**Database Connections**

**What is adodb connection?**  
The ADO (ActiveX Data Objects) Connection object is used to create a connection to a data source. Through this connection, you can access and manipulate a database.  
  
**What is adodb recordset?**

 The ADO Recordset object is used to hold a set of records from a database table. To be able to read database data, the data should be loaded into a recordset.

Option Explicit

Dim con,rs

Set con=createobject("adodb.connection")

Set rs=createobject("adodb.recordset")

con.open "Driver={Microsoft ODBC for Oracle};Server=QTPWorld; Uid=your\_username;Pwd=your\_password;"

rs.open "select \* from emp",con

Do while not rs.eof

VbWindow("Form1").VbEdit("val1").Set rs.fields("v1")

VbWindow("Form1").VbEdit("val2").Set rs.fields("v2")

VbWindow("Form1").VbButton("ADD").Click

rs.movenext

Loop

'Release objects

Set rs= nothing

Set con= nothing

# QTP - Automation Object Model

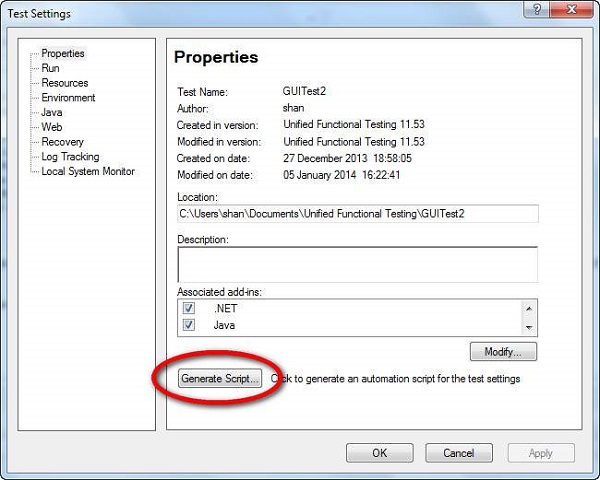
QTP itself can be automated using the COM interface that is provided by HP-QTP. Automation object model is a set of objects, methods, and properties that helps the testers to control the configuration settings and execute the scripts using the QTP interface. The Key Configurations/actions that can be controlled (but not limited to) are listed below −

* Loads all the required add-ins for a test
* Makes QTP visible while execution
* Opens the Test using the specified location
* Associates Function Libraries
* Specifies the Common Object Sync Time out
* Start and End Iteration
* Enable/Disable Smart Identification
* On Error Settings
* Data Table Path
* Recovery Scenario Settings
* Log Tracking Settings

QTP 11.5x provides an exclusive documentation on Automation Object model that can be referred by navigating to "Start" >> "All Programs" >> "HP Software" >> "HP Unified Functional Testing" >> "Documentation" >> "Unified Functional Testing Automation Reference".

## **Generate AOM Script**

A tester can generate AOM script from QTP itself, using the "Generate Script" option. Navigate to "Run" >> "Settings" >> "Properties" Tab >> "Generate Script" as shown below −



### **Example**

' A Sample Script to Demostrate AOM

Dim App 'As Application

Set App = CreateObject("QuickTest.Application")

App.Launch

App.Visible = True

App.Test.Settings.Launchers("Web").Active = False

App.Test.Settings.Launchers("Web").Browser = "IE"

App.Test.Settings.Launchers("Web").Address = "http://easycalculation.com/"

App.Test.Settings.Launchers("Web").CloseOnExit = True

App.Test.Settings.Launchers("Windows Applications").Active = False

App.Test.Settings.Launchers("Windows Applications").Applications.RemoveAll

App.Test.Settings.Launchers("Windows Applications").RecordOnQTDescendants = True

App.Test.Settings.Launchers("Windows Applications").RecordOnExplorerDescendants = False

App.Test.Settings.Launchers("Windows Applications").RecordOnSpecifiedApplications = True

App.Test.Settings.Run.IterationMode = "rngAll"

App.Test.Settings.Run.StartIteration = 1

App.Test.Settings.Run.EndIteration = 1

App.Test.Settings.Run.ObjectSyncTimeOut = 20000

App.Test.Settings.Run.DisableSmartIdentification = False

App.Test.Settings.Run.OnError = "Dialog"

App.Test.Settings.Resources.DataTablePath = "<Default>"

App.Test.Settings.Resources.Libraries.RemoveAll

App.Test.Settings.Web.BrowserNavigationTimeout = 60000

App.Test.Settings.Web.ActiveScreenAccess.UserName = ""

App.Test.Settings.Web.ActiveScreenAccess.Password = ""

App.Test.Settings.Recovery.Enabled = True

App.Test.Settings.Recovery.SetActivationMode "OnError"

App.Test.Settings.Recovery.Add "D:\GUITest2\recover\_app\_crash.qrs",

"Recover\_Application\_Crash", 1

App.Test.Settings.Recovery.Item(1).Enabled = True

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' System Local Monitoring settings

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

App.Test.Settings.LocalSystemMonitor.Enable = false

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' Log Tracking settings

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

With App.Test.Settings.LogTracking

.IncludeInResults = False

.Port = 18081

.IP = "127.0.0.1"

.MinTriggerLevel = "ERROR"

.EnableAutoConfig = False

.RecoverConfigAfterRun = False

.ConfigFile = ""

.MinConfigLevel = "WARN"

End With

**Actions**

http://www.qtpworld.com/images/border.gif

Actions is a set of logical statements to perform specific test.  
  
There are three kinds of actions:  
  
1. Non Reusable action  
2. Reusable action  
3. External action  
  
Non Reusable action:  
  
An action that can be called only in the test with which it is stored, and can be called only once.  
  
Reusable action:  
  
An action that can be called multiple times by the test with which it is stored (the local test) as well as by other tests.  
  
External action:  
  
A reusable action stored with another test.External actions are read-only in the calling test, but you can choose to use a local, editable copy of the Data Table information for the external action.  
  
  
Steps to follow to perform different operation in Actions:  
  
Create an Action:  
 Insert-->call to new action-->enter name of the action-->click OK.  
  
Rename Actions:  
Select desired action in action drop down box--> edit menu-->action-->rename action-->modify the name-->click OK.  
  
Call an Action:   
Insert-->call to existing action-->browse path of the test-->select desired action-->click OK.(Note: We cannot edit)  
  
Copy an Action:   
 Insert-->call to copy of action-->browse path of the test-->select desired action-->click OK.(Note: we can edit this action).  
  
Make an Action Reusable:   
Select Non Reusable action -->edit -->action-->action properties-->check reusable action check box -->click OK.  
  
Delete Actions:   
Select desired action in action drop down box-->edit menu-->action-->delete action-->confirm deletion.

**XML**

http://www.qtpworld.com/images/border.gif

How to get the values of the XML tags?  
  
=> Content  of  "sample.xml" file from which the tag values will retrieved using below code

**<catalog>**

**<book id="bk101">**

**<author>Gambardella, Matthew</author>**

**<title>XML Developer's Guide</title>**

**<genre>Computer</genre>**

**<price>44.95</price>**

**<publish\_date>2000-10-01</publish\_date>**

**<description>An in-depth look at creating applications**

**with XML.</description>**

**</book>**

**<book id="bk102">**

**<author>Ralls, Kim</author>**

**<title>Midnight Rain</title>**

**<genre>Fantasy</genre>**

**<price>5.95</price>**

**<publish\_date>2000-12-16</publish\_date>**

**<description>A former architect battles corporate zombies,**

**an evil sorceress, and her own childhood to become queen**

**of the world.</description>**

**</book>**

**</catalog>**

**Code**

'Create Microsoft.XMLDOM object

Set xmlDoc = CreateObject("Microsoft.XMLDOM")

xmlDoc.Async = False

'path of XML file

XMLDataFile="D:\sample.xml"

'Load the XML File

xmlDoc.Load(XMLDataFile)

'Give the XML path of XML tag you want to access and /text() to access the value of tag.

'XML path follows the hierarchy

'XML path for author tag will start from catalog->book->author

'so the XML path will be /catalog/book/author

'Text() will gie the value of the tag

Set nodes = xmlDoc.SelectNodes("/catalog/book/author/text()")

Msgbox " total count of Author tag is " & nodes.length

For i = 0 To (nodes.Length - 1)

Author= nodes(i).NodeValue

Msgbox "The name of Author is " & Author

Next

'To get the description tag value in the XML file

Set nodes = xmlDoc.SelectNodes("/catalog/book/description/text()")

Msgbox " total count of Description tag is " & nodes.length

For i = 0 To (nodes.Length - 1)

Descripton= nodes(i).NodeValue

Msgbox "The Description is < " & Descripton & " > "

Next

Set xmlDoc=nothing