Frameworks

A test automation framework is a set of assumptions, concepts, and practices that provide support for automated software testing.

Based on the above definition there are numerous framework models developed and are being used in the industry. Some of the current framework models are listed as below.

* Modular Framework
* Keyword Driven model
* Data Driven model
* Hybrid Model

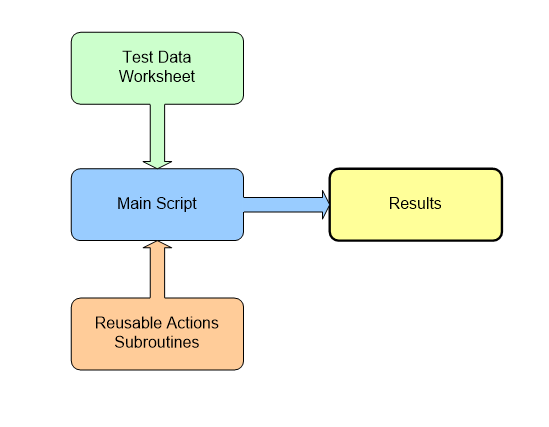
# Why framework?

* Reduce Rework
* Uniform organization of test assets for easy maintenance
* Script Independence
* Implement Best practice

## Data Driven framework

The test data is maintained in an Excel sheet which is used to data drive the test.

Any changes to the data require a change in the excel sheet only and any change in the application (UI or functional) requires modifications in the corresponding Reusable Actions/Subroutines.



**Dim** objExcel, objWorkbook, objWorksheet, RowsCount  
**Set** objExcel = **CreateObject**("Excel.Application")  
**Set** objWorkbook = objExcel.Workbooks.Open("D:\Training\Cookies\newexcelfile1.xlsx")  
**Set** objWorksheet = objWorkbook.Worksheets(1)  
  
Count = objWorksheet.usedRange.rows.Count  
**msgbox** Count  
**For** i = 1 **To** Count **Step** 1  
    **msgbox** objWorksheet.Cells(i, 1)  
**Next**  
objExcel.Quit  
**Set** objWorksheet = **Nothing**  
**Set** objWorkbook = **Nothing**  
**Set** objExcel = **Nothing**

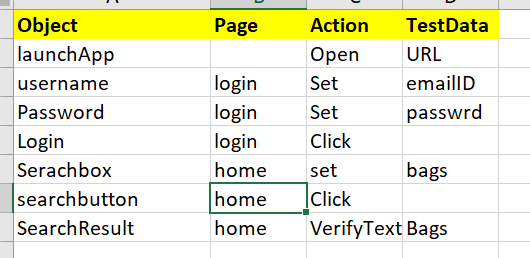
Example: Perform Data driven test for Login and search

# Keyword Driven Framework

In this approach, the aim is to build a lot of application-independent reusable keyword components so that they can be directly used for another application without spending any extra effort.

Keyword-driven testing is a technique that separates much of the programming work from the actual test steps so that the test steps can be developed earlier and can often be maintained with only minor updates, even when the application or testing needs to change significantly.

Sample Excel with steps

****

**---------------------------Functional Library-------------------------**

Function launchURL(URL)  
   SystemUtil.Run "C:\Program Files (x86)\Internet Explorer\iexplore.exe",URL  
End Function  
  
Function EnterInTextBox(TestData,PageName,ObjectName)  
 Browser("micclass:=Browser").Page("title:="&pageName).WebEdit(ObjectName).Set TestData  
End Function  
  
Function clickButton(PageName,ObjectName)  
 Browser("micclass:=Browser").Page("title:="&pageName).WebEdit(ObjectName).Click  
End Function  
  
Function verifyObjectExist(PageName,ObjectName)  
   If Browser("micclass:=Browser").Page("title:="&pageName).WebEdit(ObjectName).Exist Then  
       Reporter.ReportEvent micPass,"Object Exist" ,"The following object exist: "&ObjectName  
    else  
     Reporter.ReportEvent micFail,"Object Exist" ,"The following does not exist: "&ObjectName  
   End If  
End Function  
  
Function verifyText(PageName,ObjectName, expectedText)  
   Dim getObjectText  
   getObjectText = Browser("micclass:=Browser").Page("title:="&pageName).WebEdit(ObjectName).getRoProperty("text")  
   If expectedText = getObjectText Then  
       Reporter.ReportEvent micPass, "Expected and Acutal text match. Expected: "&expectedText&". Actual: "&getObjectText  
   else  
       Reporter.ReportEvent micPass, "Expected and Acutal text does not match. Expected: "&expectedText&". Actual: "&getObjectText  
   End If  
    
End Function

**----------------------------------------TestScript----------------------------------**

Dim objExcel, objWorkbook, objWorksheet, RowsCount  
Set objExcel = CreateObject("Excel.Application")  
Set objWorkbook = objExcel.Workbooks.Open("D:\Training\Cookies\newexcelfile1.xlsx")  
Set objWorksheet = objWorkbook.Worksheets(1)  
Count = objWorksheet.usedRange.rows.Count  
msgbox Count  
For i = 1 To Count Step 1  
    keyword = objWorksheet.Cells(i, 3)  
    If keyword = "Open" Then  
        launchURL(objWorksheet.Cells(i, 4))  
    ElseIf keyword = "Set"  Then  
        EnterInTextBox(objWorksheet.Cells(i, 4), objWorksheet.Cells(i, 2), objWorksheet.Cells(i, 1))  
    ElseIf keyword = "Click" Then  
           clickButton(objWorksheet.Cells(i, 2), objWorksheet.Cells(i, 1))  
    ElseIf keyword = "VerifyText" Then  
        verifyText(objWorksheet.Cells(i, 2), objWorksheet.Cells(i, 1), objWorksheet.Cells(i, 4))  
    End If  
Next  
  
objExcel.Quit  
Set objWorksheet = Nothing  
Set objWorkbook = Nothing  
Set objExcel = Nothing

# Modular Framework

In the Functional Decomposition/Modular Automation Framework software testers need to initially identify the reusable code, which requires automation. Furthermore, they also need to write reusable code in various functions.

Let’s consider an example of the Functional Decomposition Framework.

*Login -> Book a Ticket -> Insert Order -> Verify the Flight Number and Departure Time -> Exit*  
*Login -> Search Order-> Verify the order details -> Exit*

These functional components can be reused to build several test scenarios which are called “Main Scripts”.