1. **What is the difference between vbs and qfl files?**

* **vbs** is microsoft's vbscript file while **qfl** is quick test function library file.
* **vbs** is executable file. Just double click **.vbs** file in windows system and code will execute.
* While **Qfl** is not executable file. To Use qfl file in qtp, you must associate qfl file from test settings while **.vbs** file can be included with just "executefile" statement.

1. **What is the difference between Eval and Execute in QTP?**

|  |  |
| --- | --- |
| **Eval** | **Execute** |
| Evaluates an expression and returns the result.  Eval(expression)  *expression:* String containing any legal VBScript expression. | Executes one or more specified statements.  Execute (statement)  *statement:* The required statement argument is a string expression containing one or more statements for execution. Include multiple statements in the statement argument, using colons or embedded line breaks to separate them. |
| |  |  | | --- | --- | | a="cat" b="dog" var=Eval("a=b") Msgbox var  Returns: False | a="cat" b="cat" var=Eval("a=b") Msgbox var   Returns: True | | |  |  | | --- | --- | | a="cat" b="dog"   Execute("a=b") Msgbox a   Returns: dog | a="cat"  b="dog" c="apple" d="orange" Execute("a=b:c=d") Msgbox a Msgbox c  'Returns dog for a 'Returns orange for c | |

1. **What is the difference between executefile and loadfunctionlibrary?**

**a. ExecuteFile** - It loads the libraray at run time but with no debugging support.

**Loadfunctionlibrary** – It also loads the library at runtime.Here debugging is possible at runtime

**b. ExecuteFile** - we can load only single libraray using call executefile statement for

eg: ”C:\func1.vbs”

**Loadfunctionlibrary** - But here we can load multiple function libraries using call loadfunctionlibrary for eg: “C:\func1.vbs”,”C:\func2.vbs”

1. **What is the difference between invokeapplication and systemutil.run?**

**Systemutil**  
  
An object used to control application and process during run session.  
  
Ex: Systemutil.run "iexplore.exe","www.google.com"  
  
**Invoke Application**  
  
It can open only executable files and is used primarily for backward compatibility.  
  
InvokeApplication "C:\Program Files\Internet Explorer\IExplorer.exe

1. **What are the types of repositories?**

Object Repository is of **two** types.

**Local Object Repository**

As the name suggests, the Object Repository is applicable only for that action.

**Shared Object Repository**

The Object Repository is shared across actions/modules, which would be mapped for two or more actions. Local objects can be exported to be saved into Shared Object Repository by using the option "Export Local Objects" option.

1. **How many types of license are there and what type of license used in your organization in UFT?**

[**QTP License Types**](http://www.qtpsudhakar.com/2012/04/qtp-license-types.html)

To work with QTP we should always require a license. When you install QTP, by default it will provide a 30 day demo license (For QTP 11). In the demo period or completion of demo period, to work with QTP you must give the license information specified below.

**Seat License:**

It’s a permanent license which is specific to the computer on which QTP installed. If you want to have QTP in two machines and you want to use the both machines of QTP then you must buy 2 seat licenses. To configure seat license for QTP you must have a license code that is given by the vendor.

**Concurrent License:**

This is a network based license that can be used by multiple users. If I brought 5 concurrent QTP licenses and I have 10 people to work on, then any 5 people can connect to my license server to work with QTP. That means anybody can use QTP but at a time a maximum of 5 people can work. A concurrent license can have more licenses in it. To configure a concurrent license for your QTP, you must have concurrent license server name or IP address.

**Commuter License:**

A commuter license is also a type of concurrent license that can avail during the unavailability concurrent license server. If you’re going on a business trip and not able connect to the concurrent license server then you can ask administrator to install commuter license. A commuter license is a concurrent license that works like a seat license for a specific period.

If you have configured 5 concurrent licenses and if you want to configure a commuter license for a person’s onsite business trip then from that time onwards you should work with 4 concurrent licenses till the specified configured period of commuter license. The Commuter license validity is up to 180 days.

You can change your license type at any time (as long as you are logged in with administrator permissions on your computer). For example, if you are currently working with a seat license, you can choose to connect to a concurrent license server, if one is available on your network.

1. **How many ways for launching the browsers in UFT?**

Launch Applications Using QTP

1. **SystemUtil.Run**

SystemUtil.Run ( FileName, Path)

* **FileName** – The name of the file you want to run.
* **Path (optional)** – The default directory of the application or file.

**Example Usage:**

SystemUtil.Run "C://Program Files/Internet Explorer/IEXPLORE.EXE",”application path”

1. **InvokeApplication**

This command is now mainly used for the backward compatibility ie to use with the lower versions (below QTP 6.0) of QTP.

InvokeApplication("Full URL as Parameter")

**Example Usage:**

InvokeApplication "C: //Program Files/Internet Explorer/IEXPLORE.EXE http://www.yahoo.com"

## 3. VBscript to invoke application

1. Create a “WScript.shell” object.
2. Use the “run” object to launch the application. If the path to your executable contains spaces, use Chr(34) to ensure the path is contained within double quotes.
3. When done, set the shell object to nothing to release it.

**Example:**

Dim oShellSet oShell = CreateObject ("Wscript.shell")'

'Example 1 - run a batch file:

oShell.run "F://jdk1.3.1/demo/jfc/SwingSet2.bat"

'Example 2 - run a Java jar file:

oShell.run "java -jar F://jdk1.3.1/demo/jfc/SwingSet2/SwingSet2.jar"

'Example 3 - launch Internet Explorer:

oShell.Run Chr(34) & "C://Program Files/Internet Explorer/IEXPLORE.EXE" & Chr(34)

Set oShell = Nothing

## 4. IE Automation Object Model

Set oIE = CreateObject("InternetExplorer.Application")

oIE.Navigate "http://www.google.com/"

oIE.Visible = True

......

......

Set oIE = Nothing

1. **What is other names for local object repository and shared object repository in UFT?**
2. **What is difference between function and action in UFT?**

♦ Action is a collection of Vb statements in[**QTP**](https://www.youtube.com/playlist?list=PLyGqUe6Oa_5HAIz0A-p_H48-cKXwNK0ZA). It does not return any values.Function collection of Vb statements in QTP.  
 It returns single value.

♦We can call functions within actions but we can't call actions within functions

♦Generally functions are saved with ".vbs" extention whereas actions will save with ".mts".

♦ Every Action will have its own Datatable whereas function does not.

♦ Action can have a object repository associated with it while a function can't. A function is just lines of code with   
 some/none parameters and a single return value while an action can have more than one output parameters.

♦ Action can contains Object Repository, Data table, Active screen etc. whereas function do not have these features.

♦ Action is internal to [**QTP**](https://www.youtube.com/playlist?list=PLyGqUe6Oa_5Eh7Irm1JcJwvJWpsDtLU5F)whereas Function is just lines of code with some/none parameters and a single return value.

♦ Action can/cannot be reusable whereas functions are always reusable.

♦ Action Parameter have default values whereas VB script function do not have any default values.

♦ Action parameter type are byvalue only where vbscript functions can be passed byref.

♦ Action can have multiple output(returning) values whereas function can return only single value.

1. **How can you associate shared object repositories during runtime?**

We can add object repository at runtime two ways are there u can add  
  
**1. When u write below syntax in Action1**  
  
Syntax: RepositoriesCollection.Add(Path)

**Ex:** StrPath = "D:\FrameWork\Repository\GoogleHomePage.tsr"  
RepositoriesCollection.Add(StrPath)  
  
if write in Action1 it will automatically add the Object Respository to the Action1  
(i.e Edit Menu-->Action-->Action Properties-->Associate Repository tab) at runtime.  
  
No need to add the object repository before running.  
  
**2. Add the object repository at runtime by using AOM (Automated Object Model).**  
  
**Ex:**   
Dim qtAppn   
Dim qtObjRes   
  
Set qtAppn = CreateObject("QuickTest.Application")   
qtAppn.Launch  
qtAppn.Visible = True   
  
qtApp.Open "E:\Test\Test2", False, False   
Set qtObjRes = qtApp.Test.Actions("Login").ObjectRepositories   
  
qtObjRes.Add "E:\OR\ObjRes.tsr", 1   
  
The above example Add the Object Repository(ObjRes.tsr) to the "Login" action in Test2.  
  
Here also no need to add the object repository in Test2.

1. **How many ways associated the function libraries?**
2. By using **‘File > Settings > Resources > Associate Function Library’** option in QTP.

**Using ‘File > Settings > Resources > Associate Function Library’ option from the Menu bar**

1. By using **Automation Object Model (AOM**).

'Open QTP

Set objQTP = CreateObject("QuickTest.Application")

objQTP.Launch

objQTP.Visible = True

'Open a test and associate a function library to the test

objQTP.Open "C:\Automation\SampleTest", False, False

Set objLib = objQTP.Test.Settings.Resources.Libraries

'If the library is not already associated with the test case, associate it.

If objLib.Find("C:\SampleFunctionLibrary.vbs") = -1 Then ' If library is not already added

  objLib.Add "C:\SampleFunctionLibrary.vbs", 1 ' Associate the library to the test case

End

1. By using **ExecuteFile** method.

'Action begins

ExecuteFile "C:\YourFunctionLibrary.vbs"

1. Using **LoadFunctionLibrary** method.

'Some code from the action

LoadFunctionLibrary "C:\YourFunctionLibrary\_1.vbs" 'Associate a single function library

LoadFunctionLibrary "C:\FuncLib\_1.vbs", "C:\FuncLib\_2.vbs" 'Associate more than 1 function libraries

1. **What is extension for function libraries in UFT?**

QTP can have 3 types of function library extensions:

1) .vbs

2) .qfl

3) .txt

1. **Explain about life cycle of automation testing?**

Various Stages involved in automation life cycle are  
1.Automation Feasibility Analysis  
2.Test Plan  
3.Test Design  
4.Test Environment Set Up  
5.Automation Script Generation  
6.Test Execution  
7.Defect Analysis and fixing.  
8. Cost involved in Automation  
  
**1. Automation Feasibility Analysis**  
Requirements and expectations should be set clear. Things like

1. How and what is to be automated
2. Which all modules can be automated
3. Identify the test cases that cannot be automated due to tool limitations and / or complexity of the functionality
4. Analyse the applicability of a suitable tool to carry out automation
5. Rough estimate of the size, effort and cost involved in this automation
6. Review of feasibility analysis report

Should be decided in the beginning itself to prevent any confusion after the automation process has been flagged off. This stage hence requires great attention.  
  
**2. Test Plan and Design**

1. Analyse the manual steps from automation standpoint
2. Identify frame works
3. Anayise the frame works for thier pros and cons
4. Finalise a fram work
5. categorise scripts based on functionality
6. Identify the functionality to be parameterised
7. prepare the detailed cost, size and effort estimate
8. Review of Test Plan by client

**3. Test Environment Set up**  
Test team may not be involved in this aspect if the development team provides the test environment in which case test team is required to check the readiness of the given environment.

1. Understand the required architecture, Environment -set up
2. prepare an Environment check list
3. Prepare hardware and software requirements
4. Finalise the connectivity requirements
5. Set up Test Environment

**4. Automation Script Generation**  
The project team will review the test plan and obtain clarifications and then decide upon the type of automation to be sued (capture/playback,scripting, etc)

1. Create test scenarios
2. Identify re-usable test scenarios
3. Group common test cases together
4. Finalise the test Scripts

**5. Test Execution**

1. Perform the testing as per the test plan
2. Perform the batch test wherever necessary
3. Update the status in status report
4. Log the defects

**6. Defect Analysis and Fixing**

1. Analyse the severity and priority of the defect
2. Fix the defects
3. Document the defect along with the solution

**7. Cost involved in automation**

Fixed cost:

1. Feasibility analysis cost
2. Tool Selection and Acquisition cost
3. Hiring or training the resources
4. Time in learning the application/business process
5. Proof of concept
6. Test suite documentation effort

Variable cost

1. Script maintanance and documentations cost
2. Automated test infrastructure maintanance cost
3. Execution cost
4. **What is difference between QTP and UFT?**
   1. **HP UFT = HP QuickTest Professional + HP Service Test.**
   2. **Browsers supported by QTP and UFT.**
   3. **OS supported by QTP and UFT.**
   4. **Welcome page (Start Page) of UFT and QTP.**
   5. **QTP and UFT both have two different icons.**
   6. **UFT Installation is faster and secure than QTP.**
   7. **UFT provides all services at one common place (IDE).**
   8. **New Add-Ins – Flex and Qt.**
   9. **New Recording type has introduced INSIGHT Recording.**
   10. **UFT’s improved Debug process using many Features.**
   11. **UFT’s Integrated Print window.**

UFT has integrated print window to print debug information from scripts.

* 1. **UFT’s new GUI, API and BPT testing window.**
  2. **ALM now using external authentication.**

1. **How can you change the variable during debugging time in uft?**

Yes we can change it.

The following are the panes in the debugging window −

Debugging using Break Point

* **Output** − This Tab displays all the Output of the Print Statements.
* **Watch** − This Tab displays the Boolean output of the Given Expression.
* **LocalVariables** − This Tab displays the Output of the Local Variables.

1. **What is difference test object and runtime in UFT?**

**Test Object:**  
An object that resides in Object Repository and which QTP use while running script is called Test Object.  
  
**Run Time Object:**   
An object that actually presents in the application is called Run Time Object.  
  
While running script QTP compares Test Object with Run Time Object to check Object Existence in the application.

1. **What is data parameterization?**

Parameterization is nothing but giving multiple Input/Test data to the test script.

Different ways to Parameterize:  
  
1.  Input the Test data from External files.  
  
2.  Input the Test data through Datatable.  
  
3.  Input Test data through loop statements.  
  
4.  Input Test data dynamically through interactive submission.

1. **What is difference between import and importsheet?**

Import method will imports the specified Excel file to the run time datatable.

Ex: Datatable.Import ("C:\sample.xls")  
  
ImportSheet method will imports a specified sheet of a specified file to a specified sheet in the datatable.

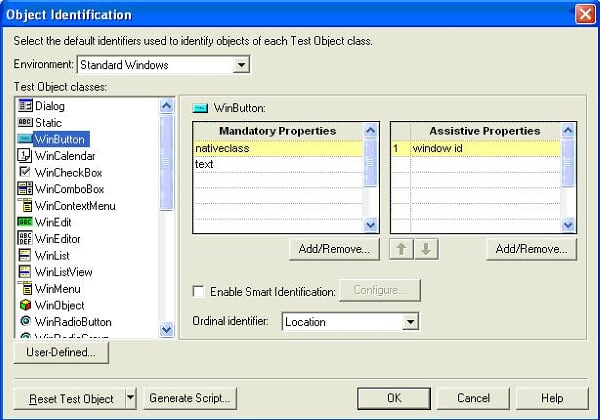
Ex: Datatable.ImportSheet "c:\sample.xls", SourceSheetName(or Number), DestinationSheetName(or Number)  
  
  
If excel file has 7 sheets, if we use import method, then all the 7 Sheets are imported into the runtime datatabe?  
The Answer is No. Only the first 2 sheets will be imported into the runtime datatable.  
Not all the sheets.  
Why because, By default datatable will have 2 sheets (Global/Local)  
If you add an action in the QTP then only, a new sheet will be added in to the datatable.

1. **Explain about object identification mechanism?**

**Object Identification Dialog box**  
  
Navigation: - Tools**-->** Object Identification  
  
It is mainly deals with how QTP is recording objects and properties to object repository.

Using Object Identification dialog box we can

* Configure the properties for each class
* Selecting the Ordinal Identifier
* Configuring smart identification
* Creating user defined classes

In object Identification Dialog box you can find list of classes, for every class you can find pre-configured Mandatory and assistive properties.  
  
**Mandatory properties** are properties that QuickTest always learns for a particular test object class.  
  
**Assistive properties** are properties that QuickTest learns only if the mandatory properties that QuickTest learns for a particular object in your application are not sufficient to create a unique description. If several assistive properties are defined for an object class, then QuickTest learns one assistive property at a time and stops as soon as it creates a unique description for the object. If QuickTest does learn assistive properties, those properties are added to the test object description.  
  
If the combination of all defined mandatory and assistive properties is not sufficient to create a unique test object description, QuickTest also records the value for the selected **ordinal identifier**.  
  
**Ordinal Identifier**  
  
In general, there are two types of ordinal identifiers:  
  
**Index**—indicates the order in which the object appears in the application code relative to other objects with an otherwise identical description.  
  
**Location**—indicates the order in which the object appears within the parent window, frame, or dialog box relative to other objects with an otherwise identical description. Values are assigned from top to bottom, and then left to right.  
  
The **Web Browser object** has a third ordinal identifier type:  
  
**CreationTime**—indicates the order in which the browser was opened relative to other open browsers with an otherwise identical description. Each test object class has a default ordinal identifier selected.  
  
[](https://v9jjsg.bay.livefilestore.com/y1mGhq3YhLPwXaVl_iDHcW54uDYcFKH1B4jtWroIJAwDjHk5hCaroDzbanuL4CR6nPJVh8xMFBXfxDCyz_qlE_KS7Qd-QJGuvbISMs8kSz9SibiuZv6SxVDnrAwBWmgRbkNfNjDa230mH4/im5.JPG)  
  
**Configuring Mandatory properties**  
  
In object identification dialog box select the object class from test object classes list and go to Mandatory properties for that class.  
  
Click on **Add/Remove** button and select or deselect the properties you want to configure.  
  
**Configuring Assistive Properties**  
  
In object identification dialog box select the object class from test object classes list and go to Assistive properties for that class.  
  
Click on **Add/Remove** button and select or deselect the properties you want to configure.  
  
**Configuring Ordinal Identifiers**  
  
To modify the selected ordinal identifier, select the desired type from the **Ordinal identifier**box.

[](https://v9jjsg.bay.livefilestore.com/y1mIymcJg7XLdXvixQQVtiwufjpBigbL0yEN0mhmEFiDERoh-9bye9OxKZkvkTqErIqHGPsM_bRZON7iJgbNLnRABMEqO7UiShvhgQawH6X3npBmTtSyhKJVNdi35vjx8mq63BYzCJJ918/im6.JPG)

**Smart Identification**  
  
When QuickTest uses the recorded description to identify an object, it searches for an object that matches every one of the property values in the description. In most cases, this description is the simplest way to identify the object and unless the main properties of the object change, this method will work.  
  
If QuickTest is unable to find any object that matches the recorded object description, or if it finds more than one object that fits the description, then QuickTest ignores the recorded description, and uses the Smart Identification mechanism to try to identify the object.  
  
The Smart Identification mechanism uses two types of properties:  
  
**Base filter properties**—The most fundamental properties of a particular test object class; those whose values cannot be changed without changing the essence of the original object  
  
**Optional filter properties**—Other properties that can help identify objects of a particular class as they are unlikely to change on a regular

1. **What is use of object spy?**

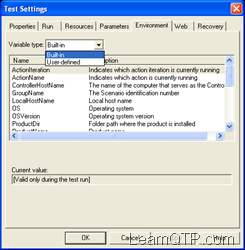
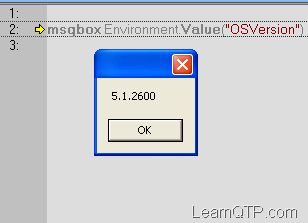
Object Spy is used for viewing the Properties and Methods of any object in an application, the Object spy pointer is used to point an object in an application.

In QTP, Object Spy has two tabs –

1. **Properties Tab –** In this tab, the selected object’s the hierarchy tree and its properties and values are displayed.
2. **Methods Tab –** In this tab, the selected object’s run-time object methods and test object methods are displayed.
3. **Explain about environment variable’s in uft?**

Environment variables in QTP are like global variables in other programming languages which can be accessed through any part of the script. The values of these variables remains same irrespective of the number of iterations (unless you change them through scripting). These variables can prove to be very useful when you want a variable to be shared across various reusable actions.

There are two types of environment variables:

1. **Built-In:** These are the internal variables that are provided by QTP. Among others they can provide you valuable information like the path of the folder where test is located, the path of the results folder, the name of the action iteration or the OS version. So, how can we access the built in environment variable? It’s simple, just have a look at screenshot. So if you want to know the OSVersion of the operating system where your test is running. You can simply type in *Environment.Value(“OSVersion”)*
2. **User-Defined:** These can be further defined into two types.
   * User defined Internal
     + These are the variables that we define within the test.
     + These variables are saved with the test and are accessible only within the test in which they were defined.
     + So how can we define and use them?  
       To define them: *Environment.Value(“name”)= “Ankur Jain”*To call them: *msgbox Environment.Value(“name”)*
   * User defined External
     + These are the variables that we predefine in the active external environment variables file.
     + These can be created using a list of variable-value pairs in an external file in **.xml** format. This is a topic of a separate post that we will discuss later.
3. **What is random number?**

Randomize will make it to return a different number after every run ....

Randomize  
Environment.Value("Client\_No\_Random")=Int((maxvalue - minvalue + 1) \* Rnd) + minvalue

1. **How can you find parameter exists or not in data table?**

The algorithm goes here:  
  
Step I : Use the On Error Resume Next statement to disable any error popups.  
Step II : Use the DataTable's GetSheet method to return the specified sheet. Then add the GetParameter method to seach for the parameter. The following syntax should be used:  
DataTable.GetSheet("Your Sheet").GetParameter(" Your Parameter")  
Step III : Check for an error number using Err.Number. If case of no error, Err.Number should return 0.  
  
With the above algorithm, our QTP Script goes here:  
  
'Disable Error Reporting in QTP  
On Error Resume Next  
  
'Check for the existence of "Global" sheet inside QTP. Then search for "HPQTP" parameter inside "Global" sheet  
DataTable.GetSheet("Global").GetParameter("HPQTP")  
  
If Err.Number<>0 Then  
Msgbox "The specified parameter does not exists"  
Else  
Msgbox "The specified parameter exists"  
End If  
  
'Enable Error Reporting in QTP  
On Error Goto 0

1. **What is difference between descriptive programming and shared object repository?**
2. While using object repository you can come across maintenance and recognition issues and one solution to this is descriptive programming and hence we can ignore object repository.
3. We also choose descriptive programming over object repository when objects are dynamic and we need to rely on special handling for identifying the same.
4. When the object repository size increases and the performance of QTP is decreased when an object is recognized at the same time.
5. Maintenance is easy using descriptive programming than object repository.
6. In QTP while we are using descriptive programming, action can be performed on an object even if the object description is not present in object repository. But it is not the case with the object repository.
7. Even if the application has not been developed we are able to create automation script using descriptive programming.

Object Repository alone cannot handle many special circumstances which can be done alone by descriptive programming. So descriptive programming can be considered as an alternative for object repository but cannot be treated as a replacement for object repository.

1. **What type of repositories you can merge and while merging what are conflicts u can get?**

We can merge only .tsr extension files means shared repositories only.

Merging two object repositories can result in conflicts arising due to similarities between the objects they contain. There are three types of conflicts:

1) **Similar Description Conflict:** Two objects which have the same name and the same object hierarchy, but which have slightly different descriptions. In this conflict type, one of the objects always has a subset of the properties set of the other object.

2) **Same Name Different Description Conflict:** Two objects which have the same name and the same object hierarchy, but differ somehow in their description (for example, they have different properties, or the same property with different values).

3) **Same Description Different Name Conflict:** Two objects, which have identical descriptions, have the same object hierarchy, but differ in their object names. By default, the conflict resolution settings for conflicts of this type are configured so that the target object repository takes the object name from the primary source file.

1. **By default which repository is there in qtp/uft?**

By default local repository is there in qtp/uft

1. **How many ways you can create shared object repository?**

* All repositories are local by default.  To create a Shared Object Repository, in the Object Repository Dialog Box  , Click File > Export Local Objects.

Repository files have an extension .tsr .Give a suitable name say "guru" and save

The Shared Repository File is now created

* By directly from Shared Object repository.

1. **What is difference between export to local objects and object repository manager?**

Local Object Repository should be used when:  
  
You are creating single-action tests. You are creating simple tests, especially under the following conditions:  
  
1.You have only one, or very few, tests that correspond to a given application, interface, or set of objects.  
2. You do not expect to frequently modify object properties.  
3. You are new to using QuickTest. You can record and run tests without creating, choosing, or modifying shared object repositories because all objects are automatically saved in a local object repository that can be accessed by its corresponding action.  
  
Shared Object Repository should be used when:  
  
1.You are creating tests using keyword-driven methodologies (not by recording).  
2. You have several tests that test elements of the same application, interface, or set of objects.  
3. You often work with multi-action tests and regularly use the Insert Copy of Action and Insert Call to Action options.  
4. You expect the object properties in your application to change from time to time and/or you regularly need to update or modify object properties.  
5. If you are familiar with testing, it is probably most efficient to save objects in a shared object repository. In this way, you can use the same shared object repository for multiple actions—if the actions include the same objects.  
6. Object information that applies to many actions is kept in one central location. When the objects in your application change, you can update them in one location for all the actions that use this shared object repository.

1. **What is the default object synchronization timeout in QTP?**

The Default object Synchronizaation timeout is 20 Sec

1. **What do you mean by checkpoints in QTP?**

A checkpoint is a verification point that compares the current value with the expected value for specified properties of an Object. If they current and expected value match it generates a PASS status otherwise FAIL status, which can be inserted at any point of time in the script.

1. **What is Environment Variable in QTP and why to use it?**

Environment variables in QTP are similar to global variables which can be accessed through any part of the script. These variables can prove to be very useful when we want a variable to be shared across various reusable actions and functions and recovery scenarios.

**Types of QTP environment variables**

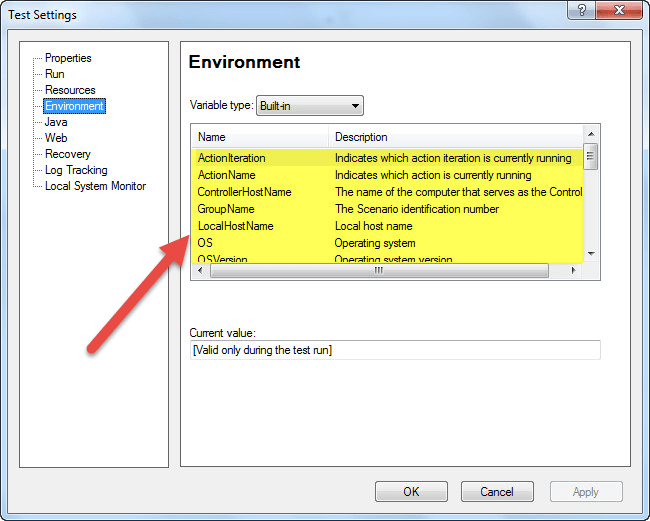
Environment variables are those variables that can be used globally in the tests. There are two types of environment variables

* **Built-in variables**
* **User defined variables (Has 2 sub-types )**
  + **Internal**
  + **External**

**Built-in Variables**

In QTP, built-in variables are pre-defined variables. It enables the user to retrieve the information about the test that is executing and to get information about the O.S (Operating Systems) on which the test is executing. Some of the built-in variables are Action iteration, Operating system, Test directory, local host names, Operating system version etc.

Go to file **->** Settings **->** Environment tab to view the environment variable section to see the list of environmental variables.

[](http://cdn.guru99.com/images/6-2015/052615_0541_Environment1.png)

The values from environment variables can be obtained and used wherever necessary during the run session.

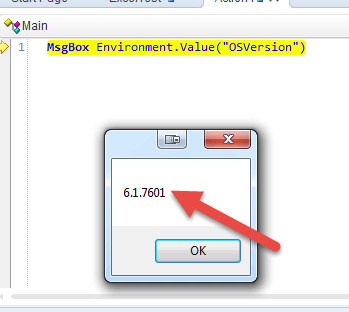
**Example:**

Msgbox Environment("OS")

Msgbox Environment("OSVersion")

Msgbox Environment.value("ProductDir")

Msgbox Environment.value("ProductName")

[](http://cdn.guru99.com/images/6-2015/052615_0541_Environment2.png)

**User-defined Variables**

Before the execution of the test, these are the variables that are defined by the user. It can be used globally across different tests, or they can also be restricted to one test.

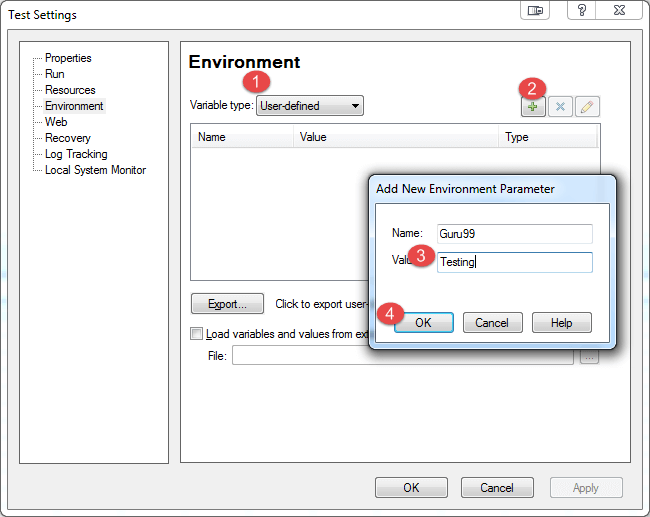
User-defined variables were classified into two types

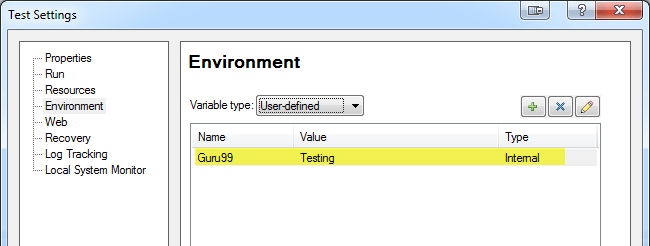
* Internal
* External

1. **User defined - Internal Variables**: These variables are defined by the user before executing the test and these are available only to a particular test
2. **User defined- External Variables:**These variables are defined by the user and it can be used globally across different tests.

External variables can be loaded in two ways. It can be done manually before executing the test through environment tab or user-defined screen by inspecting the load variables check box and then by selecting the XML file.

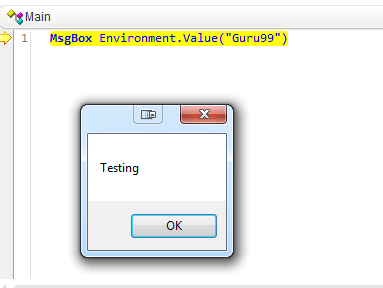
You can create a user defined variable as follows

[](http://cdn.guru99.com/images/6-2015/052615_0541_Environment3.png)

[](http://cdn.guru99.com/images/6-2015/052615_0541_Environment4.png)

You can access the variable as follows

**MsgBox** Environment.Value("Guru99")

[](http://cdn.guru99.com/images/6-2015/052615_0541_Environment5.png)

**External User defined variable**  
  
User defined External: are the variables that we predefine in the active external environment variables file.These can be created using a list of variable-value pairs in an external file in .xml format.  
  
Steps to follow:

* 1. **Create External user defined variable**  
       
     File --> settings --> Environment --> select variable type as user defined --> click add icon (+) --> enter variable name & Value --> click OK --> click export --> browse path and enter file name --> save with xml extension --> click OK.  
       
     **2. Associate environment variable file(xml)**  
       
     File --> settings -->Environment --> select variable type as user defined --> check “load variables and values from the external file” --> browse path of the xml file --> click apply --> Click OK.  
       
     XML file with variable name and value should be mentioned

1. How to select specific webcheckbox in QTP?

We can solve this problem by 3 ways as mentioned below in QTP.

* Using index property
* Using Description object
* Using HTML DOM

**Selecting the web check box using index -**  
if Browser ("myb").page("myp").webcheckbox("index:=2").getROProperty("checked") = false Then  
     Browser("myb").page("myp").webcheckbox("index:=2").click

End if

'Above code will select the third checkbox on the webpage.

**Selecting the web check box using Description Programming -**

**Set** DescObject = **Description**.create()  
DescObject("micclass").value = "webcheckbox"  
**set** col = Browser("myb").page("myp").childobjects(DescObject)  
col(2).click

**Selecting the web check box using HTML DOM -**

**Set** col = Browser("myb").Page("myp").object.getElementsByTagName("input")

'Please note that above statement will give you all the elements on the webpage having input tag.

You can check the innerhtml property of each element to see if the given input element's type is a checkbox or not.

**for** i=0 **to** i<col.count  
    x = col(i).getAttribute("type")  
    **if** **lcase**(**trim**(x)) = "checkbox" **then**  
        col(i).checked = **true**  
    **end** **if**  
**next**

1. **How can you close all browsers in qtp?**

•    Method 1  
**While** Browser("CreationTime:=0").Exist  
    Browser("CreationTime:=0").Close  
**Wend**  
  
•    Method 2  
  
*'Call Function*  
**Call** CloseAllBrowsers("IE")  
  
*'\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**  
*'SystemUtil.Closeprocessbyname closes specific process by name*  
  
**Function** CloseAllBrowsers(oBrName)  
 *' Close the specific/all browsers*  
 **Select** **Case** **ucase**(oBrName)  
  
  **Case** "IE"  
  *'Close IE*  
   SystemUtil.CloseProcessByName "iexplore.exe"  
  **Case** "FF"  
 *'Close FF*  
   SystemUtil.CloseProcessByName "firefox.exe"  
      **Case** "CHROME"  
  *'Close CHROME*  
   SystemUtil.CloseProcessByName "chrome.exe"  
  **Case** "ALL"  
  *'Close All Browsers*  
   SystemUtil.CloseProcessByName "iexplore.exe"  
   SystemUtil.CloseProcessByName "firefox.exe"  
   SystemUtil.CloseProcessByName "chrome.exe"  
  
 **End** **Select**  
  
**End** **Function**  
  
•    Method 3   
**Function** CloseAllBrowsers()  
   *'Declare Variables*  
 **Dim** oBrDes  
 **Dim** oBrObjList  
 **Dim** objIndex  
  
 *'Create Description Object with Browser class*  
 **Set** oBrDes=**Description**.Create  
 oBrDes.Add "micclass","Browser"  
  
 *'Get Browser Objects from Desktop*  
 **Set** oBrObjList=Desktop.ChildObjects(oBrDes)  
  
 *'Use For Loop to close each browser*  
 *'Use Count-1 because Object Indexing starts from "0"*  
 **For** objIndex=0 **to** oBrObjList.count-1  
  *'Close the Browser*  
  oBrObjList(objIndex).close  
 **Next**  
  
 *'Release Variables*  
 **Set** oBrObjList=**Nothing**  
 **Set** oBrDes=**Nothing**  
**End** **Function**  
  
•    Method 3 **for** close all except QC ALM  
  
**Function** CloseAllBrowsers()  
   *'Declare Variables*  
 **Dim** oBrDes  
 **Dim** oBrObjList  
 **Dim** objIndex  
  
 *'Create Description Object with Browser class*  
 **Set** oBrDes=**Description**.Create  
 oBrDes.Add "micclass","Browser"  
  
 *'Get Browser Objects from Desktop*  
 **Set** oBrObjList=Desktop.ChildObjects(oBrDes)  
  
 *'Use For Loop to close each browser*  
 *'Use Count-1 because Object Indexing starts from "0"*  
 **For** objIndex=0 **to** oBrObjList.count-1  
  **If** **lcase**(oBrObjList(objIndex).GetROproperty("name"))<>"mercury quality center" **then**  
   *'Close the Browser*  
            oBrObjList(objIndex).close  
   **Exit** **For**  
  **End** **If**  
 **Next**  
  
 *'Release Variables*  
 **Set** oBrObjList=**Nothing**  
 **Set** oBrDes=**Nothing**  
**End** **Function**

1. **How can you handle unexpected errors in qtp?**

There are various ways of handling errors in QTP. There are three possible types of errors, one would encounter, while working with QTP. They are −

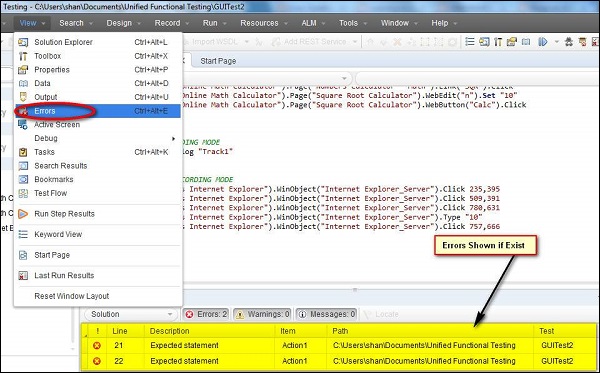
* Syntax Errors
* Logical Errors
* Run Time Errors

Error Types

**Syntax Errors**

Syntax errors are the typos or a piece of the code that does not confirm with the VBscripting language grammar. Syntax errors occur at the time of compilation of code and cannot be executed until the errors are fixed.

To verify the syntax, use the keyboard shortcut Ctrl+F7 and the result is displayed as shown below. If the window is not displayed one can navigate to "View" → "Errors".



### **Logical Errors**

If the script is syntactically correct but it produces unexpected results, then it is known as a Logical error. Logical error usually does not interrupt the execution but produces incorrect results. Logical errors could occur due to variety of reasons, viz- wrong assumptions or misunderstandings of the requirement and sometimes incorrect program logics (using do-while instead of do-Until) or Infinite Loops.

One of the ways to detect a logical error is to perform peer reviews and also verify the QTP output file/result file to ensure that the tool has performed the way it was supposed to do.

### **RunTime Errors**

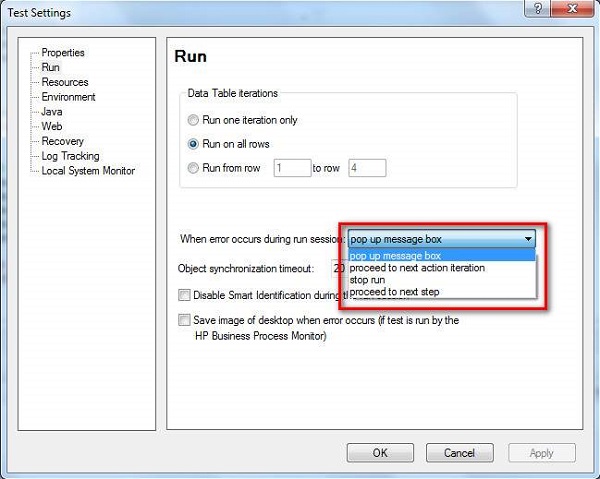
As the name states, this kind of error happens during Run Time. The reason for such kind of errors is that the script trying to perform something is unable to do so and the script usually stops, as it is unable to continue with the execution. Classic examples for Run Time Errors are −

* File NOT found but the script trying to read the file
* Object NOT found but the script is trying to act on that particular object
* Dividing a number by Zero
* Array Index out of bounds while accessing array elements

## Handling Run-Time Errors

There are various ways to handle errors in the code.

**1. Using Test Settings** − Error handling can be defined the Test Settings by Navigating to "File" >> "Settings" >> "Run" Tab as shown below. We can select any of the specified settings and click "OK".



**2. Using On Error Statement** − The ‘On Error’ statement is used to notify the VBScript engine of intentions to handle the run-time errors by a tester, rather than allowing the VBScript engine to display error messages that are not user-friendly.

* **On Error Resume Next** − On Error Resume Next informs the VBScript engine to process executing the next line of code when an error is encountered.
* **On error Goto 0** − This helps the testers to turn off the error handling.

**3. Using Err Object** − Error object is an in-built object within VBScript that captures the run-time error number and error description with which we are able to debug the code easily.

* **Err.Number** − The Number property returns or sets a numeric value specifying an error. If Err.Number value is 0 then No error has occurred.
* **Err.Description** − The Description property returns or sets a brief description about an error.
* **Err.Clear** − The Clear method resets the Err object and clears all the previous values associated with it.

### **Example**

'Call the function to Add two Numbers Call Addition(num1,num2)

Function Addition(a,b)

On error resume next

If NOT IsNumeric(a) or IsNumeric(b) Then

Print "Error number is " & err.number & " and description is :

" & err.description

Err.Clear

Exit Function

End If

Addition = a+b

'disables error handling

On Error Goto 0

End function

**4. Using Exit Statement** − Exit Statements can be used along with Err object to exit from a test or action or iteration based on the Err.Number value. Let us see each one of those Exit statements in detail.

* **ExitTest** − Exits from the entire QTP test, no matter what the run-time iteration settings are.
* **ExitAction** − Exits the current action.
* **ExitActionIteration** − Exits the current iteration of the action.
* **ExitTestIteration** − Exits the current iteration of the QTP test and proceeds to the next iteration.

**5. Recovery Scenarios** − Upon encountering an error, recovery scenarios are triggered based on certain conditions and it is dealt in detail in a separate chapter.

**6. Reporter Object** − Reporter Object helps us to report an event to the run results. It helps us to identify if the concerned action/step is pass/fail.

**Recovery Scenario:**   
  
When executing scripts we may get some Unexpected errors.To recover the test and continue running script from these unexpected errors we use Recovery Scenarios.  
  
  
**What happens in Recovery Scenarios?**  
  
A recovery scenario consists of the following:  
  
**Trigger Event:**  
  
The event that interrupts your run session. For example a window that may pop up on screen.  
  
**Recovery Operations:**  
  
The operations to perform to enable QTP to continue running the test after the trigger event interrupts the run session. For example, clicking an OK button in a pop-up window.  
  
**Post-Recovery Test Run Option:**  
  
The instructions on how QTP should proceed after the recovery operations have been performed, and from which point in the test QTP should continue. For example, you may want to restart a test from the beginning, or skip a step entirely and continue with the next step in the test. Recovery scenarios are saved in recovery scenario files having the extension .qrs. A recovery scenario file is a logical collection of recovery scenarios, grouped according to our specific requirements.  
  
  
Recovery scenario manger can handle following four events:   
  
1. Pop-up window: To handle unwanted pop ups.  
  
2. Object state: To handle object related errors at runtime.  
  
3. Test run error: To handle vb script statement errors at runtime.  
  
4. Application crash: To handle crashed applications at runtime.  
  
  
**1.Recovery Scenario for Popup Window:**  
  
Steps to follow to handle unwanted popups:  
  
**Trigger Event Steps:**  
  
Resources Menu -->Recovery Scenario Manager -->New -->Next -->Select “Popup Window” as Trigger event -->Next -->Click on Hand Icon -->Show unwanted window with Hand icon -->Next -->Next -->(Continue below mentioned steps)  
  
**Recovery Operations steps:**  
  
Select function call as Recovery Operation -->Next {Open Notepad -->Save empty file with .vbs extension} -->Browse the .vbs fie path -->Next -->Uncheck Add another Recovery Operation -->Next -->(Continue below mentioned steps)  
  
**Post-Recovery Test Run Option Steps:**  
  
Select Post-Recovery Test Run Option {Repeat current step and continue, Proceed to Next step, Proceed to Next Action, Proceed to next test iteration, Restart current test run, Stop the Test Run} -->Next -->Enter Scenario Name -->Next -->Select Option --> Finish -->Save the scenario with “.qrs” -->Record required Recovery Operation {Click ok, Click Cancel} take the script into function -->Save the library file -->Click Run  
  
 **2.Recovery Scenario for Object State:**  
  
Steps to follow to check Property values of an object in the application match specified values. User can specify property values for each object in the hierarchy.  
  
**Trigger Event steps**:  
  
Resources Menu --> Recovery Scenario Manager --> New --> Next --> Select “Object state Window” as Trigger event --> Next --> Click on Hand Icon --> Show object with and icon --> Next --> Next-->select object property with value (enabled ,false)-->click next --> (Continue below mentioned steps)  
  
**Recovery Operations steps:**  
  
Select function call as Recovery Operation --> Next {Open Notepad --> Save empty file with .vbs extension} --> Browse the .vbs fie path --> Next --> Uncheck Add another Recovery Operation --> Next -->(Continue below mentioned steps)  
  
**Post-Recovery Test Run Option Steps:**  
  
Select Post-Recovery Test Run Option {Repeat current step and continue, Proceed to Next step, Proceed to Next Action, Proceed to next test iteration, Restart current test run, Stop the Test Run} --> Next--> Enter Scenario Name --> Next --> Select Option -->Finish --> Save the scenario with “.qrs” --> Record required Recovery Operation {Click ok, Click Cancel} take the script into function --> Save the library file --> Click Run  
  
  
**3.Recovery Scenario for Test Run Error:**  
  
Steps to follow to check if test does not run successfully then Test Run Error can be raised.  
  
**Trigger Event steps:**  
  
Resources Menu -->Recovery Scenario Manager -->New -->Next -->Select “Testrunerror Window” as Trigger event -->Next -->select any error o -->Next -->Next -->(Continue below mentioned steps)  
  
**Recovery Operations steps:**  
  
Select function call as Recovery Operation -->Next {Open Notepad -->Save empty file with .vbs extension} -->Browse the .vbs fie path -->Next -->Uncheck Add another Recovery Operation -->Next -->(Continue below mentioned steps)  
 **Post-Recovery Test Run Option Steps:**  
  
Select Post-Recovery Test Run Option {Repeat current step and continue, Proceed to Next step, Proceed to Next Action, Proceed to next test iteration, Restart current test run, Stop the Test Run} -->Next >Enter Scenario Name >Next -->Select Option --> Finish >Save the scenario with “.qrs” -->Record required Recovery Operation {Click ok, Click Cancel} take the script into function -->Save the library file -->Click Run  
  
 **4.Recovery Scenario for Application Crash:**  
  
Steps to follow to check application failure during Test Run.  
  
**Trigger Event steps:**  
  
Resources Menu --> Recovery Scenario Manager--> Click New--> Click Next -->Select Application Crash as Trigger event-->Next -->Select selected executable application-->Next --> (Continue below mentioned steps)  
  
**Recovery Operations and Post-Recovery Test Run Option Steps:**  
  
Select Recovery Operation {Keyboard, Mouse Operation,Close Application Process, function Call, Restart, Microsoft Windows} -->Next -->If you want to check Add another operation else uncheck-->Next -->Next -->Enter Scenario Name -->Next-->Select Option -->Finish -->Close -->Save the scenario with “.qrs”  
  
  
**When to use Error handling(on error resume next)   vs   Recovery Scenarios ?**  
  
If you can predict that a certain event may happen at a specific point in your test or component, it is recommended to handle that event directly within your test or component by adding steps such as If statements or optional steps or "on error resume next", rather than depending on a recovery scenario. Using Recovery Scenarios may result in unusually slow performance of your tests.They are designed to handle a more generic set of unpredictable events which CANNOT be handled programmatically.  
  
Example:  
  
A recovery scenario can handle a printer error by clicking the default button in the Printer Error message box.You cannot handle this error directly in your test, since you cannot know at what point the network will return the printer error. You could try to handle this event in your test by adding an If statement immediately after the step that sent a file to the printer, but if the network takes time to return the printer error, your test may have progressed several steps before the error is displayed. Therefore, for this type of event,only a recovery scenario can handle it.  
  
On error Statements  
  
Following are error statements :  
  
1. On Error Resume Next  
2. On Error Go to 0  
3. err.number  
4. err.description  
  
**On Error Resume Next:**  
  
On Error Resume Next statement enables the Error handling in the code.If there is error in the code "On error Resume Next" ignores it and continue with next line of code.  
  
**On Error Go to 0:**  
  
On error got to 0 statement disables error handling we have previiously enabled it by using On Error resume Next.  
  
err.number and err.description:  
  
Provides the error number and the description of the error  
  
Example:

'Call the function to divide

call division

**'Call the function to divide**

**call division**

**Function division()**

**on error resume next**

**'divide by zero**

**z=40/0**

**' Report the error occured. You can see the error number and description in result summary**

**If Err.number <> 0 then**

**Reporter.ReportEvent micWarning,"Error Occured","Error number is " & err.number & " and description is : " & err.description**

**'disables error handling**

**on error goto 0**

**End if**

**End function**

1. **How to Comment a line in QTP script?**

**Select the Line press Ctrl + M**

1. **If I open three browsers in qtp and how can I find which browser is open first?**

**Using Index Number we can identify which browser is opened first.**

1. **What is Difference between ChildObjects, ChildItem and Getcelldata in QTP?**

**ChildObjects**method is to access total child objects from an object using description object. In other words, it returns the collection of child objects contained within the object.

Syntax:- object.ChildObjects ([**Description**])  
  
Ex: **for** the count of No of links present **in** a page we can use Child objects method  
  
**set** odesc = **Description**.Create()  
 odesc("Class Name").Value = "oLinks"  
**set** oLinks=Browser("title:=.\*").Page("title:=.\*").ChildObjects("odesc")  
**msgbox** oLinks.count

**ChildItem** method is to access child objects from a web table cell in web table object without using description object. In other words, it returns a test object from the cell by type and index.

Syntax:-object.ChildItem (Row, Column, MicClass, Index)  
  
Ex:  
**set** a = Browser("A").Page("B").WebTable("C").WebTable("Item").ChildItem(2,2,"WebElement",0)  
    b = a.GetROProperty("value")  
**MsgBox** b

**Getcelldata** method is to retrieve data from a web table cell in web table object.

Syntax:-object.GetCellData (Row, Column)  
  
Ex: **Dim** i,j  
**For** i=1 **to** Row count  
  **For** j=1 **to** column count  
  
     **msgbox** browser("Yahoo! Mail: The best").Page("Yahoo! Mail: The best").WebTable("Yahoo! ID:").GetCellData (i,j)  
  **Next**  
**Next**

1. **What is difference between wait and wait property?**

* **Wait Statement:** It will wait for the specified amount of time. We can specify the time in both seconds and milliseconds. But its necessary to define time in the wait statement even it is zero otherwise it will give you an error.  
    
  Example: wait, wait () both will give errors  
  while  wait(0), wait 0, wait 0, 10000 are valid statements.  
    
  also wait (0, 10000) will give an error and wait 0,10000 is a valid statement.  
    
  The disadvantage of wait statements is that it is not applied on a condition. If this statements is encountered by the compiler then it will wait for the specified amount of time.
* **Exist Property:** A very useful property when dealing with the dynamic objects which takes time to download on the web pages.As the name suggests this property checks if a objects exist before doing operations on the object. This property can be used almost all the objects. It is used as follows:  
    
  A=Browser("").Page("").Frame("").webedit("Login").exist(09)  
    
  The above exist statement will check if the "Login" object exists on the WebPage or not. If it exist then it return True to "A: and if not then it will return False to "A". In real time we generally checks if some objects exist or not before doing some operation on it.  
    
  Example:  
    
  **If** Browser("JobCenter").Page("JobCenter").WebButton(strKeywordName).Exist(10) **then**       
      Browser("JobCenter").Page("JobCenter").WebButton(strKeywordName).Click       
  **end** **if**

Difference between wait and Exist Statement:  
  
1. Execution will wait specifically for the  amount specified in Wait statement. If wait(10) is define then the QTP will wait for 10 seconds. No matter what happens. So if we have a script which run 100 time then naturally it is going to increase the run time significantly.  
Exist: Execution will move to next statement as soon as Exist return a True. So if Exist(10) is specified and the condition become true on the very first second then QTP will move to next statement. Thus we have directly saved the 9 second.  
  
2. Wait is a statement while Exist is a property. By then help of wait we pause the execution for certain amount of time while Exist checks if a specified objects exists or not for the specified time.

* **WaitProperty** – method is used to instruct QTP to wait the execution process until it matches with the object property value based on the specified time.  
    
  E.g.

Browser ("Welcome: Mercury Tours").WaitProperty "name" "Welcome: Mercury Tours" 5000  
Here name is - property name , Welcome: Mercury Tours is - property value, 5000 - number of milli seconds to wait

* **Sync:** Waits for the browser to complete the current navigation.

Syntax : object.Sync   
 Advantage : Moves to next screen or field if the expected value is observed without waiting"

1. **What is the use of getROProperty?**

* **GetTOProperty**   
  The **GetTOProperty** returns the value of the property from the test object's description, i.e., the value used by QTP to identify the object. It returns the value of a property for a test object which QTP recorded to identify during the recording (run time).   
  The GetTOProperty command will retrieve the value as it was originally recorded (or created via DP). It does not matter if the corresponding runtime object exists, or if that value was updated in "the real world" since the object was recorded.  
  Examples:

Example 1(when object is in Object Repository):   
  
SystemUtil.Run "iexplore.exe"  
Browser("Google").Navigate "http://www.yahoomail.com/"  
abc = Browser("Google").Page("Yahoo! Mail: The best").WebEdit("login").GetTOProperties ("name")  
msgbox abc

Example 2 (with descriptive programming):   
Set obj = Browser("name:=Sign in to Yahoo!").Page("title:=Sign in to Yahoo!").WebEdit("html id:=username")   
msgbox obj.GetTOProperty("html id")

* **SetTOProperty**  
  The **SetTOProperty** changes the value of a test object property. Changing the property doesn't affect the OR or Active Screen, but just the way QTP identifies the object during runtime. Actually, this changes the properties of the temporary copy of the object stored in RAM by QTP.  
  Any changes you make using the **SetTOProperty** method apply only during the course of the run session, and do not affect the values stored in the test object repository.

Example:   
Set obj = Browser("name:=Sign in to Yahoo!").Page("title:=Sign in to Yahoo!").WebEdit("html id:=username")  
msgbox obj.GetTOProperty("html id")  
*'Would retrieve the object html id from the test object description, whether it's in the OR or DP defined  
'Now we set the name property*  
obj.SetTOProperty "name", "loginzzzzz"*'And retrieve it*  
msgbox obj.GetTOProperty("name")   
   
Fig: For “Login” text field, QTP captures value of name property as “login”. 

Call Window("Notepad").SetTOProperty("nativeclass", "incorrect value")

Window("Notepad").Close 'this command will fail

* **GetTOProperties**:  
  GetTOProperties is used to enumerate all the properties of an object used for identification.  
  Example:  
  Set abc = Browser("Google").Page("Yahoo! Mail: The best").WebEdit("login")  
  Set Props = abc.GetTOProperties   
  PropsCount = Props.Count   
  For i = 0 To PropsCount - 1   
  PropName = Props(i).Name *(.Name = Returns the name of property)*  
  PropValue = Props(i).Value *(.value = Returns the value of property)*  
  MsgBox PropName & " = " & PropValue   
  Next
* **GetROProperty:**  
    
  **GetROProperty** allows us to get the current value of a test object property. This means that unlike the GetTOProperty and GetTOProperties commands, GetROProperty requires the test object’s corresponding runtime object to exist.  
  Basically you would use **GetROProperty** to get the value of an object property during runtime, such as the current list/combo item selection, or the text in a WebEdit, or the size of an object (width & height).

Print Browser("X").Page("Y"). WebList ("z").GetROProperty("selected item index")

1. **What type of approach ur following in your framework in UFT?**

**Hybrid Framework**

1. **What is regular expression and where exactly you can use regular expression in your project?**
2. **What are challenges faced in your automation?**
3. **When you go for automation?**

1. Whenever the scope of repeatability is more  
2. Whenever high amount of regression testing is possible  
3. When we are expecting the change request often

1. **What are advantages for automation testing?**

Benefits of Automation Testing

* **Fast:** Runs tests significantly faster than human users.
* **Repeatable:** Testers can test how the website or software reacts after repeated execution of the same operation.
* **Reusable:** Tests can be re-used on different versions of the software.
* **Reliable:** Tests perform precisely the same operation each time they are run thereby eliminating human error.
* **Comprehensive:** Testers can build test suites of tests that covers every feature in software software application.
* **Programmable:** Testers can program sophisticated tests that bring hidden information.

1. **In a excel sheet find "Riyaz" name is there or not?**

**Set** appExcel = **CreateObject**(“Excel.Application”)  
appExcel.visible=**true**  
**Set** objWorkBook = appExcel.Workbooks.Open(filepath)*'opens the sheet*  
  
**Set** objSheet = appExcel.Sheets(“Sheet1”) *'To select particular sheet*  
**With** objSheet.UsedRange     *'select the used range in particular sheet*  
**Set** c = .Find (“**Riyaz**”)     *'data to find*  
**For** **each** c **in** objSheet.UsedRange     *'Loop through the used range*  
     **If** c=” **Riyaz**” **then**     *'compare with the expected data*  
           c.Interior.ColorIndex = 40     *'make the gary color if it finds the data*  
    **End** **If**  
    **Set** c = .FindNext(c) *'next search*  
**next**  
**End** **With**  
objWorkBook.save  
objWorkBook.close  
**set** appExcel=**nothing**

1. **In a excel how can you find the numbers of rows and columns are used?**

**Using oSheet.UsedRange.Rows.Count ‘For Row**

**Using oSheet.UsedRangeColumns.count ‘For Cols**

1. **How can you find number of rows in a datatable?**

|  |  |  |
| --- | --- | --- |
| **Method Name** | **Description** | **Syntax** |
| AddSheet | Adds the specified sheet to the run-time data table | DataTable.AddSheet(SheetName) |
| DeleteSheet | Deletes the specified sheet from the run-time data table | DataTable.DeleteSheet SheetID |
| Export | Exports the Datatable to a new file in the specified location | DataTable.Export(FileName) |
| ExportSheet | Exports a Specific Sheet of the Datatable in run-time | DataTable.ExportSheet(FileName,SheetName) |
| GetCurrentRow | Returns the active row of the run-time data table of global sheet | DataTable.GetCurrentRow |
| GetParameterCount | Returns the number of columns in the run-time data Table of Global Sheet | DataTable.GetParameterCount |
| GetRowCount | Returns the number of rows in the run-time data table of Global Sheet | DataTable.GetRowCount |
| GetSheet | Returns the specified sheet from the run-time data table. | DataTable.GetSheet(SheetID) |
| GetSheetCount | Returns the total number of sheets in the run-time data table. | DataTable.GetSheetCount |
| Import | Imports a specific external Excel file to the run-time data table. | DataTable.Import(FileName) |
| ImportSheet | Imports the specified sheet of the specific excel file to the destination sheet. | DataTable.ImportSheet(FileName, SheetSource, SheetDest) |
| SetCurrentRow | Sets the Focus of the Current row to the Specified Row Number | DataTable.SetCurrentRow(RowNumber) |
| SetNextRow | Sets the focus of the next row in the run-time data table | DataTable.SetNextRow |
| SetPreviousRow | Sets the focus of the previous row in the run-time data Table | DataTable.SetPrevRow |

1. **How to click particular link or radio button in a web table in UFT?**

* Browser("title:=Forms.\*").Page("title:=Forms.\*").WebRadioGroup("name:=radio1", "Index:=0").**Select** "#2"
* count the radiobutton in the appliation and u can select the last radio button

**Set** desc=**description**.Create()  
desc("micclass").value="WebRadioGroup"  
**Set** radio=Browser("...").Page("..").ChildObjects(desc)  
    radiocount=radio.count  
**msgbox** radiocount  
  
**For** i=o **to** radiocount-1  
    itemcount=radio.item(i).GetRoProperty("items count")  
    **msgbox** "item count:" & itemcount  
**Next**  
*'FOR SELECTING THE LAST RADIO BUTTON*  
variable1="#" & itemcount-1

* Click on a radio button in Web Table using child item

radiosNumber = Browser("br").Page("page").WebTable("tbl").ChildItemCount (3,4, "WebRadioGroup")  
**Set** rgWithinCell = Browser("br").Page("page").WebTable("tbl").ChildItem (3, 4, "WebRadioGroup", 0)  
  
rgWithinCell.**Select** "#1"

1. **How can you count number of images are there in a webpage?**

**Function** GetAllSpecificControls(Page, MicClass) **Set** Desc = **Description**.Create()  
    Desc("micclass").Value = MicClass  
    **Set** GetAllSpecificControls = Page.ChildObjects(Desc)  
**End** **Function**  
  
**Function** GetAllEdits(Page)  
    **Set** GetAllEdits = GetAllSpecificControls(Page, "WebEdit")  
**End** **Function**  
  
**Function** GetAllButtons(Page)  
     **Set** GetAllButtons = GetAllSpecificControls(Page, "WebButton")  
**End** **Function**  
  
**Function** GetAllLinks(Page)  
    **Set** GetAllLinks = GetAllSpecificControls(Page, "Link")  
**End** **Function**  
  
**Function** GetAllImages(Page)  
    **Set** GetAllImages = GetAllSpecificControls(Page, "Image")  
**End** **Function**  
  
**Set** oPage = Browser("Browser Name").Page("Page Name")  
**MsgBox** "Number of Edits: " & GetAllEdits(oPage).Count  
**MsgBox** "Number of Buttons: " & GetAllButtons(oPage).Count  
**MsgBox** "Number of Links: " & GetAllLinks(oPage).Count  
**MsgBox** "Number of Images: " & GetAllImages(oPage).Count

1. **What are difference between normal run mode and fast run mode in UFT?**

Normal mode :   
step by step execution is done & Pointer(Cursor) is shown befor each statement.execution time taken for running a script will be more than fast mode.  
  
Fast Mode:   
General execution is done i.e no step by step execution...execution time will be less for running the scripts when compared to the Normal mode.  
  
We can set this option from below navigation in qtp :  
  
Tools Menu --> Options --> Run Tab --> fast or Normal  
(Radio Buttons we have to select)