME CONDITION**
6. Verification: expected Outcome is met or not. Assertion/checkpoint
7. Data Driven Testing: Data source is external to test Case. No hard coding of Data
8. Exception Handling:

Automation testing is also called as 24/7 testing, Night Testing etc..

1. Test Suite/ Build Run: from here defect management and documentation, reporting etc. are carried out.

**SELENIUM COMPONENTS:**

**Different versions and enhancement history**

1. Selenium IDE: Earlier IDE was Add on to FireFox saved within .html
2. Selenium RC: (Version 1.0)

\*gave vast programming support.

\*Had no ide

\*Browser could be ie, ff, chrome etc

\***Problem with architecture: RC SERVER was essential between Browser and Test case Program (Java, ruby, x, y, z).** Which converted native languages to JS. No **Parallel execution**

1. Selenium Grid: Supported parallel execution on different platforms, cross browsers with the help of grid. Still RC server was Essential
2. WebDriver (2.0) : Separate web driver for every browser. Hence RC server was removed.

Web driver could not automate native apps

3) Web Driver 3.0:

**Check list:**

**1)JDK 10+**

**2)Eclipse Photon+**

**3)Up to date browser**

**4)excel**

**5) jar files:**

**(Refer requirements.pdf)**

**USING LOCATORS:**Set of available locators are:

1. By ID
2. By name
3. By ClassName
4. By xpath:

**Types:**

**1) // Html/body/div[3]/div[2]/button[1] – absolute xpath.**

2) // Syntax for **relative path: Html-tag[@ATTR=’value’]**

//ex: div[@id=’div22’]/button[1]

**// =>first occurence**

3) By **text value: //html-tag[text()=’value’]**

// button [ text() = ’Ok’]

if more than 1 ok’s then

**// (button [ text() =’Ok’])[1]**

**Tips: x-path should be shorter in length!**

**Using xpath we can go from parent to child and child to parent! Which is not possible by CSS**

**Ex 1.1**

<div> 🡨

<input name=’abc’>

<input …………>

</div>

**//input[@name=’value’]/parent::div**

**Ex 1.2**

<div> 🡨

<span>

<div>

<div>

< img id=’logo’>

</div>

</div>

</span>

</div>

**//img[@id=’logo’]/ancestor::div[3]**

**Ex 1.3**

<div name=’toppane’>

<span> <div>

<div>

<div>

🡪 <img>

**//div[@name=’toppane’]/descendent::img[1]**

**Ex 1.4**

<div>

<div>

<div>

<img> 🡨2

<img id=’demo’>

<img>

<img> 🡨1

<img>

**1) // img[@id=’demo’]/following-sibling::img[2]**

**2) // img[@id=’demo’]/preceding-sibling::img[1]**

**NOTE :** Sometimes ids can be dynamic, or may contain partial static attribute values

**🡪**USING **starts-with()** and **contains() :**

<img name=’logo….’>

**//html-tag[starts-with(@attr,’value’)]**

**//img [starts-with (@name, ’logo’) ]**

**//img [contains (@name, ’logo’) ]**

**NOTE: Using // for descendent**

1. By CSS Selectors:

<div>

<button id=’user’>

<img> 🡨

<button>

<img>

In CSS **@** is not used

For child **>** is used

**Button[id=’user’]>img**

In CSS SELECTOR **# 🡪 id ; . 🡪 class ^ 🡪 starts with; $ 🡪ends; \* 🡪 contains**

**Ex 1: Ex 2:**

<button id=’user’> <button class=’form’>

<img > <button>

**button#user>img div[class=’form’]>button**

**div.form>button**

**Ex 3:**<div class=’reg form’>

<img>

**div.reg.form>img**

**In such case ( more than 1 class) don’t use ‘xpath’ why!?? Ans: By experience**

<div id=’mainpage…….’ >

<img>

**div[id^=’mainpage’] img** (Note: space indicates descendent )

**Ex 4:**

<div name=’tf’>

<ul>

<li>

<li> 🡨

<li>

<li>

**div[name=’tf’] li => all descendent**

**div[name=’tf’] li:first-of-type**

**div[name=’tf’] li:last-of-type**

**div[name=’tf’] li:nth-of-type(2)**

**Ex 5:**

<a href=’……..mail.com’>login</a>

//a[text()=’login’]

**//a[contains(@href,’mail.com’)]**

1. By linkText

**NOTE: Link-text() & Partial-link-text()**

1. By PartiallinkText

(Explore)

1. By TagName:

**Used while working with multiple elements. To get count etc..**

**\*\*\*\*\*\*\*\*\*\*GET HANDS DIRTY!!!!\*\*\*\*\*\*\*\*\*\*\***

**DAY 2 (***11 Feb 2020) Creating Test Scripts in WebDriver*

Create a **Web deriver**, Using WebDriver **Find Element** function to locate and select **WebElement**

**Without TestNG and using main**

**package** day1;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.chrome.ChromeDriver;

**public** **class** First {

**public** **static** **void** main(String[] args) **throws** InterruptedException {

// **TODO** Auto-generated method stub

WebDriver d;

String url="http://google.com",title,fn;

fn="C:\\Users\\karb1\\Downloads\\Selenium training\\chromedriver\_win32\\chromedriver.exe";

System.*setProperty*("webdriver.chrome.driver", fn);

d=**new** ChromeDriver();

d.manage().window().maximize();

d.get(url);

WebElement searchBox=d.findElement(By.*name*("q"));

searchBox.sendKeys("Selenium");

Thread.*sleep*(1000); //There we have two buttons where one is hidden

d.findElement(By.*name*("btnK")).click();

System.***out***.println(d.getTitle());

d.findElement(By.*cssSelector*("div#rso h3")).click();

System.***out***.println(d.getTitle());

}

}

**TESTNG**

Better code notations, reporting and PARALLEL execution. It also supports Annotations (has more annotations than Junit).

@BeforeMethod – Before Each Test case

@BeforeTest – very beginnings and only once

@BeforeClass @BeforeSuite etc.

**Refer:** <https://stackoverflow.com/questions/30587454/difference-between-beforeclass-and-beforetest-in-testng>

**package** day1;

**import** org.testng.annotations.AfterMethod;

**import** org.testng.annotations.AfterTest;

**import** org.testng.annotations.BeforeClass;

**import** org.testng.annotations.BeforeMethod;

**import** org.testng.annotations.BeforeTest;

**import** org.testng.annotations.Test;

**public** **class** TestNGDemo {

@BeforeTest

**public** **void** bt() {

System.***out***.println("Before Test");

}

@BeforeClass

**public** **void** bc() {

System.***out***.println("Before Class");

}

@BeforeMethod

**public** **void** bm() {

System.***out***.println("Before Method");

}

@Test

**public** **void** tc02() {

System.***out***.println("Test 01");

}

@Test

**public** **void** tc01() {

System.***out***.println("Test 01");

}

@AfterMethod

**public** **void** am() {

System.***out***.println("After Method");

}

@AfterTest

**public** **void** at() {

System.***out***.println("After Test");

}

}

Output:

*Before Test*

*Before Class*

*Before Method*

*Test 01*

*After Method*

*Before Method*

*Test 01*

*After Method*

*After Test*

*PASSED: tc01*

*PASSED: tc02*

Test cases are executed in alphabetical order by default. To prioritize we have property priority

**Some properties with Annotation:**

* **@Test (priority=value)** lower the value higher the priority
* **@ignore** ignores the test case
* **@Test(dependsOnMethod=”tc03”)** makes test ‘tco3’ execute first and this test is executed only if tco3 passes

**SUNCHRONIZATION:**

Need of Sync

How?

**Type of waits:**

1. **Implicit wait:** Used when element is not found By exception rise, Defined **only once**.

1. **Explicit wait:** conditions involving **‘Until’.** Until page loads, Until 3 tabs are opened etc..

**package** day1;

**import** java.util.concurrent.TimeUnit;

……………… ……………………………………………… …………..

**import** org.testng.annotations.BeforeTest;

**import** org.testng.annotations.Test;

**public** **class** GoogleEx {

WebDriver d;

String url="http://google.com";

**public** WebDriverWait wait;

@BeforeTest

**public** **void** openBrowser() {

String fn="C:\\Users\\karb1\\Downloads\\Selenium training\\chromedriver\_win32\\chromedriver.exe";

System.setProperty("webdriver.chrome.driver", fn);

d=**new** ChromeDriver();

d.manage().window().maximize();

d.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

wait= **new** WebDriverWait(d,20);

}

@Test

**public** **void** googleSearch() **throws** InterruptedException {

d.get(url);

wait.until(ExpectedConditions.titleContains("Google")); //Explicit wait

WebElement searchBox=d.findElement(By.name("q"));

searchBox.sendKeys("Selenium");

//searchBox.sendKeys(Keys.ENTER);

//Thread.sleep(1000);

WebElement btn= d.findElement(By.name("btnK"));

wait.until(ExpectedConditions.elementToBeClickable(btn));

btn.click();

System.out.println(d.getTitle());

d.findElement(By.xpath("//div[@id='rso']//h3")).click();

System.out.println(d.getTitle());

}

@AfterTest

**public** **void** closeBrowser() {

d.quit();

}

}

**Working With AJAX:**

**package** day1;

**import** java.util.concurrent.TimeUnit;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**public** **class** AjaxEx {

WebDriver d;

String url="https://www.w3schools.com/xml/ajax\_intro.asp";

**public** WebDriverWait wait;

@BeforeTest

**public** **void** openBrowser() {

String fn="C:\\Users\\karb1\\Downloads\\Selenium training\\chromedriver\_win32\\chromedriver.exe";

System.*setProperty*("webdriver.chrome.driver", fn);

d=**new** ChromeDriver();

d.manage().window().maximize();

d.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

wait= **new** WebDriverWait(d,20);

}

@Test

**public** **void** ajaxTest() {

d.get(url);

WebElement ajaxelem=d.findElement(By.*id*("demo"));

System.***out***.println("Before: "+ajaxelem.getText());

d.findElement(By.*xpath*("//div[@id='demo']/button")).click();

wait.until(ExpectedConditions.*not*(ExpectedConditions.*textToBePresentInElement*(ajaxelem, "Let AJAX change this text")));

System.***out***.println(ajaxelem.getText());

}

}

**Working with ASSERTs:**

**🡪 Hard Asserts** (if fails current test case will be aborted, no need of creating object to implement methods)

🡪 **Soft Asserts:**

WebDriver d;

String url="https://www.google.com/";

**public** WebDriverWait wait;

@BeforeTest

**public** **void** openBrowser() {

String fn="C:\\Users\\karb1\\Downloads\\Selenium training\\chromedriver\_win32\\chromedriver.exe";

System.*setProperty*("webdriver.chrome.driver", fn);

d=**new** ChromeDriver();

d.manage().window().maximize();

d.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

wait= **new** WebDriverWait(d,20);

d.get(url);

}

@Test

**public** **void** test() {

Assert.*assertEquals*(d.getTitle(), "Google");

System.***out***.println("Pass");

}

@Test

**public** **void** verifyLogo() {

WebElement logo=d.findElement(By.*id*("hplogo"));

Assert.*assertTrue*(logo.isDisplayed());

System.***out***.println("Pass");

**Verifying Attribute value**

@Test(enabled=**true**)

**public** **void** verifyGmail() {

WebElement gmail=d.findElement(By.*linkText*("Gmail"));

Assert.*assertTrue*(gmail.**getAttribute("href").**contains("mail.google.com"));

}

**Day 3** (12 Feb 2020)

**3.1 Working with multiple Elements:**

@Test

**public** **void** countSuggestions() {

openHome("http://google.com");

String s="Heloo";

driver.findElement(By.*name*("q")).sendKeys(s);

List<WebElement> suggestions =driver.findElements(By.*xpath*("//ul[@role='listbox']//li//span"));

**for**(WebElement sug:suggestions) {

System.***out***.println(sug.getText());

}

}

**3.2 Working with tables:**

Using xpath accessing the row X coloumn Element. Note that tables are usually dynamic. Avoid using row nums and col nums.

| **Tool Name** | **Tool Type** | **Vendor** | **Language** | **WebSite** |
| --- | --- | --- | --- | --- |
| Selenium | Functional Testing | OpenSource | Java | http://www.seleniumhq.org | [EDIT](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#EDIT) [DELETE](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#DELETE) |
| QuickTestPro | Functional Testing | HP | VBScript | http://hp.com | [EDIT](file:///C:\\Users\\karb1\\Downloads\\Selenium%20training\\1407405934WebTable.html" \l "EDIT) [DELETE](file:///C:\\Users\\karb1\\Downloads\\Selenium%20training\\1407405934WebTable.html" \l "DELETE) |
| LoadRunner | Performance Testing | HP | ANSI C | http://hp.com | [EDIT](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#EDIT) [DELETE](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#DELETE) **🡨** |
| QualityCenter | Test Management | HP | NA | http://www.hp.com | [EDIT](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#EDIT) [DELETE](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#DELETE) |
| TestComplete | Functional Testing | SmartBear | C# | http://smartbear.com | [EDIT](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#EDIT) [DELETE](file:///C:\Users\karb1\Downloads\Selenium%20training\1407405934WebTable.html#DELETE) |

**$x ("//td[text()='LoadRunner']/following-sibling::td/a[text()='EDIT']")**

@Test

**public** **void** tc\_Table01() {

openHome("file:///C:/Users/karb1/Downloads/Selenium%20training/1407405934WebTable.html");

WebElement tb= driver.findElement(By.*xpath*("//table[@id='table1']//tbody"));

List <WebElement> ls=tb.findElements(By.*tagName*("tr"));

**for**(WebElement row :ls) {

List <WebElement> col= row.findElements(By.*tagName*("td"));

**for**(WebElement c:col) {

**if**(c.getText().equals("LoadRunner")) {

System.***out***.println("Row: "+ls.indexOf(row));

row.findElement(By.*xpath*("//td/a[text()='EDIT']")).click();

System.***out***.println(driver.getCurrentUrl());

}

}

}

}

**3.3 Working with Drop Down Selection:**

Using Select Class

**Select class can select multiple element using implicit methods!**

@Test

**public** **void** select1() {

openHome("file:///C:/Users/karb1/Downloads/Selenium%20training/1501486869DropDown.html");

WebElement drop1=driver.findElement(By.*id*("Drop01"));

Select s1=**new** Select(drop1);

s1.selectByVisibleText("Business");

System.***out***.println(drop1.getAttribute("value"));

}

@Test

**public** **void** select2() {

openHome("file:///C:/Users/karb1/Downloads/Selenium%20training/1501486869DropDown.html");

WebElement drop2=driver.findElement(By.*id*("Drop02"));

Select s2=**new** Select(drop2);

s2.selectByValue("Bus");

s2.selectByIndex(2);

s2.selectByVisibleText("Taxi");

System.***out***.println();

}

**3.4 Hover vs Click!**

An Action Class need to be used

**Action act=new Action(driver);**

**Action()** Can be used only on **driver**

@Test

**public** **void** Move() {

openHome("https://www.naukri.com/");

WebElement jobs=driver.findElement(By.*className*("mTxt"));

Actions act=**new** Actions(driver);

act.moveToElement(jobs).perform();

driver.findElement(By.*linkText*("Jobs by Skill")).click();

}

**3.5 Copy – Paste**

@Test

**public** **void** copypaste() **throws** InterruptedException {

openHome("http://google.com");

WebElement searchBox=driver.findElement(By.*name*("q"));

searchBox.sendKeys("Sapient Test");

Actions act = **new** Actions(driver);

act.keyDown(Keys.***CONTROL***).sendKeys("a").keyUp(Keys.***CONTROL***).perform();

Thread.*sleep*(1000);

act.keyDown(Keys.***CONTROL***).sendKeys("c").keyUp(Keys.***CONTROL***).perform();

Thread.*sleep*(1000);

searchBox.clear();

searchBox.click();

act.keyDown(Keys.***CONTROL***).sendKeys("v").keyUp(Keys.***CONTROL***).perform();

}

**3.6 Data Driven Testing/ Parameterized Testing:**

**Reading from Excel File:**

**Ex 1**

@Test

**public** **void** readFromExcel() **throws** Exception

{

String xlFile = "TestData/Testingdatademo.xlsx";

FileInputStream fileIn = **new** FileInputStream(xlFile);

XSSFWorkbook wb = **new** XSSFWorkbook(fileIn);

XSSFSheet ws = wb.getSheetAt(0);

**int** rc = ws.getLastRowNum()+1;

System.***out***.println("Row count: "+rc);

**int** cc = ws.getRow(0).getLastCellNum();

System.***out***.println("Column count: "+cc);

**for**(**int** i=1;i<rc;i++)

{

XSSFRow row = ws.getRow(i);

**for**(**int** j=0;j<cc;j++)

{

XSSFCell cell = row.getCell(j);

String cellValue = cell.getStringCellValue();

System.***out***.println(cellValue);

}

}

}

**Data provider using TestNG:**

@DataProvider() 🡪

@DataProvider(name="dp")

**public** Object[][] readFromExcel() **throws** Exception

{

String xlFile = "TestData/Testingdatademo.xlsx";

FileInputStream fileIn = **new** FileInputStream(xlFile);

XSSFWorkbook wb = **new** XSSFWorkbook(fileIn);

XSSFSheet ws = wb.getSheetAt(0);

**int** rc = ws.getLastRowNum()+1;

**int** cc = ws.getRow(0).getLastCellNum();

Object[][] xlData=**new** Object[rc-1][cc-1];

**for**(**int** i=1;i<rc;i++)

{

XSSFRow row = ws.getRow(i);

**for**(**int** j=0;j<cc;j++)

{

XSSFCell cell = row.getCell(j);

String cellValue = cell.getStringCellValue();

xlData[i-1][j]=cellValue;

}

}

wb.close();

**return** xlData;

}

@Test(dataProvider="dp",dataProviderClass=MyDps.**class**)

**public** **void** login(String x,String y) **throws** Exception {

openHome("http://softest-training.com/");

driver.findElement(By.*xpath*("//div[@class='navigation']//li[8]")).click();

Thread.*sleep*(1000);

driver.findElement(By.*name*("username")).sendKeys(x);

driver.findElement(By.*name*("password")).sendKeys(y);

Thread.*sleep*(1000);

driver.findElement(By.*id*("submit")).click();

driver.findElement(By.*xpath*("//a[@class='text12']")).click();

}

**DAY 4** *(13 Feb 2020)* **Advanced Selenium**

**Removing hard coded dependencies: Data Driven Test**

**\*\*PRACTICE\*\***

|  |
| --- |
| Reading from Excel file |
| Reading from csv/Database/properties file |
| TestNG DataProvider |
| Hands-on Assignments |

**Help files:** <https://github.com/karthik-bhat98/SeleniumExcelTraining>

**Handling JavaScript Alerts:**

3 Types of alert box may pop up in a web application.

[https://github.com/karthik-bhat98/SeleniumExcelTraining - POPUP.HTML](https://github.com/karthik-bhat98/SeleniumExcelTraining%20-%20POPUP.HTML)

Alert a=driver.switchTo().alert();

**Handling frames!:**Frames needs to be identified and driver is switched to that frame document.

* driver.switchTo().frame(frame);

@Test(enabled=**true**)

**void** testLogin() **throws** Exception {

openHome("file:///C:/Users/karb1/Downloads/Selenium%20training/1521781750IframeExample.html");

WebElement topframe=driver.findElement(By.*id*("Frame1"));//

driver.switchTo().frame(topframe);

driver.findElement(By.*linkText*("Login")).click();

Thread.*sleep*(1000);

driver.findElement(By.*name*("username")).sendKeys("user");

driver.findElement(By.*name*("password")).sendKeys("pass");

Thread.*sleep*(1000);

driver.findElement(By.*id*("submit")).click();

driver.findElement(By.*xpath*("//a[@class='text12']")).click();

}

**Handling Tab!:**

Using window handler. Selenium considers both windows and new tabs as WINDOW

openHome("https://www.naukri.com/");

wait.until(ExpectedConditions.*numberOfWindowsToBe*(3));

String pwh=driver.getWindowHandle();

Set <String> winHandles=**driver.getWindowHandles();**

**for**(String wh:winHandles) {

**driver.switchTo().window(wh);**

System.***out***.println(driver.getTitle());

**if**(!wh.equalsIgnoreCase(pwh))

driver.close();

}

driver.switchTo().window(pwh);

**Working with AutoIT :**

**--------------------------------**

**DAY 5 (**14 - Feb – 2020**):**

**Test Suites:**

Using XML support.

Parallel execution, conditions etc., can be handled

**Day 6**

**Javascript execution**

JavascriptExecutor js= (JavascriptExecutor)driver;

js.executeScript("window.scrollBy(0,5000)");

**JENKINS**

Configuration:

🡪configure system 🡪 check maven directory, git, etc.

**CASE STUDY**