**Selenium Automation testing A16**

**Local host: localhost brings up if any application is hosted on the localmachine.**

**Selenium IDE:**

• / points to base URL

• Toggle breakpoint : puts a pause

• Assert text present – when a step fails

• Echo ${mc1} – capturesvalue in mc1

• Verify text present

**• Xpath syntax** : //htmltag[@attname=’attvalue’]

• //htmltag[@att1=’value’ and @att2=’value’]

• Example :record[field[@id='220' and @value='Red'] and field[@id='221' and @value='Large']]

•

**• Html** : <tr> is a row, in that row, there is a column <td> and alignments and values to it. ‘a’ is a html tag for hyperlinks.

• Selenium identifies the properties uniquely in the following order.

• Locating through direct references:

**• By ID property** (if ID is unique, it puts ID property value in the target else it takes additional properties and put an xpath)

**• By Name property**(if name is unique, it puts name property value in the target else it takes additional properties and put an xpath)

**• By using link** (eg link = “sign out”).( It captures the property before the html tag)

**• Locating through DOM or CSS**

**• By using Xpath**

**Day3: Eclipse set up and Basic code**

**Eclipse:**

• File new java project give a project name click finish.

• Right click src folder under project go to new select either a package or a class select class give a class name and leave the options as default and click finish.

• Delete everything on the class except public class classnamne{

} (make sure this class name matches the class name created) f2 under default package for rename

• Get selenium server(selenium-server-standalone-2.4.jar) from seleniumhq.org

• Junit Jar

• Right click on project properties Java Build path Libraries add external JARs

• Selenium server(jar fi le) – for Eclipse to understand Selenium components

• Selenium Java – connector for Selenium to understand the Java commands

• Junit Jar file – all the components with Junit is understandable by Eclipse.

Add 1 and 2 from above. Run the batch file

• Javadriver: we need eclipse to communicate to the browser as a connector between eclipse and browser.

• Go to cmd cd to the ç (rec ch2 video 1 38 mins)

•

• Port is saying where the communication should happen. Example there is 3 connections with 3 ports. Server is on one site and my computer on a different site. Now I need to establish a channel or port for these to talk.

• After adding those 2 files, the red marks have gone. There are some yellow exclamations. Next to imports. You can remove what is not required. When you highlight the code if you see that the import pattern is not used anywhere, you can remove it.

• Run the test. Run using junit.

• To put comments on the code you can use //

• Putting .\* will import everything from junit. So to make code efficient we only try using what we need.

**Annotations**:

• Create a new class, right click on default package and write your own code. Basic Junit has 3 annotations : before, test, after

• Public class test\_1{

@Before ‘anything that I put here will get executed before every test

@Test

@After ‘anything that I put here will get executed after every test

}

**• Skeleton for any junit testing**

Public class test\_2{

@Before

Public void setUp() throws Exception {

}

@Test

Public void testname() throws Exception {

}

@After

Public void tearDown() throws Exception{

}

}

• Import the annotations for before, test, after

**• Print statements:** system.*out*.println(“Hello this is @before”); ‘ Semicolumn at the end of a java statement is the end of that line.

• You can have multiple @tests with different test names. But the execution happens 1 at a time. Example like below are the prints

Hello, this is @Before

Hello, this is @Testm1

Hello, this is @AFter

Hello, this is @Before

Hello, this is @Test m2

Hello, this is @After

**Day4:**

• Debug is a built in tool which will identify us to find the bugs in our own automated program that we are building now in Selenium

• Debug run will let us execute one at a time. You need to put breakpoints. For example I want to add a breakpoint at line 13, double click before line 13 or right click and select toggle breakpoint’.

**Code**:

Driver = new FirefoxDriver(); This is a java code.

• There are 2 sides of this equation. What you have on the right side is taken and put into the left side of the equation.

• This is where we create a driver object that points to a specific browser. Every browser has specific drivers. The drivers are pointed using this specific class called “FirefoxDriver”.

• So it creates or returns a new object of type firefox driver.

• Example pen is an object, it has properties such as color, plastic, cap, material, ink, heavy etc. properties define this object. Pen is a specific class. Class gives an overall type of an object. Every pen has some similarities such as all pens write, all pens have certain color. All these belong to a specific class. The properties that are used to identify an object and what you can do on them or methods are defined depending on the class of an object.

\*\*\*Properties and methods are the key things that define the class of an object.

• Another example is a car. A car is a class of an object. Cars are made of metal object, some have 4 wheels, 4 doors, 1 passenger seat, all these are properties of a car. Then there are methods to the car, you can drive, park or what you want to do. So set of properties define what the class is. Within the class there is subclass such as 6 wheels etc.

• If I want to take an instance of a class, it becomes an object.

• Objecto1 = new car(); - this is a car, I want to take an instance or define an object. Now this O1 becomes my car

•

• Driver = new FirefoxDriver(); driver is an object of class FirefoxDriver.

• We need to import for the class firefox browser and define the object driver.

• To define an object, I need to give the class name to the object

*• Classname objectname = new classname - syntax for creating object.*

• FirefoxDriver driver = new FirefoxDriver();

Base URL = “http://www.youtube.com”

**Variables**: Variables are set of information that you store in your program. It’s like storing contacts on the phone book on the object called phone. So we name every variable.

• Eg: say v1 = “Priya” - im trying to name a variable and say there is value in this variable, so I can use these values wherever I want.

• But Java says before you start using something, create one or define one.

• Every variable has a specific type.

• There are few defined types. They are

• String ‘alpha numeric. Anything within quotes becomes a string.

• Int ‘ integer. All the positive , negative numbers without decimals

• Array

• Boollean : true or false

• Double - ‘ a number with decimals.

• Short – is a small variable than a integer. It cannot hold large numbers

• Long : Long variables are extended size variables for number storage, and store 32 bits (4 bytes),

• So the syntax is String v1 = “Priya”

• Ctrl+shift+/ = comments selected code.

• To print the variable, system.out.println(v1);

• Instead of declaring all objects and variables in every line, you can declare within 1 annotation

\*\*\* every variable, objects are print are limited only with their annotation such as with @before only works within before, so the best way is to bring it out of all 3 annotations and put it in the public class so it can be used for all

**Methods**:

**Get**:

• If you type the object created, all the methods associated with it appear.

• To get the base url, the method is driver.*get*(baseUrl); - get is the method to get the url

• + icon joins 2 strings together.

• Example

String S1;

String S2;

S1 = “hello”;

S2 = “Team”

System.out.println(S1+S2);

This joins the strings and prints hello team

• So driver.get(baseUrl + “/”); ‘ Selenium webdriver method that navigates the browser to the Url

**Find Element:**

Driver.findelement /(by). Do something;

*• Find the element* and *do something* is very important.

**Quit:**

Driver.quit(); // closes the browser – you can have this in @after where you can close the browser,

@before: specify browsers

@test : find element and do something

@after : quit

**Select: select an option from drop down**

To select by text value you can use text() function. And normalize spaces is required, because they are not removed by default. Here is an example:

select/option[normalize-space(text())="Grass"]

@value - value of "value" attribute

@val - value of "val" attribute

/select/option[.='Grass']

I'd need to check re whitespace, though. You could always normalize:

/select/option[normalize-space(.)='Grass']

normalize-space() - function returns the argument string with whitespace normalized by stripping leading and trailing whitespace and replacing sequences of whitespace characters by a single space

To check the value that is currently selected in a drop down

//select/option[@selected='selected'] - Will match the selected option, i am using this successfully.

//select/option[@selected='selected' and @value='specific value'] - Will only match the selected option if it has a 'specific value', i'm also using this.

**Day 5**

• On you tube,

I/P : qtp

o/p About 5070 results

i/p : selenium

o/p: output has 13300 results

This is a scenario with different inputs and gives different output.

So declare

String iSearchTerm;

String oSearchResults

iSearch term = “QTP”;

• Xpath : id, name are the main attributes

Edit path of youtube:

Syntax : //htmltag[@attname=’attvalue’]

Where part: xpath is //input[@id=’ gbqfq’]

Driver.findelement(by. – when you press. It gives the common paths, most commonly used is xpath

Driver.findelement(by.xpath(where part )) - by is what method.

What part : I need to enter something, so the command is type iSearchterm

Driver.findelement(by.xpath(where part )). Enter text

Driver.findelement(by.xpath(where part )).sendKeys(isearchterm)

Sendkeys is a method to type

Send key takes a string.

• There are 2 types of xpaths

//input[@id=’username’]

//\*[id=’button’]

If it has a \* symbol, it is searching for any html tag as long as the id is button. But we can use specific

• verify Results from youtube. the text with the search results

code is gettext();

• wait command in java : thread.sleep(3000)  -  3000millisecond - 3 secs

you can also declare and use it

int iwait;

iwait = 3000;

thread.sleep(iwait);

you can use this line wherever required.

**• Output results:**

oSearchResults = driver.findelement(by.xpath()).gettext();

system.out.println("the output is" + oSearchresults);

+

**Day6:**

**Loops**

**For Loops:** Helps you repeat a specific lines of code certain number of times.

Syntax:  for (int i=1;  i<5; i=i+1);

**public** **class** forloop {

      @Test

**public** **void** tes1() **throws** Exception{

**for** (**int** i=1; i<5; i=i+1){

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

                  System.*out*.println("hello, the i value is" + i);

            }

      }

}

• The whole above i=i+1 can be replaced by i++

• Implement this in the main test. But if I implement this in the main code with one input, the same activity gets executed 4 times. So declare some more strings

String iSearchTerm, oSeacrhTerm;

• If statement: conditions. So you have 2 sides ‘x’ and ‘y’,, so you compare both sidescheck are they equal, < or >

**If syntax:**

If (i1==i2) {

System.out.println(“Pass”);

} else {

System.out.println(“Fail”);

}

Implement if for int’s

@test

Public Void myiftest() {

String s1, s2;

int i1, i2;

s1 = Priya;

s2 = Selenium;

i1 = 10;

i2 = 20;

If (i1==i2) {

System.out.println(“Pass1”);

} else {

System.out.println(“Fai1l”);

}

If (i1<i2) {

System.out.println(“Pass”);

} else {

System.out.println(“Fail”);

}

}

Depending on the condition pass or fail within the curl brackets , it gets executed

**If statement for Strings**

We cannot compare 2 texts charactes with the symbols. Instead the correct format is

If (s1.equals(s2)){

System.out.println(“Pass”);

} else {

System.out.println(“Fail”);

}

• When verify the results like the results from youtube such as “About 13000 results”, declare an expected result string eSearchResults;

eSearchResults = “About 13000 results”; ‘ spaces are important.

So I use a if statement to implement this

System.out.println(“The Search numbers are :” + oSearchResults);

If (eSearchResults.equals(oSearchResults)){

System.out.println(“Pass”);

}else {

System.out.println(“Fail”);

‘}

}

**• To compose different inputs, do the following**

iSearchTerm = “Selenium”;

string i1,i2,i3, i4;

i1 = “Testing”;

i2 = “Automation”;

i3 = “Training”;

i4 = “Videos”;

for (i=1; I = 4; i++) {

if (i==1){

iSearchTerm = iSearchTerm + i1;

}

If (i==2){

iSearchterm = iSearchterm + i2;

If (i==3){

iSearchterm = iSearchterm + i3;

If (i==4){

iSearchterm = iSearchterm + i4;

}

If (eSearchResults.equals(oSearchResults)){

System.out.println(“Pass” + 1);

}else {

System.out.println(“Fail” + 1);

}

The above code is not giving space between each word.

You give a space as “ “ or give a space after “Selenium “

**• New verification text from youtube**

String iwtext, owtext;

Wtext =” sign in to add channles to your homepage”:

I want to get the test, compare if it’s the same and print.

//1. Get the test from the home page

Driver.get(baseUrl);

Thread.sleep(iwait);

Owtext = driver.findElement(by.(xpath(“xpath”)).getText();

//2. Compare the text

If( iwtext .equalsto(owtext)){

Print pass

//3. print

**DAY 7 – DDF**

**Sample Test Cases**

**Sample Test Data**

**• Reading from Excel**

//Read each set of data (eclipse java reads as front slash)

//String xlPath = “C:\Users\priyap@airloom.com.au\Desktop\ test cases.xls”;

//(eclipse java reads as front slash) or uses 2 back slashes,

String xlPath = “C:/Users/priyap@airloom.com.au/Desktop/test cases.xls”;

//Read xl data – we need Arrays for this

What is an Array? Arrays are nothing but variables

Let’s use array. So I want an array called A16. The series starts with 0,1,2,3..

String A16 []; // defining the variable as an array

A16[0] = “Test1”

A16[1] = “Mon”

A16[2] = “Tues”

A16[3] = “Wed”

String A16 [] = null; // it created an empty array with no data in it.

A16[0] = “Test1”

A16[1] = “Mon”

A16[2] = “Tues”

A16[3] = “Wed”

//how does Arrays execute, for example let me create a for loop

For (int i=0; i<4; i++){

System.out.println(“place “ + I + “is” + A16[i]);

}

This is the basic structure of Array, but this dint work. SO modified to below

1 dimensional Array

@Test

**public** **void** Arraytest() **throws** Exception {

String[] A1 = **new** String[4]; // One dimensional array

A1[0] = "Mon";

A1[1] = "Tue";

A1[2] = "Wed";

A1[3] = "Thur";

**for** (**int** i = 0; i <4; i++){

System.*out*.println("The sequesnce is " + i + " " + A1[i]);

}

}

}

2 Dimensional array:

String [] [] TwoD = **new** String [2] [3];

TwoD[0][0] = "Row1 Col1";

TwoD[0][1] = "Row1 Col2";

TwoD[0][2] = "Row1 Col3";

TwoD[1][0] = "Row2 Col1";

TwoD[1][1] = "Row2 Col2";

TwoD[1][2] = "Row2 Col3";

**for** (**int** i = 0; i<2; i++){

//System.out.println("Row" + (i+1) + " is from " + TwoD[i][0]);

**for** (**int** j = 0; j<3; j++){ //nested for loop

System.*out*.println("i is " + i);

System.*out*.println("j is " + j);

}

**Day8**

**Methods**:

Methods are specific set of activities that can be performed based on the inputs. Example, to a car when you want to put break, multiple things happen. i/p is the method you brake and o/p is it slows down and you could talk about how much time it took to slow down.

How do you define a specific method?

//the basic skeleton of any method is as follows:

**public** **void** Method1() {

}

• In the above Skeleton, Public is the accessibility of Method1 within the other scripts that you want to call.

• We can have public methods and Private

• Void is what did that method return, if there is no return to that method, then this void has no meaning.

**public** **class** Methods {

@Test

**public** **void** mytest() {

System.*out*.println("Hello, this is the main test");

Method1();

Method2("Priya");

Method2("Is Blessed");

Method3("Yes");

String Temp = Method3("Yes"); //the whole mthd3(“yes”) gets replaced with whatever is in return in method3

System.*out*.println(Temp);

}

**public** **void** Method1() {

System.*out*.println("this is my method1");/simple method no i/p no o/p

}

**public** **void** Method2(String name){//passing input 1 i/p 0 o/p

System.*out*.println("my method 2 has got a input called " + name);

}

**public** String Method3(String email){ //mthod 3 is returning 1i/p 1 o/p

System.*out*.println("My Method3 is going to return a value " + email);

**return** "Test is a pass";

}

}

Result:

Hello, this is the main test

this is my method1

my method 2 has got a input called Priya

my method 2 has got a input called Is Blessed

My Method3 is going to return a value Yes

My Method3 is going to return a value Yes

Test is a pass

**• Functions**:

**import** org.junit.Test;

**public** **class** MathFunctions {

@Test

**public** **void** mytest() {

**int** a; **int** b;

a=10; b=10;

System.*out*.println("a is " + a);

System.*out*.println("b is " + b);

System.*out*.println(addF(1,2));

System.*out*.println(addF(2,20));

}

**public** **int** addF(**int** a1, **int** b1) {

**return** (a1+b1);

}

}

Result is

a is 10

b is 10

3

22

**public** **class** MathFunctions {

@Test //to call this function

**public** **void** mytest(){

System.*out*.println(addF(1,2));

System.*out*.println(addF(2,2));

}

**public** **int** addF(**int** a, **int** b) { //method to add

**return** (a+b);

}

}

**Reading test data from xl**

**For Java to read the Excel, it uses APachi POI. Search for APACHI POI jar. Download the latest release version.**

• Declare test data as local variables.

String vTDID, vURL, vSreachterm, vEmail, vPswd;

Load the local variables into arrays

For (int k=1; k<xrows; k++) {

vTDID = myXL[k][0];

vURL = myXL[k][1];

vSreachterm = myXL[k][0];

vEmail = myXL[k][0]

vPswd = myXL[k][0];

Proper function that reads from EXCEL

**import** java.io.File;

**import** java.io.FileInputStream;

**import** org.apache.poi.hssf.usermodel.HSSFCell;

**import** org.apache.poi.hssf.usermodel.HSSFRow;

**import** org.apache.poi.hssf.usermodel.HSSFSheet;

**import** org.apache.poi.hssf.usermodel.HSSFWorkbook;

**import** org.junit.Test;

**public** **class** newread{

//public //declared as a global data

//public int //declaring as a global data

@Test

**public** **void** mytest() **throws** Exception{

String myPath ="C:/Priya/Silverback 4.3/Automation/QService.xls";

String[][] myExel = xlRead (myPath, "TestCases"); //calling function to read Excel

System.*out*.println("some data"+ myExel[1][2]);

}

//Read from Excel

**public** String[][] xlRead(String sPath, String iSheet) **throws** Exception{ // This line xlRead is the call to this function. We are passing the xl path in this

//String[][] myExcel = null;

**int** xRow, xCol;

String[][] xData;

//file, wb,sheets

File myxl = **new** File(sPath); // File is telling Java that it will use I/p o/p streams to interact with Java. The name of the file is myxl here.

FileInputStream myStream = **new** FileInputStream(myxl); // FIle input stream reads i/p and o/p into the file

HSSFWorkbook myWB = **new** HSSFWorkbook(myStream); // Creates workbook within this file.

HSSFSheet mySheet = myWB.getSheet(iSheet); // getSheetAt(0) method - Referring to 1st sheet, 1 -2nd sheet and so on

//Get row count

xRow = mySheet.getLastRowNum()+1; //gets number of rows in my sheet. SInce it starts with 0, adding +1 to it will get total number of rows which is the last row.

xCol = mySheet.getRow(0).getLastCellNum(); // to get cells or number of columns, first go to row 0, then get the last cell number of that row.

xData = **new** String[xRow][xCol]; // we are creating an Array of type string. This puts the number of rows and cols.

System.*out*.println("Rows are " + xRow);

System.*out*.println("Cols are " + xCol);

// Go to each cell

**for** (**int** i = 0; i < xRow; i++) {

HSSFRow row = mySheet.getRow(i); // select and point to a specific row in my sheet it points according to the iteration

**for** (**int** j = 0; j < xCol; j++) {

HSSFCell cell = row.getCell(j); // on that row, get the value of each column of each row.

String value = *cellToString*(cell); //convert a specific cell type(int,string/double etc) to a simple string - this I can get from the function HSSFcell

xData[i][j] = value;

}

}

**return** xData;

}

// A new function to convert an object of type excel cell to a string value.

**public** **static** String cellToString(HSSFCell cell) {

// This function will convert an object of type excel cell to a string value

**int** type = cell.getCellType(); //cell.getcell type returns an integer which

Object result;

**switch** (type) {

**case** HSSFCell.*CELL\_TYPE\_NUMERIC*: //0

result = cell.getNumericCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_STRING*: //1

result = cell.getStringCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_FORMULA*: //2

**throw** **new** RuntimeException("We can't evaluate formulas in Java");

**case** HSSFCell.*CELL\_TYPE\_BLANK*: //3

result = "-";

**break**;

**case** HSSFCell.*CELL\_TYPE\_BOOLEAN*: //4

result = cell.getBooleanCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_ERROR*: //5

**throw** **new** RuntimeException ("This cell has an error");

**default**:

**throw** **new** RuntimeException("We don't support this cell type: " + type);

}

**return** result.toString();

}

}

// Method to write into an XL

**public** **static** **void** writeXL(String sPath, String iSheet, String[][] xData) **throws** Exception{

File outFile = **new** File(sPath);

HSSFWorkbook wb = **new** HSSFWorkbook();

HSSFSheet osheet = wb.createSheet(iSheet);

**int** xR\_TS = xData.length;

**int** xC\_TS = xData[0].length;

**for** (**int** myrow = 0; myrow < xR\_TS; myrow++) {

HSSFRow row = osheet.createRow(myrow);

**for** (**int** mycol = 0; mycol < xC\_TS; mycol++) {

HSSFCell cell = row.createCell(mycol);

cell.setCellType(HSSFCell.*CELL\_TYPE\_STRING*);

cell.setCellValue(xData[myrow][mycol]);

}

FileOutputStream fOut = **new** FileOutputStream(outFile);

wb.write(fOut);

fOut.flush();

fOut.close();

}

}

**• The convert cell to numeric has an issue where it was reading a numeric with a decimal which was fixed with this code below**

**switch** (type) {

**case** HSSFCell.*CELL\_TYPE\_NUMERIC*: //0

String stringValue=""+cell.getNumericCellValue();

String[] splitValue=stringValue.split(".0");

//System.out.println(splitValue[0]);

result = splitValue[0];

**break**;

To print the whole excel

@Test

**public** **void** mytest() **throws** Exception{

String myPath ="C:/Priya/Silverback 4.3/Automation/QService.xls";

//read one line in that excel

String[][] myExel = xlRead (myPath, "TestCases"); //calling function to read Excel

System.*out*.println("some data"+ myExel[1][2]);

//print everything in the excel

**for** ( **int** i = 0; i<xRow; i++){

**for** (**int** j=0 ; j<xCol; j++){

System.*out*.println("some data"+ myExel[i][j]);

}

}

}

**Day9**

**DDF – hard coding the test data,**

**Declare test data as local variables.**

//Create a driver to run automation

webDriver driver = new InternetExplorerDriver(); //import web driver and internet explorer. Download the IEdriver to wherever the works space is pointed to and run the java code with the following:

File file = **new** File("C:/Selenium/iexploredriver.exe");

System.*setProperty*("webdriver.ie.driver", "IEDriverServer.exe");

WebDriver Driver = **new** InternetExplorerDriver();

//Day9

**public** **class** DDF {

**public** **static** **int** *xRow*; **static** **int** *xCol*;

@Test

**public** **void** DDF1() **throws** Exception {

//Declare local variables in the code from the test data.

String vTDID, vURL,vilogin,vipaswd,vqinput,vExecute,vExpectedResults,vActualResults;

//Read each set of data

String xlPath = "C:/Priya/Silverback 4.4/Automation Test Cases/QService.xls";

String [][] myXL = *xlRead*(xlPath, "TestData");

//see what was read from excel.

**for** (**int** i=0; i<*xRow*; i++){

**for** (**int** j=0; j<*xCol*; j++){

System.*out*.println(myXL[i][j]);

}

//create a webdriver to run automation

File file = **new** File("C:/Selenium/iexploredriver.exe");

System.*setProperty*("webdriver.ie.driver", "IEDriverServer.exe");

WebDriver Driver = **new** InternetExplorerDriver();

// Take values from each row of 2 dimensional local array from test data and load them into local variables

**for** (**int** k =1 ; k<*xRow*; k++){

vTDID = myXL[k][0];

vURL = myXL[k][1];

vilogin = myXL[k][2];

vipaswd = myXL[k][3];

vqinput= myXL[k][4];

vExecute = myXL[k][5];

vExpectedResults =myXL[k][6];

vActualResults = myXL[k][7];

//Execute the test

// TC001 //go to Silverback console

Driver.navigate().to(vURL);

//enter the log in

Driver.findElement(By.*xpath*("//input[@id='Username']")).sendKeys(vilogin);

}

//Write into Excel;

*writeXL*("C:/Priya/Silverback 4.4/Automation Test Cases/Results.xls", "TD REsults", myXL);

}

}

}

//Function to READ and WRITE EXCEL.

**Day 10**

**KEYWORD DRIVEN FRAMEWORK**

• How do I read test data into Java code? Apachi POI Jar

Download it from google.

**• DDF – multiple test data into 1-2 test steps.**

**• KDF – multiple test steps into 1 set of test data.**

**• Keywords:** Test steps broken into keywords.

**• Functional Keywords**

**• Low level user interaction keywords.**

**• Functional keywords:**

Eg: banking website

TC1 - log into bank, verify, log out

TC2 – login, transfer, verify and logout

TC3 – login, verify, transfer, verify

In the above, there are logins used in every test case, so make it a keyword.

•

\* [@type='button'])[2] is same as //button[@type='button'])[2]

• Means replace with any html tag on the page. ‘ use it if you don’t know what is the correct html

• Click failed on the test, try using sendkeys(“\n”); ‘ we are sending the enter key to it instead of click

Calling functions

**public** **class** kdf1 {

FirefoxDriver driver;

String baseURL;

@Before

**public** **void** setUp() **throws** Exception{

driver = **new** FirefoxDriver();

baseURL = "http://youtube.com";

driver.navigate().to(baseURL);

}

@Test

**public** **void** TC002() **throws** Exception { //incorrect login

**int** iwait;

iwait = 3000;

fLogin();

String S1 = "//span[@id='errormsg\_0\_Passwd']";

String S2 = "The email or password you entered is incorrect.";

System.*out*.println(" test is a " + fCompare(S1,S2));

driver.quit();

}

**public** **void** fLogin() **throws** Exception {

**int** iwait;

iwait = 3000;

driver.navigate().to("youtube.com");

driver.findElement(By.*xpath*("(//button[@type='button'])[2]")).click();

Thread.*sleep*(iwait);

driver.findElement(By.*xpath*("//input[@id ='Email']")).sendKeys("jeagkfkja");

driver.findElement(By.*xpath*("//input[@name ='Passwd']")).sendKeys("lkdshsfjl");

Thread.*sleep*(iwait);

driver.findElement(By.*xpath*("//input[@id ='signIn']")).click();

}

**public** String fCompare(String C1, String C2){

String vResult = driver.findElement(By.*xpath*(C1)).getText();

**if** (vResult.equalsIgnoreCase(C2)){

**return** "Pass";

}**else** {

**return** "Fail";

}

}

}

**Day11**

**• Building library with specific keywords.**

• Get attribute gets something from the page and gives you the value like get text. Can be used for check and uncheck boxes.

Keywords and executable functions

**import** org.junit.Test;

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

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.ui.Select;

**public** **class** webKeywords {

@Test

**public** **void** KWtest() **throws** Exception{

**int** iwait;

iwait = 3000;

FirefoxDriver myD = **new** FirefoxDriver();

navigateBrowser(myD, "https://qa.silverbackmdm.com" + "/admin");

enterText(myD, "//input[@id='Username']", " priyap");

enterText(myD, "//input[@id='Password']", "S1lverb@ck");

Thread.*sleep*(iwait);

clickElement(myD, "//input[@type='submit']");

Thread.*sleep*(iwait);

clickElement(myD, "/html/body/div/div/ul/li[4]/a");

Thread.*sleep*(iwait);;

clickElement(myD, "/html/body/div[2]/div[2]/div/ul/li/a");

Thread.*sleep*(iwait);

clickElement(myD, "//input[@type='button']");

Thread.*sleep*(iwait);

selectList(myD, "//Select[@id = 'ApplicationType']", "App Store");

Thread.*sleep*(iwait);

checkBox(myD, "//input[@id='IsDeviceAppPortalVisible']");

}

**public** **void** enterText(FirefoxDriver driver, String xPath, String Data) {

// Purpose: Enter text into a edit field

//i/p: driver, xPath, Data

//o/p: Nil

//CreatedBy : Priya

//When :8/05/14

//EditedBy : Priya

//When : 8/5/14

driver.findElement(By.*xpath*(xPath)).clear();//some applications already have a default values in edit field, so this line is to clear the default path

driver.findElement(By.*xpath*(xPath)).sendKeys(Data);

}

**public** String readText(FirefoxDriver driver, String xPath) { //B'coz im going to get an o/p, I changed void to String to return the data

// Purpose: read text from any field.

//i/p: driver, xPath

//o/p: Data

//CreatedBy : Priya

//When :8/05/14

//EditedBy : Priya

//When : 8/5/14

**return** driver.findElement(By.*xpath*(xPath)).getText(); //returns whatever comes here

}

**public** **void** clickElement(FirefoxDriver driver, String xPath) { //B'coz im going to get an o/p, I changed void to String to return the data

// Purpose: Click any element on the AUT

//i/p: driver, xPath

//o/p: NIl

//CreatedBy : Priya

//When :8/05/14

//EditedBy : Priya

//When : 8/5/14

driver.findElement(By.*xpath*(xPath)).click(); // sometimes click may not work, so I write a code to send enter

//driver.findElement(By.linkText(linkText)).click();

//driver.findElement(By.xpath(xPath)).sendKeys("\n");

}

**public** **void** clickElementbylink(FirefoxDriver driver, String link) { //B'coz im going to get an o/p, I changed void to String to return the data

// Purpose: Click any element on the AUT

//i/p: driver, xPath

//o/p: NIl

//CreatedBy : Priya

//When :8/05/14

//EditedBy : Priya

//When : 8/5/14

driver.findElement(By.*linkText*(link)).click();

driver.findElement(By.*linkText*(link)).sendKeys("\n");

}

**public** **void** selectList(FirefoxDriver driver, String xPath, String Data) {

// Purpose: Select from a drop down list

//i/p: driver, xPath, Data

//o/p: NIl

//CreatedBy : Priya

//When :8/05/14

//EditedBy : Priya

//When : 8/5/14

Select myList = **new** Select(driver.findElement(By.*xpath*(xPath)));//creating memory in myList

myList.selectByVisibleText(Data);

myList = **null**; //clears the memory in my list

}

**public** **void** checkBox(FirefoxDriver driver, String xPath) { //this has to be checked.

// Purpose: checks a checkbox

//i/p: driver, xPath

//o/p: NIl

//CreatedBy : Priya

//When :8/05/14

//EditedBy : Priya

//When : 8/5/14

//Is it checked?

**if**(driver.findElement(By.*xpath*(xPath)).getAttribute("checked").equalsIgnoreCase("checked")){

//then do nothing

} **else** {

//click

driver.findElement(By.*xpath*(xPath)).click();

}

}

//uncheckBox

// selectRadio

**public** **void** navigateBrowser(FirefoxDriver driver, String Data){

// Purpose: Navigates a browser to a URL

//i/p: driver, xPath

//o/p: NIl

//CreatedBy : Priya

//When :8/05/14

//EditedBy : Priya

//When : 8/5/14

driver.navigate().to(Data);

}

}

• How do you compare xpaths for headings?

• In any application we are identifying the elements by using xptah.

• Xpath has a html tag, attribute name and value – a combination of these

• If there is more than 1 element with exactly the same attributes, then what does Selenium does? It identifies the first object with that xpath.

• Example the following are the 2 xpaths with exactly same attributes

//input[@id =’username’]

**DYNAMIC APPLICATIONS**

• Then you write it as **//descendant:: input[@id =’username’][1**] – this finds the first element that matches this xpath

**• //descendant::div[1]** – it shows the first element of that type

• XPath for headers

**//descendant::h2[1]** – first heading of size H2

**//descendant::h2[2**] – Second heading of size H2

• This is very similar to index in qtp

**Checkbox Keyword dint work:**

**Day12**

**26 mins – read excel**

**import** org.junit.Test;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** kdf2 {

@Test

**public** **void** KDF(){

String[][] xTC, xTS;

String vKW, vxPtah, vTestData;

FirefoxDriver myd = **new** FirefoxDriver();

//Read the entire TC or TS sheet

xTC = readXL("C:\\Priya\\Silverback 4.3\\Automation\QService.xlsx", "TestCases");

xTS = readXL("C:\Priya\Silverback 4.3\Automation\QService.xlsx", "TestSteps");

//Go to each row in TC and identify which TC to execute

**for**(**int** i=1; i<xTC.length; i++) { //xtc.lenght will give the number of arrays in a 2 dimensional array

System.out.println("TCID is" +xTC[i][0]);

System.out.println("TCID is" + xTC[1][3]);

**if** (xTC[i][3].equalsIgnoreCase("Y")){ //identify which test case to execute

System.out.println("Yes this needs to be executed");

//go to corresponding TS in each TS sheet and identify corresponding KW

**for** (**int** j=1; j<xTS.length; j++){

**if**(xTC[i][0].equalsIgnoreCase(xTS[j][1]))

System.out.println("Keyword is" + xTS[j][4]);

System.out.println("xPath is" + xtS[j][5]);

System.out.println("Data is " + xTS[j][6])

//Call the corresponding keyword function

vKW = xTS[j][4];

vxPath = xts[j][5];

vTestData = xTS[j][6];

**if** (vKW.equalsIgnoreCase("enterText")) {

enterText(myD, vxPath, vtestData)

}

**if** (vKW.equalsIgnoreCase("navigateBrowser")) {

enterText(myD, vtestData)

}

**if** (vKW.equalsIgnoreCase("clickElement")) {

enterText(myD, vxPath, vtestData)

}

**if** (vKW.equalsIgnoreCase("readText")) {

System.out.println( "Test step is a " + readText(myD, vxPath));

}

**if** (vKW.equalsIgnoreCase("verifyText")) {

System.out.println( "Test step is a " + verifytext(myD, vxPath, vtestData));

}

}

}

}

}

}

// Capture o/p results

}

}

**Record – Day4 – ANT installation:**

**ANT Installation**

Steps to Install ANT

Step 1: Goto http://archive.apache.org/dist/ant/binaries/ link.

Step 2: Click on apache-ant-1.8.2-bin.zip. This allows you to download on your local machine.

Step 3: Create a new folder ANT in C:\Selenium.

Step 4: Unzip the file and store the unzipped folder “apache-ant-1.8.2” in new folder. Example,

“C:\Selenium\ apache-ant-1.8.2”.

Step 5: Go to Start > My Computer > Properties. Click Advanced Settings.

Step 6: Click on Environment Variables. Step 7: Add a New System Variable “AntTest”. Click on New in System Variables and add values as shown below: Step 8: Click OK

Step 9: Now goto Path variable in System Variables. Click on Edit as shown below: Step 10: DO NOT CHANGE ANY VALUE THAT IS ALREADY THERE. At the end of Variable Value, put a semi-colon(;) and

paste the complete path of ANT\bin folder again here like below

Step 11: Click on OK and you are done with ANT installation.

Step 12: Now Confirm ANT is installed. Go to command prompt.

Step 13: On the command prompt type ‘ant’ from any directory that you are present in.

Step 14: Now go to “C:\Program Files\Java\jdk1.6.0\_27\lib” and copy tools.jar file.

Step 15: Now go to “C:\Program Files\Java\jre6\lib” and paste tools.jar file here.

Now run cmd, it says build.xml does not exist!. Build failed.

Follow the steps below:

• Ant needs this xml file to interact during run to be able to report errors

• Now we need to create a file called build.xml in your project

• How to create a build.xml file for our project

Build.xmll file.

• From the cmd prompt, Go to the location where the build file is usually in the default package name you gave or the project folder.

• Type dir and press enter

• Ant clean

• Ant compile

An Ant\_build will be created in the project folder structure

• Ant build

• Ant run

•

**To call from another function example calling from mymath to math**

**Syntax: classname.methodname(f1, f2)**

Type mymath.fadd inwhcihever test

**Day12 KW with read excel**

**import** java.io.File;

**import** java.io.FileInputStream;

**import** org.apache.poi.hssf.usermodel.HSSFCell;

**import** org.apache.poi.hssf.usermodel.HSSFRow;

**import** org.apache.poi.hssf.usermodel.HSSFSheet;

**import** org.apache.poi.hssf.usermodel.HSSFWorkbook;

**import** org.junit.Test;

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

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** kdf4 {

**public** **int** xRow; **int** xCol;

@Test

**public** **void** qS1() **throws** Exception{

String[][] xTC, xTS;

//WebDriver myD = new FirefoxDriver();

String xPath = "C:/Priya/Silverback 4.3/Automation/QService.xls";

//Read the entire TC and TS

xTC = xlRead(xPath,"TestCases");

xTS = xlRead(xPath, "TestSteps");

System.*out*.println("Test Step sheet is" + xTS);

//Go to each row in TC and identify which TC to execute

**for** (**int** i = 1; i <xTC.length; i++) {

System.*out*.println("TCID is " + xTC[i][0]);

**if** (xTC[i][3].equalsIgnoreCase("Y")) {

System.*out*.println("Yes, execute this");

}

//go to every test step if it is y

**for** (**int** j=1; j<xTS.length; j++){

System.*out*.println("TSID is" +xTS[i][0]);

**if** (xTC[i][0].equalsIgnoreCase(xTS[j][1])){

System.*out*.println("KW to execute is " + xTS[j][4]);

System.*out*.println("xpath to execute is " + xTS[j][4]);

System.*out*.println("Data to execute is " + xTS[j][4]);

}

}

}

//Go to corresponding TS in each sheet and and function with req input variable

// Capture o/p results

}

//Read from Excel

**public** String[][] xlRead(String sPath, String iSheet) **throws** Exception{ // This line xlRead is the call to this function. We are passing the xl path in this

//String[][] myExcel = null;

//int xRow, xCol; commenting as I will declare it main code.

String[][] xData;

//file, wb,sheets

File myxl = **new** File(sPath); // File is telling Java that it will use I/p o/p streams to interact with Java. The name of the file is myxl here.

FileInputStream myStream = **new** FileInputStream(myxl); // FIle input stream reads i/p and o/p into the file

HSSFWorkbook myWB = **new** HSSFWorkbook(myStream); // Creates workbook within this file.

HSSFSheet mySheet = myWB.getSheet(iSheet); // getSheetAt(0) method - Referring to 1st sheet, 1 -2nd sheet and so on

//Get row count

xRow = mySheet.getLastRowNum()+1; //gets number of rows in my sheet. SInce it starts with 0, adding +1 to it will get total number of rows which is the last row.

xCol = mySheet.getRow(0).getLastCellNum(); // to get cells or number of columns, first go to row 0, then get the last cell number of that row.

xData = **new** String[xRow][xCol]; // we are creating an Array of type string. This puts the number of rows and cols.

System.*out*.println("Rows are " + xRow);

System.*out*.println("Cols are " + xCol);

// Go to each cell

**for** (**int** i = 0; i < xRow; i++) {

HSSFRow row = mySheet.getRow(i); // select and point to a specific row in my sheet it points according to the iteration

**for** (**int** j = 0; j < xCol; j++) {

HSSFCell cell = row.getCell(j); // on that row, get the value of each column of each row.

String value = *cellToString*(cell); //convert a specific cell type(int,string/double etc) to a simple string - this I can get from the function HSSFcell

xData[i][j] = value;

}

}

**return** xData;

}

// Method to write into an XL

**public** **static** **void** writeXL(String sPath, String iSheet, String[][] xData) **throws** Exception{

File outFile = **new** File(sPath);

HSSFWorkbook wb = **new** HSSFWorkbook();

HSSFSheet osheet = wb.createSheet(iSheet);

**int** xR\_TS = xData.length;

**int** xC\_TS = xData[0].length;

**for** (**int** myrow = 0; myrow < xR\_TS; myrow++) {

HSSFRow row = osheet.createRow(myrow);

**for** (**int** mycol = 0; mycol < xC\_TS; mycol++) {

HSSFCell cell = row.createCell(mycol);

cell.setCellType(HSSFCell.*CELL\_TYPE\_STRING*);

cell.setCellValue(xData[myrow][mycol]);

}

FileOutputStream fOut = **new** FileOutputStream(outFile);

wb.write(fOut);

fOut.flush();

fOut.close();

}

}

// A new function to convert an object of type excel cell to a string value.

**public** **static** String cellToString(HSSFCell cell) {

// This function will convert an object of type excel cell to a string value

**int** type = cell.getCellType(); //cell.getcell type returns an integer which

Object result;

**switch** (type) {

**case** HSSFCell.*CELL\_TYPE\_NUMERIC*: //0

result = cell.getNumericCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_STRING*: //1

result = cell.getStringCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_FORMULA*: //2

**throw** **new** RuntimeException("We can't evaluate formulas in Java");

**case** HSSFCell.*CELL\_TYPE\_BLANK*: //3

result = "-";

**break**;

**case** HSSFCell.*CELL\_TYPE\_BOOLEAN*: //4

result = cell.getBooleanCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_ERROR*: //5

**throw** **new** RuntimeException ("This cell has an error");

**default**:

**throw** **new** RuntimeException("We don't support this cell type: " + type);

}

**return** result.toString();

}

}

Now call the corresponding KW

Read method which was reading as a decimal was fixd by this coded

**import** java.io.File;

**import** java.io.FileInputStream;

**import** org.apache.poi.hssf.usermodel.HSSFCell;

**import** org.apache.poi.hssf.usermodel.HSSFRow;

**import** org.apache.poi.hssf.usermodel.HSSFSheet;

**import** org.apache.poi.hssf.usermodel.HSSFWorkbook;

**import** org.junit.Test;

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

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

**import** org.openqa.selenium.firefox.FirefoxDriver;

**import** org.openqa.selenium.support.ui.Select;

**public** **class** dummy1 {

**public** **int** xRow; **int** xCol;

**int** iWait;

@Test

**public** **void** qS1() **throws** Exception{

iWait = 3000;

//Thread.sleep(iWait);

String[][] xTC, xTS;

String vKW, vXpath, vTestData;

FirefoxDriver myD = **new** FirefoxDriver();

String xPath = "C:/Priya/Silverback 4.3/Automation/QService.xls";

//Read the entire TC and TS

xTC = xlRead(xPath,"TestCases");

xTS = xlRead(xPath, "TestSteps");

System.*out*.println("Test Step sheet is" + xTS);

//Go to each row in TC and identify which TC to execute

**for** (**int** i = 1; i<xTC.length; i++) { //Go to each row in TC

System.*out*.println("TCID is " + xTC[i][0]);

//Thread.sleep(iWait);

**if** (xTC[i][3].equalsIgnoreCase("Y")) {

Thread.*sleep*(iWait);

System.*out*.println("Yes, execute this");

Thread.*sleep*(iWait);

//go to every test step if it is y

**for** (**int** j=1; j<xTS.length; j++){

//Thread.sleep(iWait);

System.*out*.println("TSID is" +xTS[i][0]);

Thread.*sleep*(iWait);

**if** (xTC[i][0].equalsIgnoreCase(xTS[j][1])){

//Thread.sleep(iWait);

System.*out*.println("KW to execute is " + xTS[j][4]);

Thread.*sleep*(iWait);

System.*out*.println("xpath to execute is " + xTS[j][5]);

System.*out*.println("Data to execute is " + xTS[j][6]);

// call the corresponding KW function

vKW = xTS[j][4] ;

vXpath = xTS[j][5];

vTestData = xTS[j][6];

**if** (vKW.equalsIgnoreCase("enterText")) {

enterText(myD, vXpath, vTestData);

}

**if** (vKW.equalsIgnoreCase("navigateBrowser")){

navigateBrowser(myD, vTestData);

}

**if**(vKW.equalsIgnoreCase("clickElement")){

clickElement(myD, vXpath);

}

**if** (vKW.equalsIgnoreCase("verifyText")){

System.*out*.println("Test is a" + verifyText(myD, vXpath, vTestData));

}

}

}

}

}

}

// Capture o/p results

//Read from Excel

**public** String[][] xlRead(String sPath, String iSheet) **throws** Exception{ // This line xlRead is the call to this function. We are passing the xl path in this

//String[][] myExcel = null;

//int xRow, xCol; commenting as I will declare it main code.

String[][] xData;

//file, wb,sheets

File myxl = **new** File(sPath); // File is telling Java that it will use I/p o/p streams to interact with Java. The name of the file is myxl here.

FileInputStream myStream = **new** FileInputStream(myxl); // FIle input stream reads i/p and o/p into the file

HSSFWorkbook myWB = **new** HSSFWorkbook(myStream); // Creates workbook within this file.

HSSFSheet mySheet = myWB.getSheet(iSheet); // getSheetAt(0) method - Referring to 1st sheet, 1 -2nd sheet and so on

//Get row count

xRow = mySheet.getLastRowNum()+1; //gets number of rows in my sheet. SInce it starts with 0, adding +1 to it will get total number of rows which is the last row.

xCol = mySheet.getRow(0).getLastCellNum(); // to get cells or number of columns, first go to row 0, then get the last cell number of that row.

xData = **new** String[xRow][xCol]; // we are creating an Array of type string. This puts the number of rows and cols.

System.*out*.println("Rows are " + xRow);

System.*out*.println("Cols are " + xCol);

// Go to each cell

**for** (**int** i = 0; i < xRow; i++) {

HSSFRow row = mySheet.getRow(i); // select and point to a specific row in my sheet it points according to the iteration

**for** (**int** j = 0; j < xCol; j++) {

HSSFCell cell = row.getCell(j); // on that row, get the value of each column of each row.

String value = *cellToString*(cell); //convert a specific cell type(int,string/double etc) to a simple string - this I can get from the function HSSFcell

xData[i][j] = value;

}

}

**return** xData;

}

// A new function to convert an object of type excel cell to a string value.

**public** **static** String cellToString(HSSFCell cell) {

// This function will convert an object of type excel cell to a string value

**int** type = cell.getCellType(); //cell.getcell type returns an integer which

Object result;

**switch** (type) {

**case** HSSFCell.*CELL\_TYPE\_NUMERIC*: //0

String stringValue=""+cell.getNumericCellValue();

String[] splitValue=stringValue.split(".0");

//System.out.println(splitValue[0]);

result = splitValue[0];

**break**;

**case** HSSFCell.*CELL\_TYPE\_STRING*: //1

result = cell.getStringCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_FORMULA*: //2

**throw** **new** RuntimeException("We can't evaluate formulas in Java");

**case** HSSFCell.*CELL\_TYPE\_BLANK*: //3

result = "-";

**break**;

**case** HSSFCell.*CELL\_TYPE\_BOOLEAN*: //4

result = cell.getBooleanCellValue();

**break**;

**case** HSSFCell.*CELL\_TYPE\_ERROR*: //5

**throw** **new** RuntimeException ("This cell has an error");

**default**:

**throw** **new** RuntimeException("We don't support this cell type: " + type);

}

**return** result.toString();

}

**Day13**

Calling methods from a different class such as Read/Write XL:

For example, create a class with the name XL and put Read/Write in a class. To call this method into a different class, in that class, put the name of this read/write excel in front of that function you are calling.

Example, Im calling xread to the following line

xTC = XL.xlRead(xPath,"TestCases");

xTS = XL.xlRead(xPath, "TestSteps");

However you have to declare the XL read write method to static. The original was declared as follows.

**public** **class** XL {

**public** String[][] xlRead(String sPath, String iSheet) **throws** Exception{ // This line xlRead is the call to this function. We are passing the xl path in this

//String[][] myExcel = null;

**int** xRow, xCol; //declare it main code.

String[][] xData;

When changed to Static, it got changed to the following

**public** **class** XL {

**public** **static** String[][] xlRead(String sPath, String iSheet) **throws** Exception{ // This line xlRead is the call to this function. We are passing the xl path in this

//String[][] myExcel = null;

**int** xRow, xCol; //declare it main code.

String[][] xData;

**Static**: We are not going to personalise this method any further. If I need to execute a method of a class and not of a object from that class is STATIC.

The instance of a class is an object. Object has properties and methods. Some properties and methods remain static. That means if I create Object1, the methods and properties specific to this are the same. If I create Object2, methods and properties may be different in terms of their values. Different objects keep changing in terms of what their values are.

So every time I call a static method, put the name of that class before the function or method I am calling.

**Errors**:

Excel - Automation – Testing – Application

**Scenario1**: you don’t know where the error can occur either in the code or excel or application

There are multiple scenarios that may fail and we do not know where it can fail.

**Scenario2**: let’s say after login, we expect a result, or on YouTube the expected results. So we know where the error can occur. inputs such as excel is given correct, we put verifications.

How do you know if the issue is with the application or the java code or the excel.

Rule out the issue. Order to check where is

**Excel**: if my excel is correct

**Code**: if my code is correct

**Application**: then at the end I can point to my application

Errors can occur anywhere. For example if my excel went wrong. What should I do? I should still **continue** the test so other tests can continue.

Note where it happened and what happened. That is what we are going to see now.

How do I know that an issue has happened and how do I let it come? To do that, there is a simple thing called try.

Syntax:

**try**{

} **catch**{

}

For example I gave a wrong xpath as input in the excel, and the test on the java gave errors and stopped. I want the code to capture the error and continue testing.

Wherever that issue may occur, put this try code. Example KW is very common as that’s where most of it goes onto application.

**try**{

vKW = xTS[j][4] ;

vXpath = xTS[j][5];

vTestData = xTS[j][6];

**if** (vKW.equalsIgnoreCase("enterText")) {

KW.enterText(myD, vXpath, vTestData);

}

**if** (vKW.equalsIgnoreCase("navigateBrowser")){

KW.navigateBrowser(myD, vTestData);

}

**if**(vKW.equalsIgnoreCase("clickElement")){

KW.clickElement(myD, vXpath);

}

**if** (vKW.equalsIgnoreCase("verifyText")){

System.out.println("Test is a" + KW.verifyText(myD, vXpath, vTestData));

}

} **catch** {

}

**try**{

vKW = xTS[j][4] ;

vXpath = xTS[j][5];

vTestData = xTS[j][6];

**if** (vKW.equalsIgnoreCase("enterText")) {

KW.*enterText*(myD, vXpath, vTestData);

}

**if** (vKW.equalsIgnoreCase("navigateBrowser")){

KW.*navigateBrowser*(myD, vTestData);

}

**if**(vKW.equalsIgnoreCase("clickElement")){

KW.*clickElement*(myD, vXpath);

}

**if** (vKW.equalsIgnoreCase("verifyText")){

System.*out*.println("Test is a" + KW.*verifyText*(myD, vXpath, vTestData));

}

} **catch** (Exception e){ //If an error occurred then the code will go to the catch exception.

System.*out*.println("Error happened" + e);

}

Now when I execute this test, it will try the code within the try curls and loops back. If an error occurred then the code will go to the catch exception.

I can also put where the error occurred.

All exceptions can be checked in the console.

Where are the tests stored in my Java code? From excel to Java – Java to App and App to Excel.

In the 2 dimensional arrays I declared.

**Day14**

vKW = xTS[j][4] ;

vXpath = xTS[j][5];

vTestData = xTS[j][6];

// call the corresponding KW function

**try**{

**if** (vKW.equalsIgnoreCase("enterText")) {

KW.*enterText*(myD, vXpath, vTestData);

}

**if** (vKW.equalsIgnoreCase("navigateBrowser")){

KW.*navigateBrowser*(myD, vTestData);

}

**if**(vKW.equalsIgnoreCase("clickElement")){

KW.*clickElement*(myD, vXpath);

}

**if** (vKW.equalsIgnoreCase("verifyText")){

System.*out*.println("Test is a" + KW.*verifyText*(myD, vXpath, vTestData));

}

System.*out*.println("Pass" + xTC[i][0] + "\_" + vKW);

} **catch** (Exception e){ //If an error occurred then the code will go to the catch exception. Anything to do with Application will get captured here.

System.*out*.println("Fail" + xTC[i][0] + "\_" + vKW);

System.*out*.println("Error happened" + e);

}

Wait statement for page to wait

}

} **catch** (Exception e){ //If an error occurred then the code will go to the catch exception.

The error gets captured in the variable e.

**Get the results back to Excel.**

// Method to write into an XL

**public** **static** **void** writeXL(String sPath, String iSheet, String[][] xData) **throws** Exception{

File outFile = **new** File(sPath);

HSSFWorkbook wb = **new** HSSFWorkbook();

HSSFSheet osheet = wb.createSheet(iSheet);

**int** xR\_TS = xData.length;

**int** xC\_TS = xData[0].length;

**for** (**int** myrow = 0; myrow < xR\_TS; myrow++) {

HSSFRow row = osheet.createRow(myrow);

**for** (**int** mycol = 0; mycol < xC\_TS; mycol++) {

HSSFCell cell = row.createCell(mycol);

cell.setCellType(HSSFCell.*CELL\_TYPE\_STRING*);

cell.setCellValue(xData[myrow][mycol]);

}

FileOutputStream fOut = **new** FileOutputStream(outFile);

wb.write(fOut);

fOut.flush();

fOut.close();

}

}

//Write back the test steps to Excel

XL1.*writeXL*("C:\\Priya\\Silverback 4.3\\KDFResults.xls", "TestSteps Results", xTS);

}

}

**ERROR HANDLING CODE**

**import** org.junit.Test;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** Errorhandling2 {

**int** iWait;

@Test

**public** **void** qS1() **throws** Exception{

iWait = 3000;

//Thread.sleep(iWait);

String[][] xTC, xTS;

String vKW, vXpath, vTestData;

FirefoxDriver myD = **new** FirefoxDriver();

String xPath = "C:/Priya/Silverback 4.4/Automation Test Cases/QService.xls";

//Read the entire TC and TS

xTC = xlreadwrite.*xlRead*(xPath,"TestCases");

xTS = xlreadwrite.*xlRead*(xPath, "TestSteps");

System.*out*.println("Test Step sheet is" + xTS);

//Go to each row in TC and identify which TC to execute

**for** (**int** i = 1; i<xTC.length; i++) { //Go to each row in TC

System.*out*.println("TCID is " + xTC[i][0]);

//Thread.sleep(iWait);

**if** (xTC[i][3].equalsIgnoreCase("Y")) {

Thread.*sleep*(iWait);

System.*out*.println("Yes, execute this");

Thread.*sleep*(iWait);

//go to every test step if it is y

**for** (**int** j=1; j<xTS.length; j++){

//Thread.sleep(iWait);

System.*out*.println("TSID is" +xTS[i][0]);

Thread.*sleep*(iWait);

**if** (xTC[i][0].equalsIgnoreCase(xTS[j][1])){

//Thread.sleep(iWait);

System.*out*.println("KW to execute is " + xTS[j][4]);

Thread.*sleep*(iWait);

System.*out*.println("xpath to execute is " + xTS[j][5]);

System.*out*.println("Data to execute is " + xTS[j][6]);

// call the corresponding KW function

vKW = xTS[j][4] ;

vXpath = xTS[j][5];

vTestData = xTS[j][6];

String vError = "Pass"; // This means Before the try is executed it is pass and if it correctly executed all through try, vError will be pass

**try**{

**if** (vKW.equalsIgnoreCase("enterText")) {

KW.*enterText*(myD, vXpath, vTestData);

}

**if** (vKW.equalsIgnoreCase("navigateBrowser")){

KW.*navigateBrowser*(myD, vTestData);

}

**if**(vKW.equalsIgnoreCase("clickElement")){

KW.*clickElement*(myD, vXpath);

}

**if** (vKW.equalsIgnoreCase("verifyText")){

vError = KW.*verifyText*(myD, vXpath, vTestData);

System.*out*.println("Test is a" + vError );

}

**if** (vKW.equalsIgnoreCase("readText")){

vError = KW.*readText*(myD, vXpath);

System.*out*.println("Test is a" + vError);

}

**if**(vKW.equalsIgnoreCase("waitTime")){

KW.*waitTime*(3000);

}

**if** (vError.equalsIgnoreCase("Pass")) {

System.*out*.println("Pass:" + xTC[i][0] + "\_" + vKW); //If it's come this far then pass

xTS[j][8] ="Pass";

} **else** {

System.*out*.println("Fail:" + xTC[i][0] + "\_" + vKW); //If it's come this far then pass

xTS[j][8] ="Error happened as a function returned fail";

}

} **catch**(Exception myError) {

System.*out*.println("Fail:" + xTC[i][0] + "\_" + vKW); //Fail if the try loop fails.

System.*out*.println("Error Happened" + myError); //prints the errors into the logs.

xTS[j][9] ="Error Happened" + myError;

}

}

}

}

}

myD.quit();

//Write back the test Steps with the error

xlreadwrite.*writeXL*("C:/Priya/Silverback 4.4/Results/TestResults.xls", "TestSteps Results", xTS );

}

}

//Write back the test steps to Excel

XL1.writeXL("C:\\Priya\\Silverback 4.3\\KDFResults.xls", "TestSteps Results", xTS);

}

}

**HYBRID FRAMEWORK**

**Multiple sets of test data into one framework**

**Data Driven framework:** Data is hard coded

**Keyword driver framework:** 1 set of Test data and multiple Test Cases

**Hybrid Framework** – Different sets of data and multiple Test Cases

**import** org.junit.Test;

**import** org.openqa.selenium.firefox.FirefoxDriver;

**public** **class** HF2 {

**int** iWait;

@Test

**public** **void** qS1() **throws** Exception{

iWait = 3000;

//Thread.sleep(iWait);

String[][] xTD, xTC, xTS; //I need a 2 dimensional test Data to store/read the test data

String vKW, vXpath, vTestData;

FirefoxDriver myD = **new** FirefoxDriver();

String dTDID, dURL, dilogin, dipaswd, dqinput, dExecute, dExpectedResults, dActualResults;

String xPath = "C:/Priya/Silverback 4.4/Automation Test Cases/QService.xls";

//Read the entire TC and TS

xTC = xlreadwrite.*xlRead*(xPath,"TestCases");

xTS = xlreadwrite.*xlRead*(xPath, "TestSteps");

xTD = xlreadwrite.*xlRead*(xPath, "TestData");

//int vRows = xTC.length;

//int vCols = xTC[0].length;

System.*out*.println("Test Step sheet is" + xTS);

//Go to each row in TC and identify which TC to execute

**for** (**int** h=1; h<xTD.length; h++){

**if** (xTD[h][5].equalsIgnoreCase("Y")){ //is the test Data execution for yes

//Then read the values of the following local variables data from each row

dTDID = xTD[h][1];

dURL = xTD[h][2];

dilogin = xTD[h][3];

dipaswd = xTD[h][4];

dqinput = xTD[h][5];

**for** (**int** i = 1; i<xTC.length; i++) { //Go to each row in TC

System.*out*.println("TCID is " + xTC[i][0]);

//Thread.sleep(iWait);

**if** (xTC[i][3].equalsIgnoreCase("Y")) {

Thread.*sleep*(iWait);

System.*out*.println("Yes, execute this");

Thread.*sleep*(iWait);

//go to every test step if it is y

**for** (**int** j=1; j<xTS.length; j++){

//Thread.sleep(iWait);

System.*out*.println("TSID is" +xTS[i][0]);

Thread.*sleep*(iWait);

**if** (xTC[i][0].equalsIgnoreCase(xTS[j][1])){

//Thread.sleep(iWait);

System.*out*.println("KW to execute is " + xTS[j][4]);

Thread.*sleep*(iWait);

System.*out*.println("xpath to execute is " + xTS[j][5]);

System.*out*.println("Data to execute is " + xTS[j][6]);

// call the corresponding KW function

vKW = xTS[j][4] ;

vXpath = xTS[j][5];

vTestData = xTS[j][6];

String vError = "Pass"; // This means Before the try is executed it is pass and if it correctly executed all through try, vError will be pass

//Get the corresponding Test Data value based on the name if the TD variable in TS

getTDValue (vTestData); // Function to load the TD values and load into variables

**try**{

**if** (vKW.equalsIgnoreCase("enterText")) {

KW.*enterText*(myD, vXpath, vTestData);

}

**if** (vKW.equalsIgnoreCase("navigateBrowser")){

KW.*navigateBrowser*(myD, vTestData);

}

**if**(vKW.equalsIgnoreCase("clickElement")){

KW.*clickElement*(myD, vXpath);

}

**if** (vKW.equalsIgnoreCase("verifyText")){

vError = KW.*verifyText*(myD, vXpath, vTestData);

System.*out*.println("Test is a" + vError );

}

**if** (vKW.equalsIgnoreCase("readText")){

vError = KW.*readText*(myD, vXpath);

System.*out*.println("Test is a" + vError);

}

**if** (vKW.equalsIgnoreCase("clickElementbylink")){

KW.*clickElementbylink*(myD, vTestData);

}

**if** (vKW.equalsIgnoreCase("selectList")){

KW.*selectList*(myD, vXpath, vTestData);

}

**if** (vKW.equalsIgnoreCase("checkBox")){

KW.*checkBox*(myD, vXpath);

}

**if**(vKW.equalsIgnoreCase("waitTime")){

KW.*waitTime*(3000);

}

**if** (vError.equalsIgnoreCase("Pass")) {

System.*out*.println("Pass:" + xTC[i][0] + "\_" + vKW); //If it's come this far then pass

xTS[j][8] ="Pass";

} **else** {

System.*out*.println("Fail:" + xTC[i][0] + "\_" + vKW); //If it's come this far then pass

xTS[j][8] ="Error happened as a function returned fail";

}

} **catch**(Exception myError) {

System.*out*.println("Fail:" + xTC[i][0] + "\_" + vKW); //Fail if the try loop fails.

System.*out*.println("Error Happened" + myError); //prints the errors into the logs.

xTS[j][9] ="Error Happened" + myError;

}

}

}

}

}

myD.quit();

//Write back the test Steps with the error

xlreadwrite.*writeXL*("C:/Priya/Silverback 4.4/Results/TestResults.xls", "TestSteps Results", xTS );

}

}

}

}

//try{

// }

// catch(Exception e){

// System.out.println("Error Happened" +e);

// }

//Some times the error may occur if we are not waiting for the page to load. So use wait statements.

WebDriver driver = new FirefoxDriver();

WebDriverWait wdw = new WebDriverWait(driver, 30);

driver.get(pageURL);

wdw.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector("#mybutton")));

driver.findElement(By.cssSelector("#mybutton")).click();

Day15

Hybrid Framework

import org.junit.Test;

import org.openqa.selenium.firefox.FirefoxDriver;

import java.io.File;

import java.io.FileInputStream;

import org.apache.poi.hssf.usermodel.HSSFCell;

import org.apache.poi.hssf.usermodel.HSSFRow;

import org.apache.poi.hssf.usermodel.HSSFSheet;

import org.apache.poi.hssf.usermodel.HSSFWorkbook;

import org.junit.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.support.ui.Select;

public class HybridFW {

//public int xRow; int xCol;

public int iWait;

public String dURL, dilogin, dipswd, dqinput, dExpText;

@Test

public void qS1() throws Exception{

String[][] xTC, xTS, xTD;

String vKW, vXpath, vTestData;

FirefoxDriver myD = new FirefoxDriver();

dURL = "https://qa.silverbackmdm.com/admin";

dilogin = "Priyap";

dipswd ="S1lverb@ck";

String xPath = "C:/Priya/Silverback 4.3/Automation/QServicehf.xls";

//Read the entire TC and TS

xTC = XL1.xlRead(xPath,"TestCases");

xTS = XL1.xlRead(xPath, "TestSteps");

xTD = XL1.xlRead(xPath, "TestData");

System.out.println("Test Step sheet is" + xTS);

//Go to each row in TC and identify which TC to execute

for (int k = 1; k<xTD.length; k++){ //Go to each row of test data and read the values of each variable

if (xTD[k][6].equalsIgnoreCase("Y")){ //Is the TD for executiin

//Read the values of the variables from test Data SHeet

for (int i = 1; i<xTC.length; i++) { //Go to each row in TC

System.out.println("TCID is " + xTC[i][0]);

if (xTC[i][3].equalsIgnoreCase("Y")) {

xTC[i][4] = "Pass"; //default TC result is a pass

System.out.println("Yes, execute this");

//go to every test step if it is y

for (int j=1; j<xTS.length; j++){

Thread.sleep(iWait);

System.out.println("TSID is" +xTS[i][0]);

Thread.sleep(iWait);

if (xTC[i][0].equalsIgnoreCase(xTS[j][1])){

//Thread.sleep(iWait);

System.out.println("KW to execute is " + xTS[j][4]);

Thread.sleep(iWait);

System.out.println("xpath to execute is " + xTS[j][5]);

System.out.println("Data to execute is " + xTS[j][6]);

vKW = xTS[j][4] ;

vXpath = xTS[j][5];

vTestData = xTS[j][6];

String vError = "Pass";

// call the corresponding KW function

//Get the corresponding Test Data value based on the name of the TD variable in TS

vTestData = getTDValue(vTestData); //

try{

if (vKW.equalsIgnoreCase("enterText")) {

KW.enterText(myD, vXpath, vTestData);

}

if (vKW.equalsIgnoreCase("navigateBrowser")){

KW.navigateBrowser(myD, vTestData);

}

if(vKW.equalsIgnoreCase("clickElement")){

KW.clickElement(myD, vXpath);

}

if (vKW.equalsIgnoreCase("verifyText")){

vError = KW.verifyText(myD, vXpath, vTestData);

System.out.println("Test is a" + vError );

}

if (vKW.equalsIgnoreCase("readText")){

vError = KW.readText(myD, vXpath);

System.out.println("Test is a" + vError); // These methods return something and they are only print. If this line fails, it still going as a pass.They will not throw exceptions

//

}

if (vKW.equalsIgnoreCase("waitTime")){

KW.waitTime(3000);

}

if (vError.equalsIgnoreCase("Pass")){

System.out.println("Pass" + xTC[i][0] + "\_" + vKW);

xTS[j][8] = "Pass";

} else {

System.out.println("Fail" + xTC[i][0] + "\_" + vKW);

xTS[j][8] = "Fail";

xTS[j][9] = "Error happened as function returned Fail" ;

xTC[i][4] = "Fail";

}

} catch (Exception myError){ //If an error occurred then the code will go to the catch exception. Anything to do with Application will get captured here.

System.out.println("Fail" + xTC[i][0] + "\_" + vKW);

xTS[j][8] = "Fail";

//System.out.println("Error happened" + myError);

xTS[j][9] = "Error happened" + myError;

xTC[i][4] = "Fail";

}

}

}

}

}

/Write back the test steps to Excel

XL1.writeXL("C:\\Priya\\Silverback 4.3\\Results\\KDF-TCResults-"+xTD[k][0]+".xls", "TestCases Results", xTC);

XL1.writeXL("C:\\Priya\\Silverback 4.3\\Results\\KDF-TSResults-"+xTD[k][0]+".xls", "TestSteps Results", xTS);

}

}

myD.quit();

}

private String getTDValue(String vTestData) {

// TODO Auto-generated method stub

if (vTestData.equals("dURL")){

return dURL; //returns the value in dURL

}

if (vTestData.equals("dilogin")){

return dilogin; //returns the value in dURL

}

if (vTestData.equals("dipswd")){

return dipswd; //returns the value in dURL

}

if (vTestData.equals("dExpText")){

return dExpText; //returns the value in dURL

}

//return "No test data found "; //either return the values set above,or return the values as is such as the expected results or

return "vTestData"; //returns the vTest Data as is such as the verifications. Taking vtest data as is...

}

public void executeKW(String vKW, String vXP, String vTD){

}

}

Time taken… 33 mins

Wait for Element:

Record on Selenium – Show All Available Commands – WaitForElementPresent id=wherebox

Wait for Element Keyword

**public** **static** String waitForElement(FirefoxDriver driver, String xPath) **throws** Exception{

// Purpose: it will go into a finite loop and see if a spcific element is present. If element is present, it

//i/p: time in milli seconds

//o/p: NIl

//CreatedBy : Priya

//When :8/08/14

//EditedBy : Priya

//When : 8/8/14

**for** (**int** second = 0;; second++) {

**if** (second <20) {

**if** (*isElementPresent*(driver, By.*xpath*(xPath))) {

**return** "Pass";

}

} **else** {

**return** "Fail";

}

Thread.*sleep*(1000);

}

}

**private** **static** **boolean** isElementPresent(FirefoxDriver driver, By by) {

**try** {

driver.findElement(by);

**return** **true**;

} **catch** (NoSuchElementException e) {

**return** **false**;

}

}

}

Uploading files:

WebElement El = *driver*.findElement(By.*id*("IpaFile"));

El.sendKeys("C:\\Priya\\work\\Apps files\\Applications\\RESIGNED\\Configuration App360RESIGNED.ipa");

WebElement El1 = *driver*.findElement(By.*id*("PlistFile"));

El1.sendKeys("C:\\Priya\\work\\Apps files\\Applications\\RESIGNED\\Configuration AppSPBrowser.plist");

*driver*.findElement(By.*xpath*("/html/body/div[2]/div[2]/form/div/div/p/input[1]")).click();

**Day16**

**SVN (Apache SubVersion)**

Apache Subversion is a tool to store the documents centralised.

Go to website https://subversion.apache.org/

Download the latest SVN software ‘**2014-05-19 — Apache Subversion 1.7.17 Released**

**To store the files,**

**• Go to** code.google.com and create a development project – here is where we store all our files for the project.

**• When we go into Automation, there are project files distributed, it is good to use and store in SVN.**

• Create a new project on code.google.com

•

• After the project is created, this is the url where we store the project

https://code.google.com/p/sampleproject-priya/

• Click Source and it gives an external link as below:

• Copy the location of the trunk and out paste in a new url

**http**://sampleproject-priya.googlecode.com/svn/trunk/ .

• It will display the following, if any files loaded, will show up here.

•

• http://sampleproject-priya.googlecode.com/svn/trunk/

• username priyasilverback@gmail.com

• Click Generate password and put it here : **NW6Et3Qz3Da2**

**How to interact with it:**

• To interact with it, you should be able to use the subversion client so you can view the files.

• It helps to take code from the system and puts it to the server that I created.

• Search for subversion client on google and download the latest tortisesvn client: http://tortoisesvn.net/downloads.html

• Install the client, When right clicked on any Selenium folder, it should give the following:

• It creates SVN check out and Tortise SVN

• These 2 above options lets me take my files and put it on the server on that trunk url. So I can upload from my system and put in there

• How do I take the local files and put onto the server.

Go to that Selenium folder right click on Selenium server TortiseSVN Import (This imports to the server)

•

**• https**://priya-selenium-codes.googlecode.com/svn/trunk

• Put the URL – it will then prompt for a username and password

• priyasilverback@gmail.com; NW6Et3Qz3Da2

• The files get uploaded to the trunk url

• Getting the files to the server – export from the trunk to the folder.

• Create a folder and right click export, it will prompt from where I want to export

• The url of repository lets you browse accessing that folder and get those files back to the folder on the system.

• Now Priya has made a change. If another user has made changes, the url doesn’t get the changes made by another user on their local systems. On the system, right click and check out, then make any changes and re-import. It creates a folder on the local system. Right click on that new folder and click **commit SVN**

• So in a project, users are in different locations across the world. When we start the project, we start by creating a folder into it and then add files. There could be an admin user who controls access of the structure. Every user that wants to read it takes the latest version from that central repository. If a user wants to change or modify, on the local machine they check out that file. After they made changes to that specific file, they do a commit.

• So the next user can see the changes the second user made. 2 users cannot check out at the same time.

• How do we use it in Eclipse kind of environment? For Eclipse, we have an Eclipse plug in. Search on google for “Subclipse” – go to subclipse.tigris.org – click on download and install and click ‘latest version’. Copy the url of the latest version http://subclipse.tigris.org/update\_1.10.x

• Subclipse is like tortoise SVN but specifically triggered for Eclipse.

• Eclipse has a folder structure

• Launch Eclipse, go to help – install new software – add the url from above

• Select the below 2 and click next, accept the terms and finish it.

• Restart Eclipse.

• On Eclipse now create a new project. File New project other SVN (instead of Java project)

• Select checkout projects from SVN and click next.

• Select create a new repository location. And give the trunk url. Click finish. Prompts you to select a folder. Select Java new project and give a name. This will sync the online svn content to Eclipse.

• On the Eclipse, after the sync, it creates all codes to under jre system library.

• Right click on source and create a new class. Created an annotation @test. (usually any new project created needs the jar files added…. But when I selected import, I choose the option add junit4 library to the build path. It imported the junit class.

• Write a simple code such as a print statement. Now I can take this code and put it over the trunk url.

• Everytime I make changes, I right click on the project folder (the entire trunk) and select

• Right click on the project folder team commit. Write in the text what changes you made and click on.

• Now it’s going to **check in** whatever changes I have made.

• Refresh the trunk URL, it shows the changes I made in the src folder

**• Checkout – make changes – put it back**

• Make changes, right click on the code, click commit, put the comments and click ok. The changes are uploaded to the URL.

• You can also update revisions.

• You can also check the history.

• You can also compare the code changes. Bring the history. Right click on code and compare.

• Checkout – I am going to work on this file. When it is in checked out, no one else can view it. It is like taking control

• Take an existing project to SVN.

• Create a project and then take the entire folders and update.

**Day17**

**Maven – Continuous Build Integration**

• With SVN or subversion, I am using different Java or text files, xml, jar files instead of putting on local machine you are putting them in a centralised location.

• So I wrote a test annotation and it needed the junit import. If I have a webDriver myD = new FirefoxFriver(); I need to associate various Selenium projects with it by going into java build path. I am adding jar files and selenium server standalone from the local machine. Unless I do these, my firefox driver doesn’t recognise.

• Let’s say I go to myD.naviagte().to(“Google.com”); this is an SVN code. Now if I synchronise the SVN code with what we saw on the server, the jar files will not go into the server (JRE system library files) because those are the large executable files. So these files have to be associated with specific dependencies on them. For example for my projected I associated with Selenium server version 2.5. I want everyone in the team to be associated with the same JAR version. If Selenium launches newer version. Then each of them have to go and download the latest one. So if I go and upgrade to the newer version and commit changes. Everyone may not be able to see the changes I made. So I should be able to point and get this from the cloud directly.

• So I have a project and have jar files associated with my project. Instead of storing the jar files that we associated with the project to this project, we will associate them in a larger place cloud. So that jar file or selenium server , point to cloud, it goes and gets it from cloud every time I commit the code.

• Similarly other teams are also pointing to that specific location. It’s a common place where everyone can go and fetch it

• So if Selenium is upgrading the version, instead of we physically downloading, we can automate to go and fetch it and upgrade it automatically.

• SVN and Maven are version control and build integration

• Search for Maven on google. http://maven.apache.org/

• You can also go to the apapchi maven wikipedia

• Download the latest bin file.

• Unzip and copy somewhere such as in Selenium folder. All I need to do is to create some system environment variables.

• Go to cmd prompt and type mvn , maven should get reconised.

• When I type java from any folder, it returns the java bcoz it is created in environmental variables.

• Add the following to the system variable. The variable value is wherever the Maven folder is extracted to.

• The above dint work. Try the following:

• Set up Maven Home

• Properties -> Advanced -> Environment Variables -> System variables -> New

• Variable Name: *MAVEN\_HOME*  
Variable Value: *C:\\Software\\apache-maven-2.1.0-M1*

• Note: The variable value depends on where you extracted the maven. The above is as on my system.

• 1.3. Update Windows Path

•  If you have closed System Properties window, repeat Mini Step 1 to launch it.

• Properties -> Advanced -> Environment Variables -> System Variables

• Select "**Path**" and then Click "**Edit**". Then Add ";**%MAVEN\_HOME%\\bin**" at the end and  then click "OK".

•

• Add JAVA\_HOME to the system variable : C:\Program Files\Java\jdk1.8.0\_05

• Go to cmd and type mvn, it brings up BUILD failure etc, its ok if it shows failure but atleast system is recognising the mvn

• So whenever I run mvn as a command, it executes.

• Now we need to create a Maven project to a command prompt or from Eclipse.

• Go to file java project and create one example Mavetest

• Right click on the project created Team select Share folder Select SVN click next – Point to existing repository or create a new location

• Select project name as the folder name and click finish. It will upload whatever empty code has been created to trunk.

• Now I need to know how to point to a specific version of junit

• Go to cmd and go to the location of that project created. In this case, it is C:\Users\priyap@airloom.com.au\workspace\ and type DIR. It shows the project created.

• type ‘**mvn archetype:generate** press enter This will show number of projects and associated versions on the system.

• Either generate from cmd or create a java project associate with SVN right click on project configure to maven project.

• You could start with a specific template called pom.xml

• Go to Apache maven – wiki - Maven projects are configured using a Project Object Model, which is stored in a pom.xml-file. Here's a minimal example:

**<project>**

*<!-- model version is always 4.0.0 for Maven 2.x POMs -->*

**<modelVersion>**4.0.0**</modelVersion>**

*<!-- project coordinates, i.e. a group of values which*

*uniquely identify this project -->*

**<groupId>**com.mycompany.app**</groupId>**

**<artifactId>**my-app**</artifactId>**

**<version>**1.0**</version>**

*<!-- library dependencies -->*

**<dependencies>**

**<dependency>**

*<!-- coordinates of the required library -->*

**<groupId>**junit**</groupId>**

**<artifactId>**junit**</artifactId>**

**<version>**3.8.1**</version>**

*<!-- this dependency is only used for running and compiling tests -->*

**<scope>**test**</scope>**

**</dependency>**

**</dependencies>**

**</project>**

• XML: Project is the parent element. Under it there are different child elements and nodes and there are values between each attribute and nodes.. The group Id is to with the specific project. Dependencies: What is your project dependent on, junit jar files, test NG jar files, Selenium jar files. All these different dependencies can be mentioned . Under dependencies, there can be more dependencies.

• Group id is the common recognisable project id : http://search.maven.org/#search%7Cga%7C1%7Cjunit

The following shows for junit. There are group ids and artifacts versions is what is showing on this xml above

• Artificat id is the name of the project

• To create an XML file from the scratch, copy the above file into a notepad: and edit the group id’s artifacts etc or go to cmd and generate the the maven

• It will create a pom..xml

• Edit the pom file and we can copy the edited notepad and put in whatever dependencies we need and remove dependencies that we don’t need.

• There will 2 folders with every Maven test, A **TEST** folder and a main folder within which we have all the different tests created. **MAIN** is where we will build and execute

• A pom.xml gets generated. Click on pom.xml, it opens up with all the dependencies.

• I can put the dependencies.

• Now we need to check in the changes to the trunk. Right click on the project Team commit

• So the point of this so far is we don’t have to upload jar files from local files but get updated from online.

• Capturing Screeshots: code

@Test

**public** **void** ss1() **throws** IOException {

FirefoxDriver myD = **new** FirefoxDriver();

**try** {

myD.navigate().to("http://www.google.com");

myD.findElement(By.*xpath*("//\*[@id='gbqfq']")).sendKeys("plugins");

myD.findElement(By.*xpath*("//\*[@id='q']")).sendKeys("eclips");

} **catch** (NoSuchElementException e1) { //exception syntax

System.*out*.println("Element not found" + e1);

String Screenshotpath = "C:\\Priya\\err.jpg";

File scrN = **null**;

scrN = (File)((TakesScreenshot)myD).getScreenshotAs(OutputType.*FILE*);

FileUtils.copyFile(scrN, **new** File(Screenshotpath));

}

**DAY18 (Maven part2)**

• Maven does continues dependencies

• When working on a project, there are some external dependencies that have their own versions such as jar files, selenium server files and poi files that are on my local machine currently.

• Maven helps to integrate with the versions.

• Lets say we are using Selenium server 2.5 and other person is using 2.1 . So when I say POM.xml is 2.5, every one will get upgraded to 2.5. What files and dependencies, we will mention in the POM file.

• These are all different types of artifcats.

• All the lists with the current version gets updated in this link : http://search.maven.org/

• I go search for Selenium for example and click on the version number, I can see the dependencies

• There are 2 ways of starting these Maven projects. 1. CMD 2. Eclipse.

Install a software called M2E

• Go to CMD

• Create a Base skeleton structure

• Go to that folder where the projects are and type the following

Mvn archetype:generate –DgroupId=myGroup –DartifactId=my-proj –DarchtypeArtifactId=maven-archtype-quickstart –DinteractiveMode=false

Mvn archetype:generate –DgroupId=myGroup –DartifactId=mva12 -DarchtypeArtifactId=maven-archtype-quickstart –DinteractiveMode=false

• From Eclipse

I can go import the existing project into workspace, and specify the folder from the path for the project created above, it says no projects are found to import.

In that case, go to that folder (basically I was getting the file generated from cmd but doesn’t show on Eclipse) for that type this

Mvn eclipse:eclipse

Go to Eclipse and import

Browser to that folder which was created by cmd and add it. It is now created and it has a jar file under referenced libraries

• Go to main main test – add an annotation, Junit annotation is still not downloaded.

• You can go to Maven bin folder to check the source path : C:\Priya\Selenium\apache-maven-3.2.2\Bin – open the m2 config file. This shows the path where the jar files will be stored

• Previously I used to add the external jar files from my local machine in order to import, now I can specify all that in a POM file in a dependency

• I can go to that central repository and search for Selenium , click on Selenium server – version, copy the maven dependency and paste in the POM file or

•

OR

• On POM file, I can click on dependencies and add the version numbers etc, It will do the same thing instead of updating physically

• Save everything and before I do anything run the command from the project folder as

mvn eclipse:eclipse

This will go and locate all the associated jar files and gets them.

• Go to Eclipse, and do a refresh on the project and it will download all the jar files. Now all the reference libraries are downloaded onto the local machine in .m2 folder

**References**:

http://svnbook.red-bean.com/

http://en.wikipedia.org/wiki/Apache\_Maven

http://search.maven.org/#search%7Cga%7C1%7Cjunit : List of central repositories you can search.

http://search.maven.org/

Downloads:

http://maven.apache.org/

Install M2E plugin - http://www.eclipse.org/m2e/

FAQ’s

• Generating pom.xml

Hi,  
I tried to generate a pom.xml via command prompt and it fails. I can see the project I created does exist in the directory and I run mvn.archetype generate command from the same directory. It takes me through the group id , artificateis version etc. it also says the build is SUCCESS.  
But when I go to Eclipse and refresh the projects, I don't see a pom file generated. please help

To create POM.XML file in Eclipse:

Install M2E plugin (http://www.eclipse.org/m2e/)

Right click on project -> Configure -> Convert to Maven Project

Multiple browsers:

Create a method to call the type of browser Firefox or chrome or IE  
  
Following is a method to send the type of browser  
  
public class Keyword {  
    static WebDriver myDriver;  
  
public static void openBrowser(String vType){  
        
        myDriver = new FirefoxDriver();  
        myDriver = **new** InternetExplorerDriver();   
        myDriver =new ChromeDriver();    
          
    }

Sort a flexgrid : This is what I tried:

Following is my code. When I run this, I get the list of data from a flexigrid, Once I get the data, how do I sort it in Ascending/descending order?

package Experiments;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Iterator;

import java.util.List;

import java.util.SortedSet;

import java.util.TreeSet;

import junit.framework.Assert;

import org.junit.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.firefox.FirefoxDriver;

public class getdatafromawebtable {

static FirefoxDriver driver = new FirefoxDriver();

String BaseURL = “https://qa.example.com/”;

int iwait = 3000;

@Test

public void sorting() throws Exception {

driver.get(BaseURL);

driver.findElement(By.id(“Username”)).sendKeys(“username”);

driver.findElement(By.id(“Password”)).sendKeys(“pswd”);

driver.findElement(By.xpath(“//input[@type=’input’]”)).click();

driver.findElement(By.xpath(“"xpath)).click();

WebElement table = driver.findElement(By.xpath(“//descendant::table[4]”));

List rows = table.findElements(By.tagName(“tr”));

int row\_count = rows.size();

System.out.println(“Total rows is ” + row\_count);

IteratorI1 = rows.iterator();

while(I1.hasNext()){

WebElement data = I1.next();

List columns= data.findElements(By.tagName(“td”));

Iterator I2=columns.iterator();

while(I2.hasNext()){

WebElement text=I2.next();

String columndata=text.getText();

System.out.print(” “);

System.out.print(columndata);

System.out.print(” “);

}

System.out.println(“”);

}

}

}

Get ALL the values from a dropw down:

List<WebElement> allElements = driver.findElements(By.*xpath*("//select[@id='DeviceTypeId']"));

**for** (WebElement element: allElements) {

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

}

Get Tag ID/row id from the tags tab

String tagId = driver.findElement(By.xpath("/html/body/div[2]/div[2]/div[2]/div/div[6]/table/tbody/tr/td[1]/div")).getText();

CSS

Web App Automate tool:

Auto it and suikuli