**Hybrid Framework**

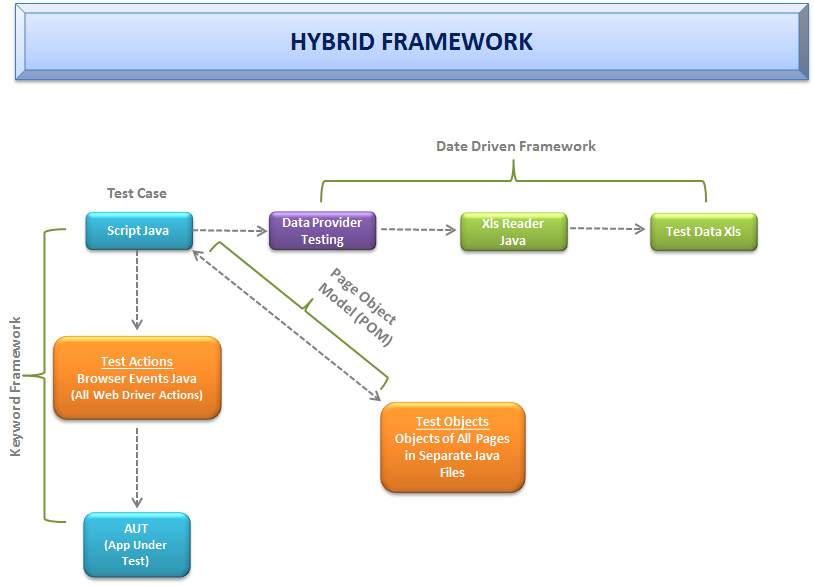
In this example, we use ***Data Driven***, ***Keyword Driven*** and ***Page Object Model*** with ***TestNG***.

**Advantages of Hybrid Framework:**

1. Through ***Data Driven*** Framework- since we store all the test data in excel sheet, we can execute our test script with *Multiple Data* sets.

2. Through ***Keyword Driven*** Framework- We can maintain all *Web Actions* in a separate java file and call them from scripts using a keyword/method name.

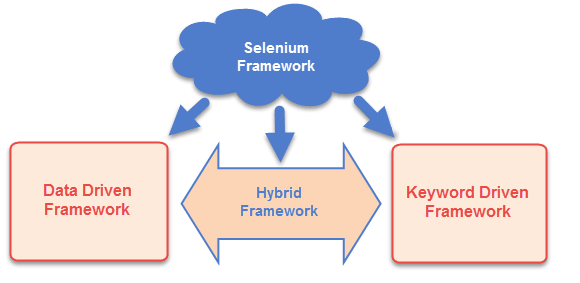
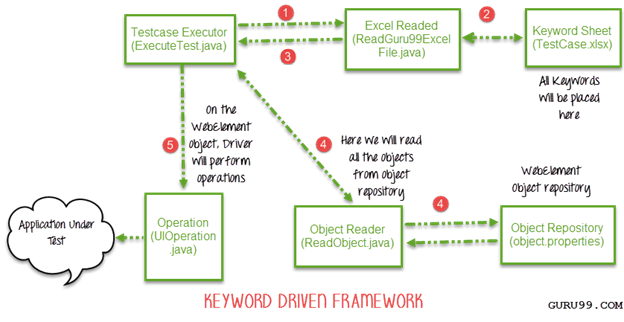
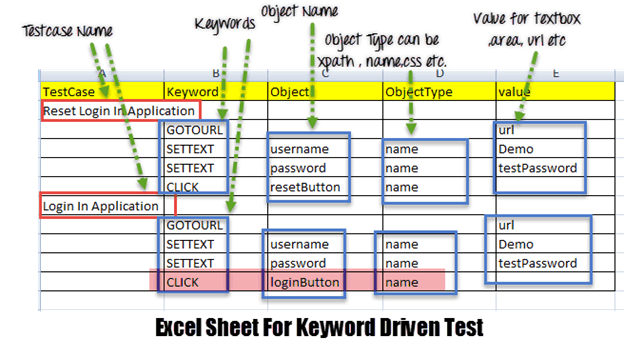
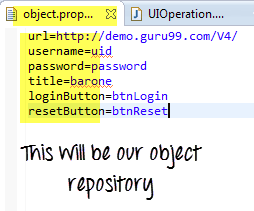
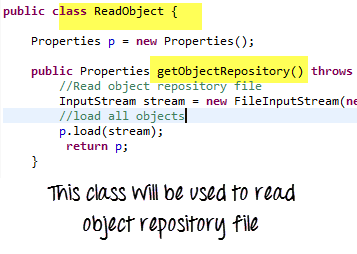
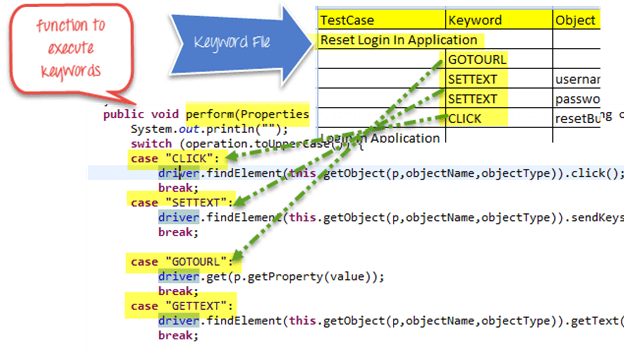
3. Through ***Page Object Model***- For each *Web page*, we can maintain all *test objects* in a java file separately and call them from scripts. If any object changes in one page, we can selectively modify the object for that page. Hence we can loosely couple the data from test cases.

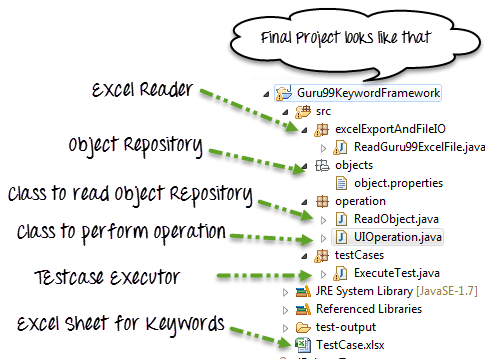


Creating Keyword & Hybrid Frameworks with Selenium

Frameworks help to structure our code and make maintenance easy. Without frameworks we will place all our code and data in same place which is neither re-usable nor readable. Using Frameworks, produce beneficial outcomes like increase code re-usage , higher portability , reduced script maintenance cost etc

There are mainly three type of frameworks created by Selenium WebDriver to automate manual testcases

* Data Driven Test Framework
* Keyword Driven Test Framework
* Hybrid Test Framework
* Test Framework
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey1.png)
* Data Driven Test Framework
* In data driven framework all of our test data is generated from some external files like excel, csv, XML or some database table. We already learned about Data Driven [testing](http://www.guru99.com/software-testing.html) in our previous [tutorial](http://www.guru99.com/all-about-excel-in-selenium-poi-jxl.html)
* 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
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey2.png)
* As you can see it's a 5 step framework. Let's study it stepwise in detail
* Step 1)
* The driver script Execute.java will call ReadGuru99ExcelFile.java
* ReadGuru99ExcelFile.java has POI script to read data from an Excel
* Step 2)
* ReadGuru99ExcelFile.java will read data from TestCase.xlsx
* Here is how the sheet looks like-
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey3.png)
* 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 name (see highlighted row in above image). The Object Type could be xpath,name css or any other value
* Step 3) ReadGuru99ExcelFile.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.)
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey4.png)
* 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.
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey5.png)
* NOTE: You may think why do we need to create an object repository. The answer it helps in code maintainence. For example, we are using the a 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 change in all the 10 test cases. In case of an object repository, you will make change just once in the repository.
* Step 5)
* The driver will pass the data from Excel & Object Repositoy to UIOperation class
* UIOperation class has functions to perfom 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
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey6.png)
* The complete project will looks like-

[](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey7.png)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | 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 ReadGuru99ExcelFile {        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 guru99Workbook = 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      guru99Workbook = 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          guru99Workbook = new HSSFWorkbook(inputStream);      }      //Read sheet inside the workbook by its name      Sheet guru99Sheet = guru99Workbook.getSheet(sheetName);       return guru99Sheet;      }  } |

* ReadObject.java
* [?](http://www.guru99.com/creating-keyword-hybrid-frameworks-with-selenium.html)

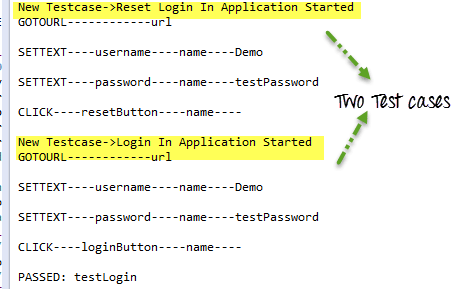
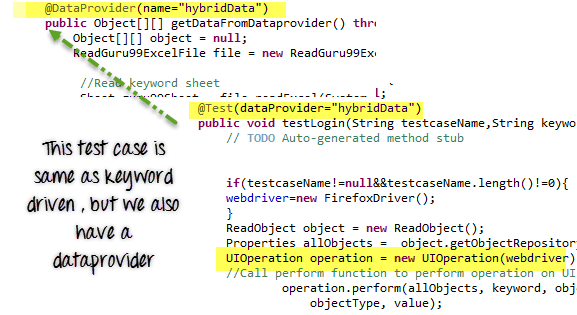
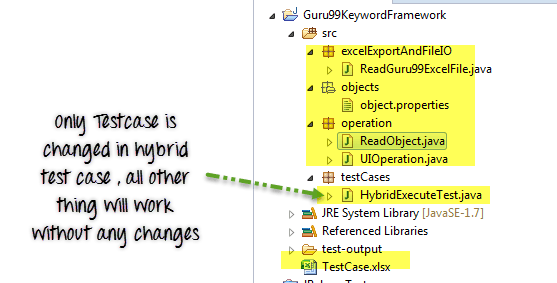
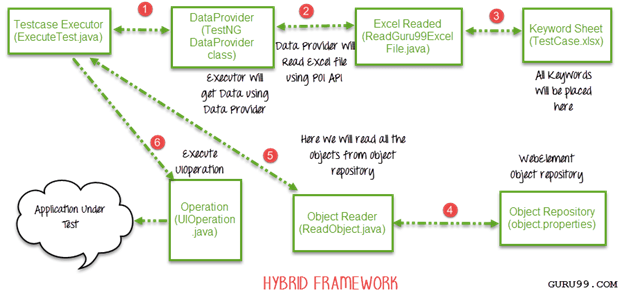
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | package operation;  import java.io.File;  import java.io.FileInputStream;  import java.io.IOException;  import java.io.InputStream;  import java.util.Properties;  public class ReadObject {      Properties p = new Properties();      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
* [?](http://www.guru99.com/creating-keyword-hybrid-frameworks-with-selenium.html)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83 | 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
* [?](http://www.guru99.com/creating-keyword-hybrid-frameworks-with-selenium.html)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43 | 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.ReadGuru99ExcelFile;  public class ExecuteTest {  @Test      public void testLogin() throws Exception {          // TODO Auto-generated method stub  WebDriver webdriver = new FirefoxDriver();  ReadGuru99ExcelFile file = new ReadGuru99ExcelFile();  ReadObject object = new ReadObject();  Properties allObjects = object.getObjectRepository();  UIOperation operation = new UIOperation(webdriver);  //Read keyword sheet  Sheet guru99Sheet = file.readExcel(System.getProperty("user.dir")+"\\","TestCase.xlsx" , "KeywordFramework");  //Find number of rows in excel file      int rowCount = guru99Sheet.getLastRowNum()-guru99Sheet.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 = guru99Sheet.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 -
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey8.png)
* [Download the Selenium Project Files for the Demo in this Tutorial](https://drive.google.com/uc?export=download&id=0B_vqvT0ovzHcZkxzUmpPcHhnOUU)
* Hybrid Test Framework
* Hybrid Test framework is a concept where we are using advantage of both Keyword and Data driven framework.
* Here for keywords we will use excel files to maintain test cases and for test data we can use data provider of TestNG framework.
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey9.png)
* Here in our hybrid framework we don't need to change anything in Keyword driven framework , here we just need to replace ExecuteTest.java file with HybridExecuteTest.java file.
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey10.png)
* This HybridExecuteTest file has all the code for keyword driven with data provider concept.
* The complete pictorial representation of hybrid framework will look like
* [](http://www.guru99.com/images/AdvanceSelenium/071514_0715_CreatingKey11.png)
* HybridExecuteTest.java
* [?](http://www.guru99.com/creating-keyword-hybrid-frameworks-with-selenium.html)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51 | package testCases;  import java.io.IOException;  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.DataProvider;  import org.testng.annotations.Test;  import excelExportAndFileIO.ReadGuru99ExcelFile;  public class HybridExecuteTest {      WebDriver webdriver = null;  @Test(dataProvider="hybridData")      public void testLogin(String testcaseName,String keyword,String objectName,String objectType,String value) throws Exception {          // TODO Auto-generated method stub        if(testcaseName!=null&&testcaseName.length()!=0){      webdriver=new FirefoxDriver();      }  ReadObject object = new ReadObject();  Properties allObjects = object.getObjectRepository();  UIOperation operation = new UIOperation(webdriver);      //Call perform function to perform operation on UI              operation.perform(allObjects, keyword, objectName,                  objectType, value);        }  @DataProvider(name="hybridData")      public Object[][] getDataFromDataprovider() throws IOException{      Object[][] object = null;      ReadGuru99ExcelFile file = new ReadGuru99ExcelFile();  //Read keyword sheet  Sheet guru99Sheet = file.readExcel(System.getProperty("user.dir")+"\\","TestCase.xlsx" , "KeywordFramework");  //Find number of rows in excel file      int rowCount = guru99Sheet.getLastRowNum()-guru99Sheet.getFirstRowNum();      object = new Object[rowCount][5];      for (int i = 0; i < rowCount; i++) {          //Loop over all the rows          Row row = guru99Sheet.getRow(i+1);          //Create a loop to print cell values in a row          for (int j = 0; j < row.getLastCellNum(); j++) {              //Print excel data in console              object[i][j] = row.getCell(j).toString();          }      }      System.out.println("");       return object;      }  } |

* Summary:
* We can create three types of test framework using selenium WebDriver.
* These are Data Driven, Keyword driven and Hybrid test framework.
* We can achieve Data driven framework using TestNG's data provider.
* In Keyword driven framework , keywords are written in some external files like excel file and java code will call this file and execute test cases.
* Hybrid framework is a mix of keyword driven and data driven framework.