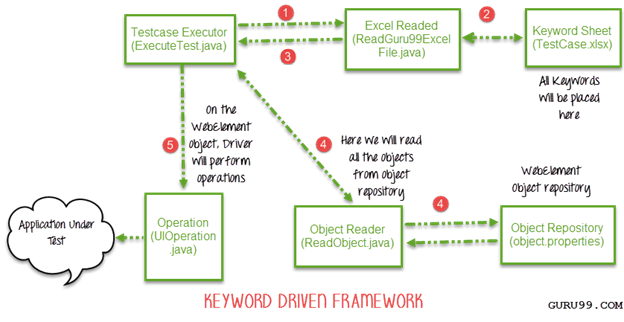
**Keyword Driven Test Framework:**

In keyword driven test framework, all the operations and instructions are written in some external file like Excel worksheet. Here is how the complete framework looks like

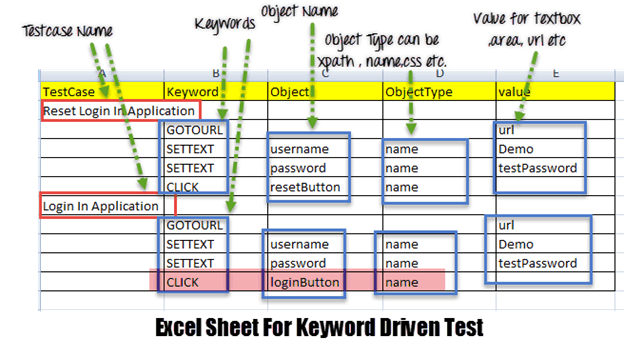


**Step 1)**

* The driver script Execute.java will call Read ExcelFile.java
* Read ExcelFile.java has POI script to read data from an Excel

**Step 2)**

* Read ExcelFile.java will read data from TestCase.xlsx
* Here is how the sheet looks like-

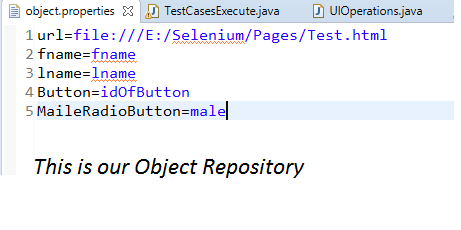


* According to the keywords written in Excel file, the framework will perform the operation on UI.
* For example, we need to click a button 'Login.' Correspondingly, our Excel will have a keyword 'Click.' Now the AUT can have hundreds of button on a page, to identify a Login button, in Excel we will input Object Name as loginButton & object type as a name (see highlighted the row in above image). The Object Type could be Xpath, name CSS or any other value

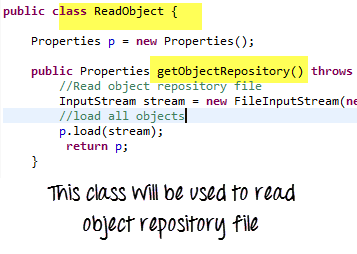
**Step 3)** ReadExcelFile.java will pass this data to the driver script Execute.java

**Step 4)**

* For all of our UI web elements, we need to create an object repository where we will place their element locator (like Xpath, name, CSS path, class name etc.)



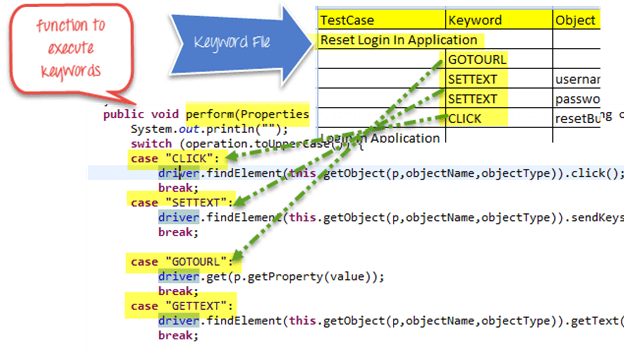
* Execute.java (our driver script) will read the entire Object Repository and store it in a variable
* To read this object repository, we need a ReadObject class which has a getObjectRepository method to read it.

[](https://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey5.png)

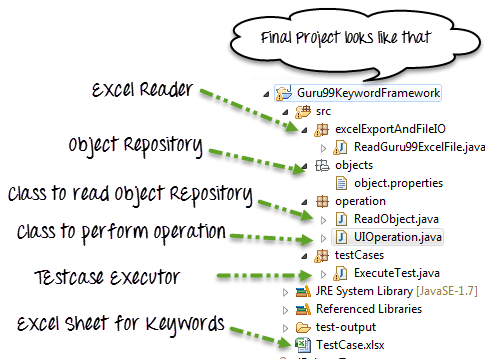
**NOTE:** Uses of OR – it helps in code maintenance. For example, we are using the button with name = btnlogin in 10 different test cases. In future, the developer decides to change the name from btnlogin to submit. You will have to make a change in all the 10 test cases. In the case of an object repository, you will make the change just once in the repository.

**Step 5)**

* The driver will pass the data from Excel & Object Repository to UIOperation class
* UIOperation class has functions to perform actions corresponding to keywords like CLICK, SETTEXT etc… mentioned in the excel
* UIOperation class is a Java class which has the actual implementation of the code to perform operations on web elements



The complete project will look like-



Test Scenario

* We are executing 2 test cases
* Test Case 1:
* Goto http://demo. .com/V4/
* Enter FirstName Data
* Enter LastName Data
* Click Submit
* Test Case 2:
* Goto http://demo. .com/V4/
* Enter FirstName Data
* Enter LastName Data
* Click Submit

**object.properties**

url=E:/Selenium/Pages/Test.html

username=uid

password=password

title=barone

loginButton=btnLogin

resetButton=btnReset

**Read ExcelFile.java**

package excelExportAndFileIO;

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

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

import org.apache.poi.ss.usermodel.Sheet;

import org.apache.poi.ss.usermodel.Workbook;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class Read ExcelFile {

public Sheet readExcel(String filePath,String fileName,String sheetName) throws IOException{

//Create a object of File class to open xlsx file

File file = new File(filePath+"\\"+fileName);

//Create an object of FileInputStream class to read excel file

FileInputStream inputStream = new FileInputStream(file);

Workbook Workbook = null;

//Find the file extension by spliting file name in substing and getting only extension name

String fileExtensionName = fileName.substring(fileName.indexOf("."));

//Check condition if the file is xlsx file

if(fileExtensionName.equals(".xlsx")){

//If it is xlsx file then create object of XSSFWorkbook class

Workbook = new XSSFWorkbook(inputStream);

}

//Check condition if the file is xls file

else if(fileExtensionName.equals(".xls")){

//If it is xls file then create object of XSSFWorkbook class

Workbook = new HSSFWorkbook(inputStream);

}

//Read sheet inside the workbook by its name

Sheet Sheet = Workbook.getSheet(sheetName);

return Sheet;

}

}

**ReadObject.java**

package operation;

import java.io.File;

import java.io.FileInputStream;

import java.io.IOException;

import java.io.InputStream;

import java.util.Properties;

public class ReadObject {

Properties p = new Properties();

public Properties getObjectRepository() throws IOException{

//Read object repository file

InputStream stream = new FileInputStream(new File(System.getProperty("user.dir")+"\\src\\objects\\object.properties"));

//load all objects

p.load(stream);

return p;

}

}

**UIOperation.java**

package operation;

import java.util.Properties;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

public class UIOperation {

WebDriver driver;

public UIOperation(WebDriver driver){

this.driver = driver;

}

public void perform(Properties p,String operation,String objectName,String objectType,String value) throws Exception{

System.out.println("");

switch (operation.toUpperCase()) {

case "CLICK":

//Perform click

driver.findElement(this.getObject(p,objectName,objectType)).click();

break;

case "SETTEXT":

//Set text on control

driver.findElement(this.getObject(p,objectName,objectType)).sendKeys(value);

break;

case "GOTOURL":

//Get url of application

driver.get(p.getProperty(value));

break;

case "GETTEXT":

//Get text of an element

driver.findElement(this.getObject(p,objectName,objectType)).getText();

break;

default:

break;

}

}

/\*\*

\* Find element BY using object type and value

\* @param p

\* @param objectName

\* @param objectType

\* @return

\* @throws Exception

\*/

private By getObject(Properties p,String objectName,String objectType) throws Exception{

//Find by xpath

if(objectType.equalsIgnoreCase("XPATH")){

return By.xpath(p.getProperty(objectName));

}

//find by class

else if(objectType.equalsIgnoreCase("CLASSNAME")){

return By.className(p.getProperty(objectName));

}

//find by name

else if(objectType.equalsIgnoreCase("NAME")){

return By.name(p.getProperty(objectName));

}

//Find by css

else if(objectType.equalsIgnoreCase("CSS")){

return By.cssSelector(p.getProperty(objectName));

}

//find by link

else if(objectType.equalsIgnoreCase("LINK")){

return By.linkText(p.getProperty(objectName));

}

//find by partial link

else if(objectType.equalsIgnoreCase("PARTIALLINK")){

return By.partialLinkText(p.getProperty(objectName));

}else

{

throw new Exception("Wrong object type");

}

}

}

**ExecuteTest.java**

package testCases;

import java.util.Properties;

import operation.ReadObject;

import operation.UIOperation;

import org.apache.poi.ss.usermodel.Row;

import org.apache.poi.ss.usermodel.Sheet;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.annotations.Test;

import excelExportAndFileIO.Read ExcelFile;

public class ExecuteTest {

@Test

public void testLogin() throws Exception {

// TODO Auto-generated method stub

WebDriver webdriver = new FirefoxDriver();

Read ExcelFile file = new Read ExcelFile();

ReadObject object = new ReadObject();

Properties allObjects = object.getObjectRepository();

UIOperation operation = new UIOperation(webdriver);

//Read keyword sheet

Sheet Sheet = file.readExcel(System.getProperty("user.dir")+"\\","TestCase.xlsx" , "KeywordFramework");

//Find number of rows in excel file

int rowCount = Sheet.getLastRowNum()- Sheet.getFirstRowNum();

//Create a loop over all the rows of excel file to read it

for (int i = 1; i < rowCount+1; i++) {

//Loop over all the rows

Row row = Sheet.getRow(i);

//Check if the first cell contain a value, if yes, That means it is the new testcase name

if(row.getCell(0).toString().length()==0){

//Print testcase detail on console

System.out.println(row.getCell(1).toString()+"----"+ row.getCell(2).toString()+"----"+

row.getCell(3).toString()+"----"+ row.getCell(4).toString());

//Call perform function to perform operation on UI

operation.perform(allObjects, row.getCell(1).toString(), row.getCell(2).toString(),

row.getCell(3).toString(), row.getCell(4).toString());

}

else{

//Print the new testcase name when it started

System.out.println("New Testcase->"+row.getCell(0).toString() +" Started");

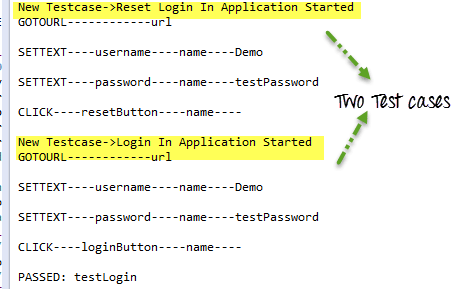
}

}

}

}

After execution, output will look like -

[](https://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey8.png)