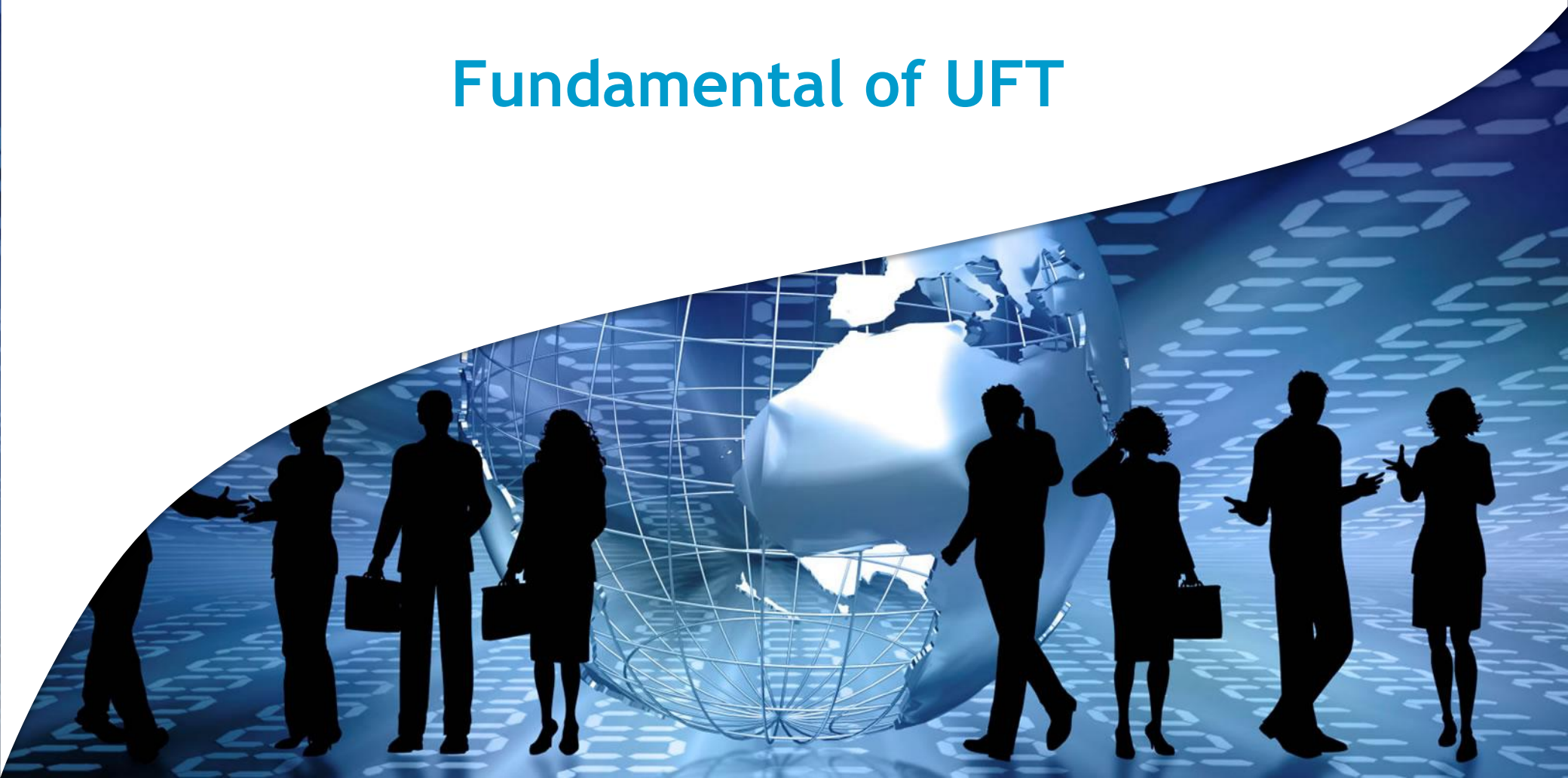
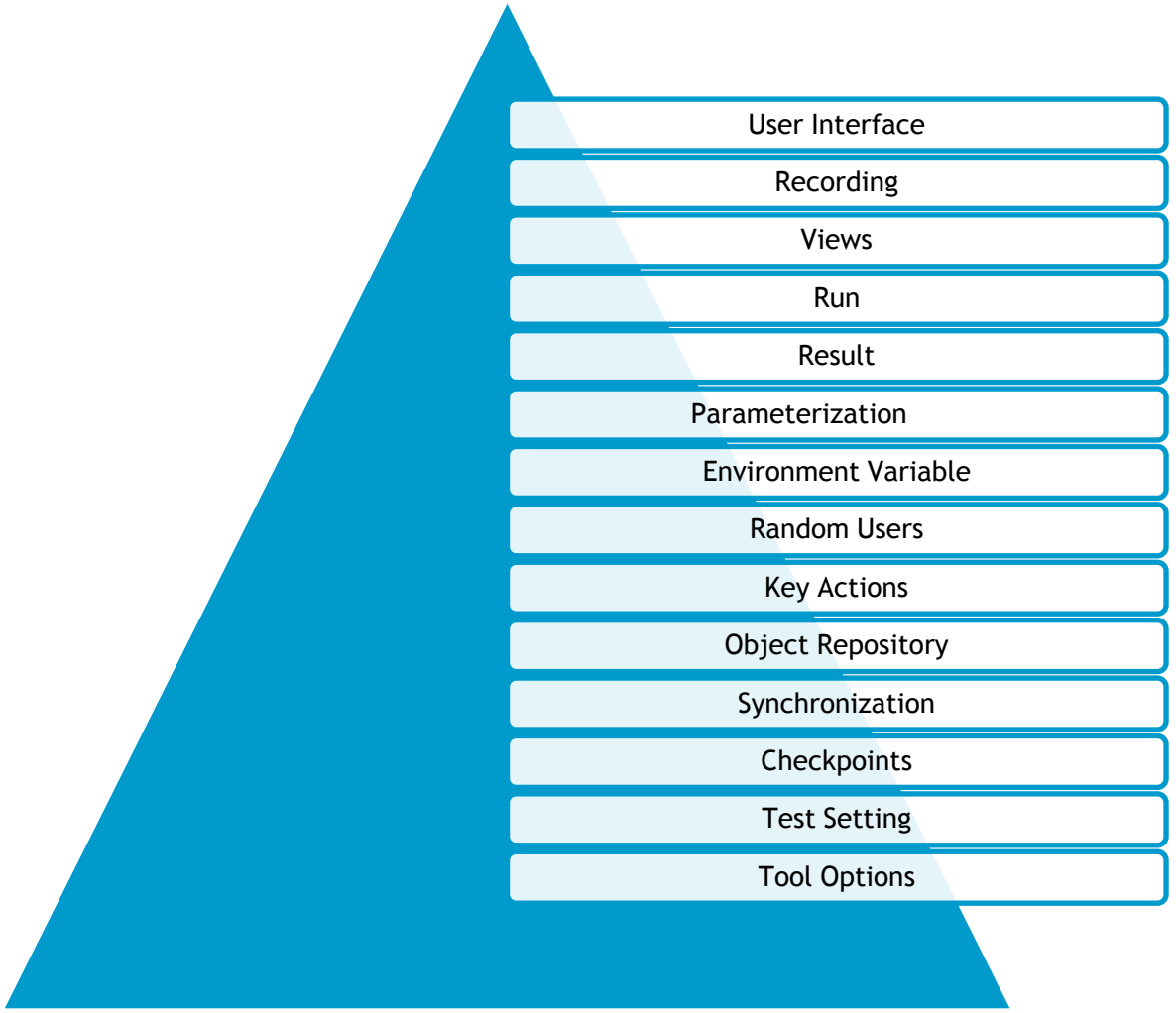
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

SOGETI

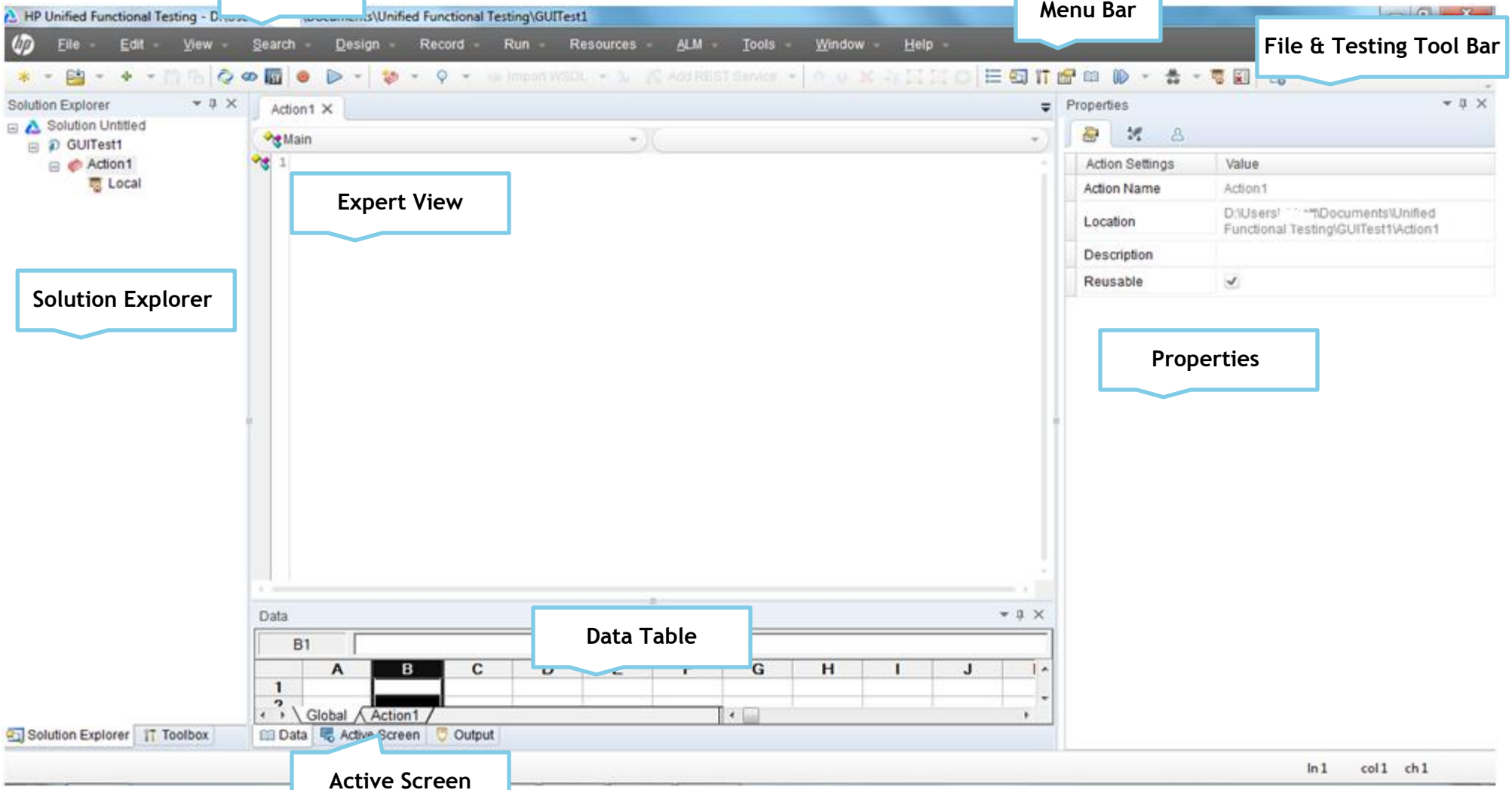
Fundamental of UFT



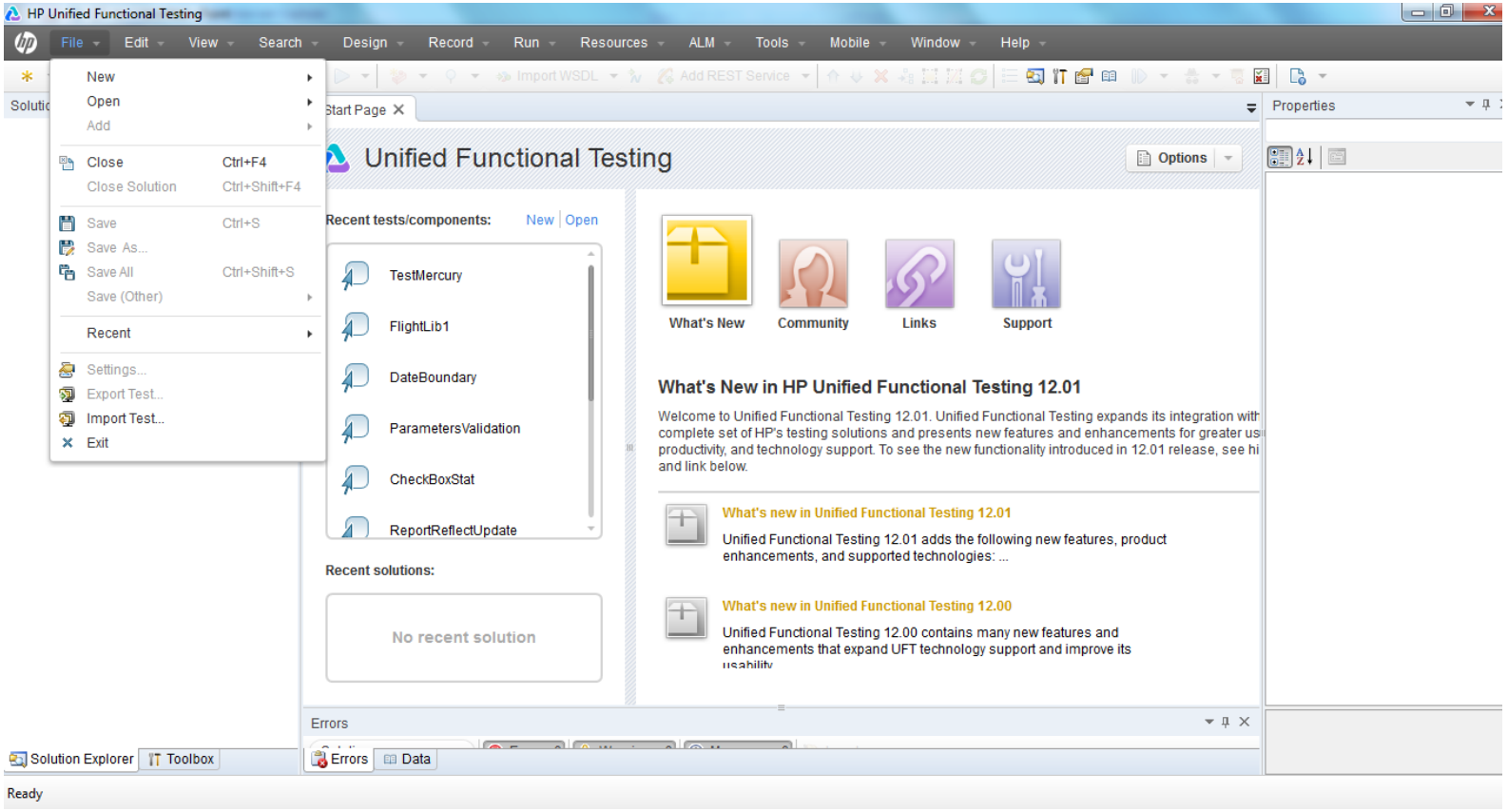
Index



User Interface of UFT

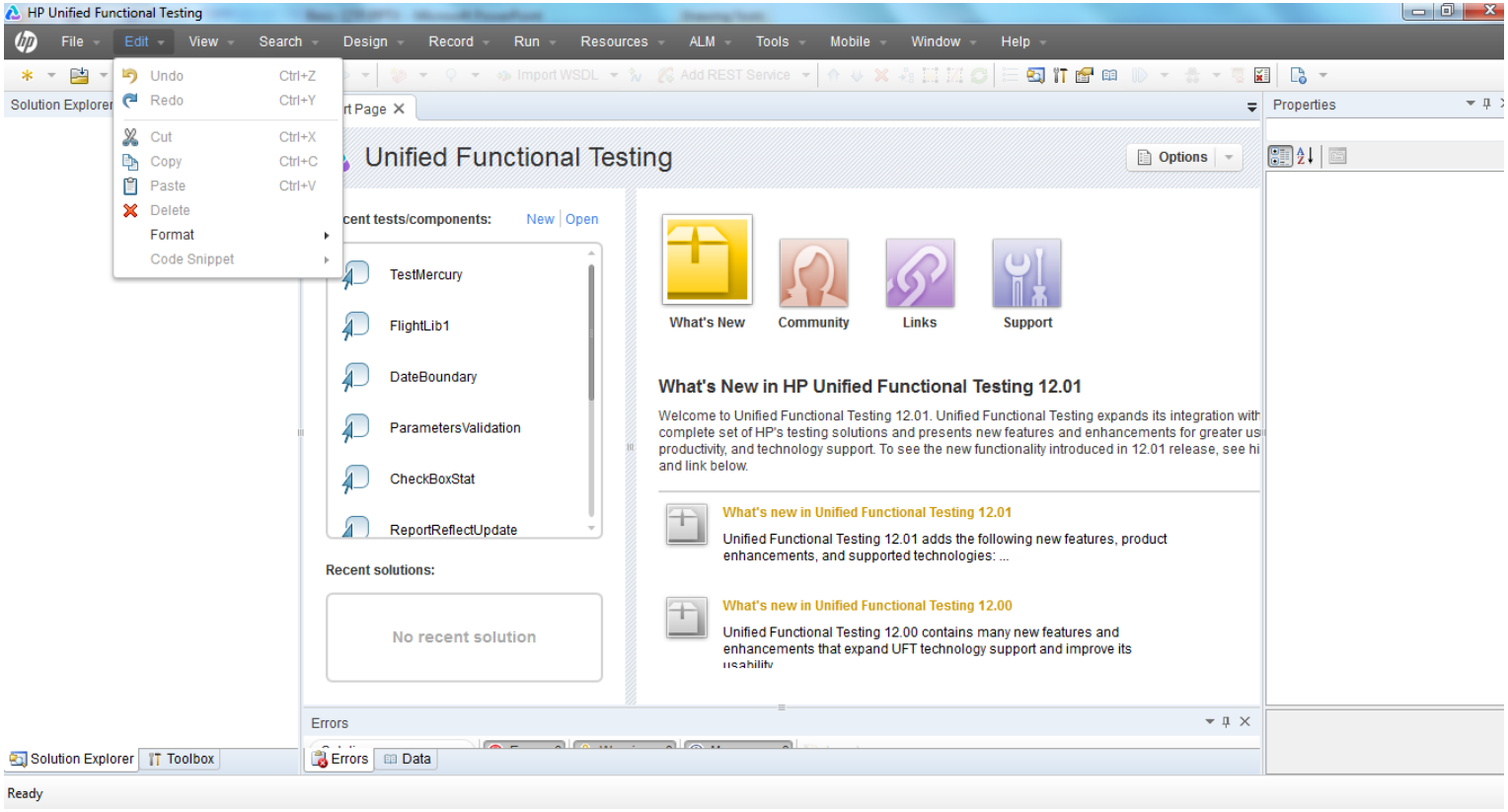


User Interface of UFT : File Menu



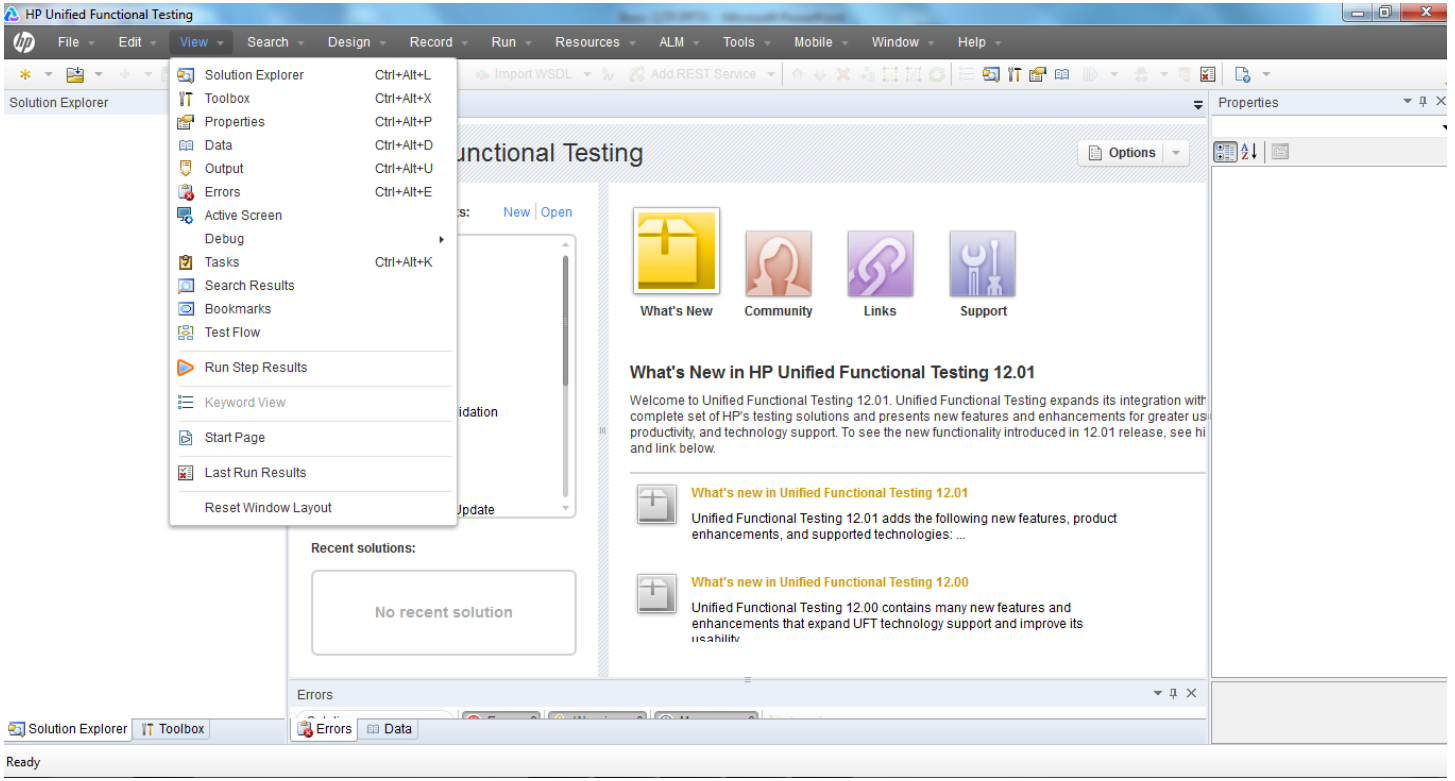
The File menu help user to open new/existing test script.

User Interface of UFT : Edit Menu



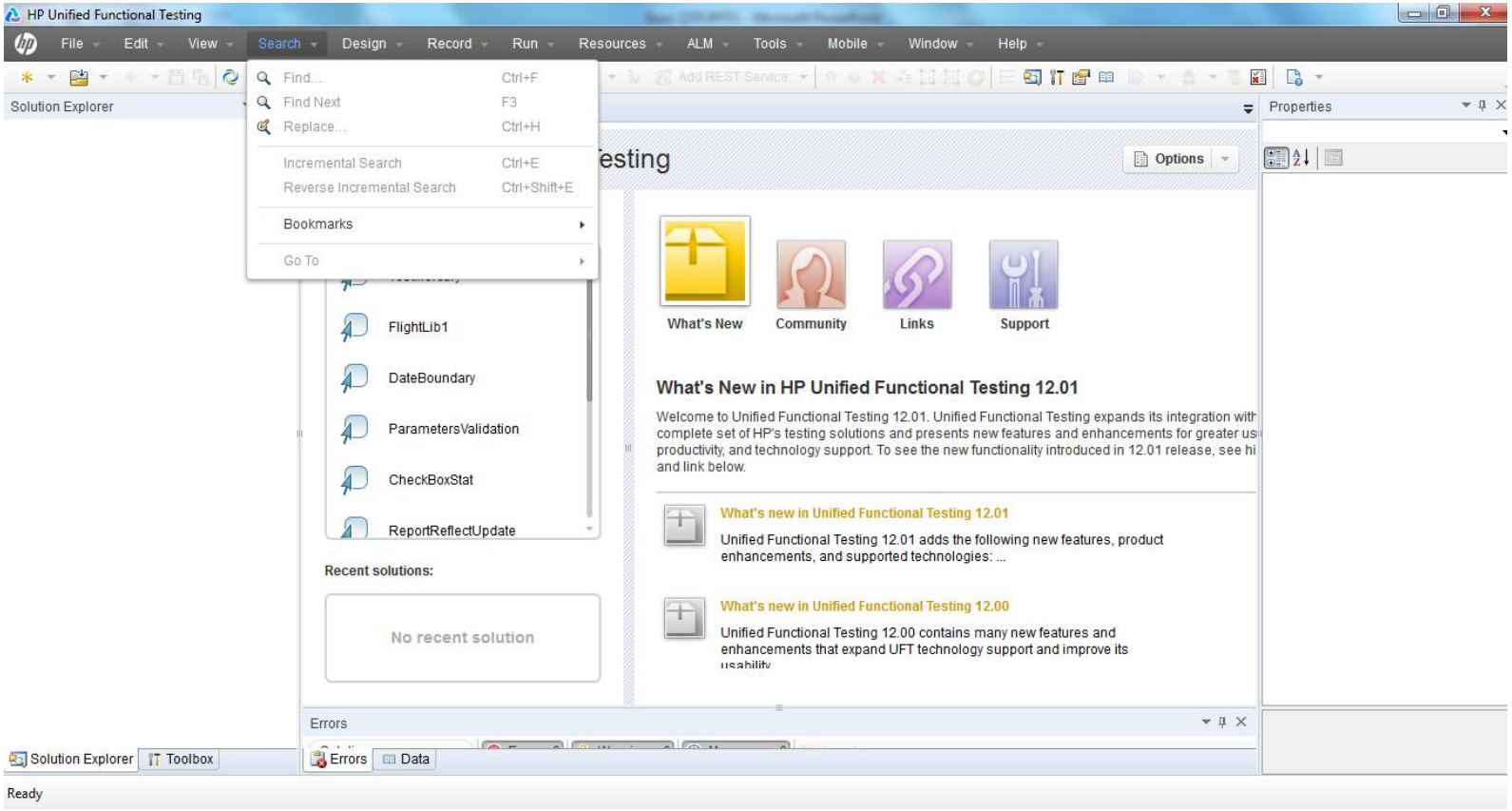
Using edit menu user can do Cut/Copy/Past/Delete/Undo/Redo operations on test script.

User Interface of UFT : View Menu



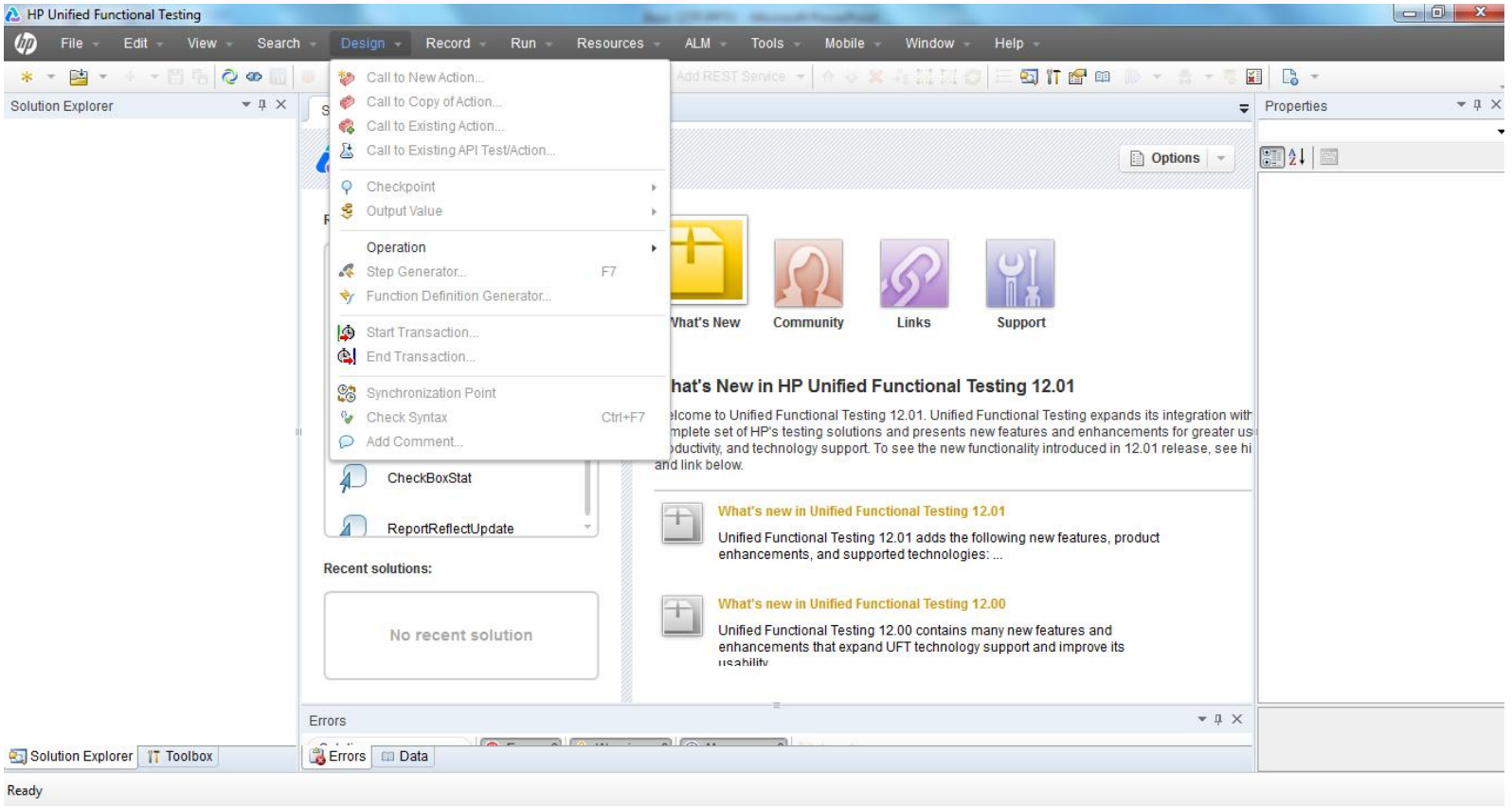
Using View menu user can view different components of test script.

User Interface of UFT : Search Menu



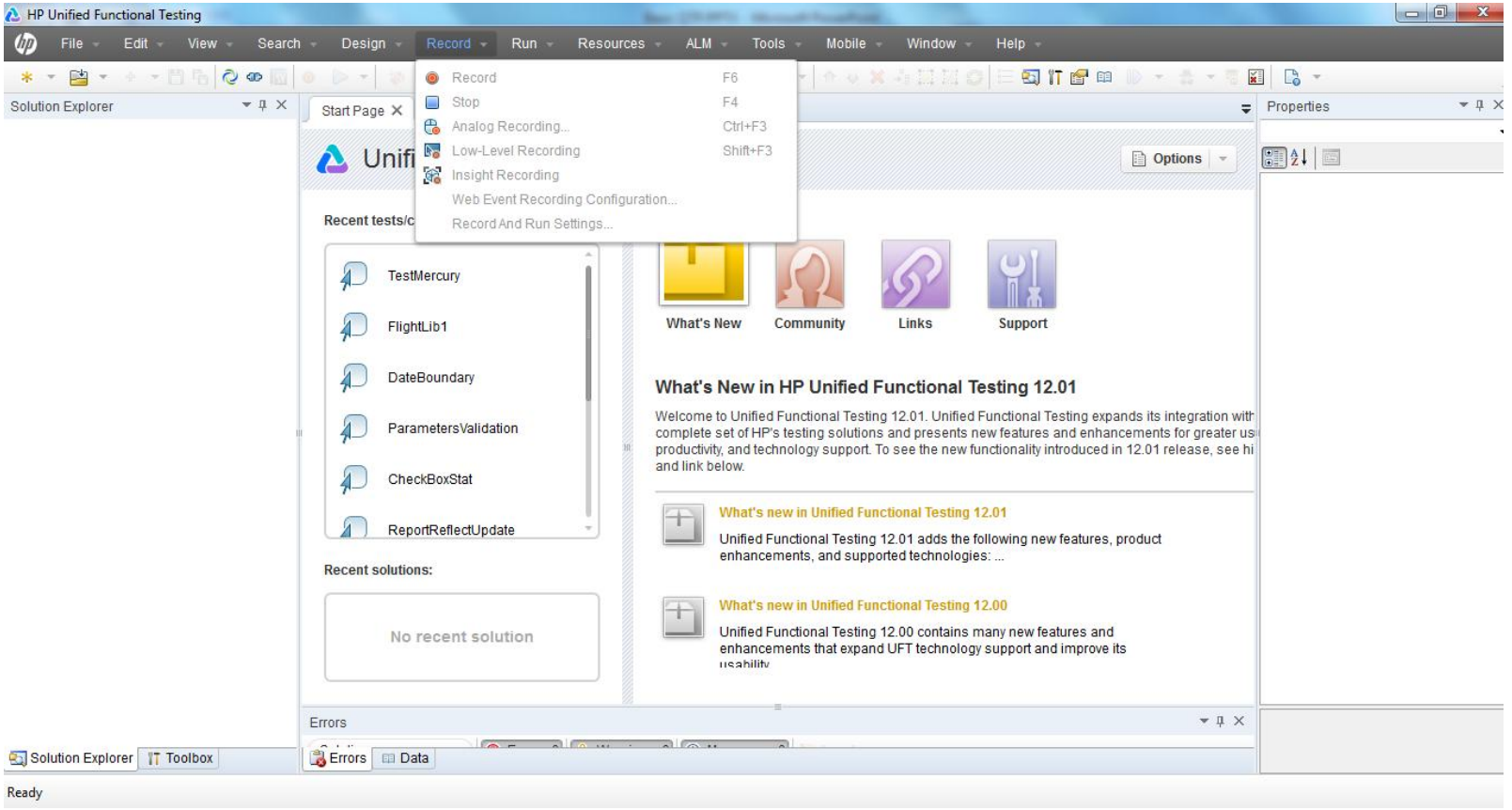
Using Search menu user can perform different search operation on test script.

User Interface of UFT : Design Menu



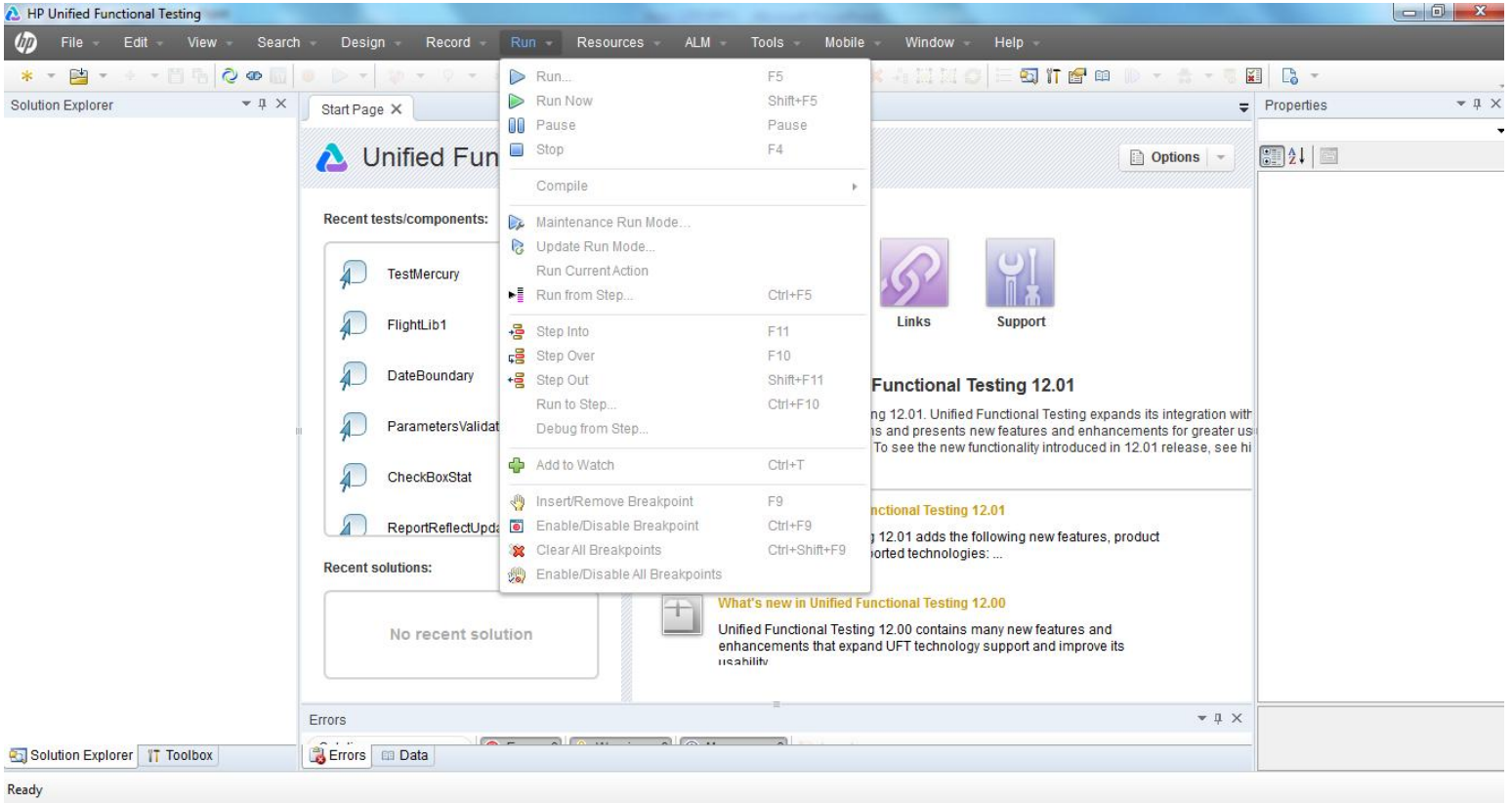
Using Design menu user can design structure of actions.

User Interface of UFT : Record Menu



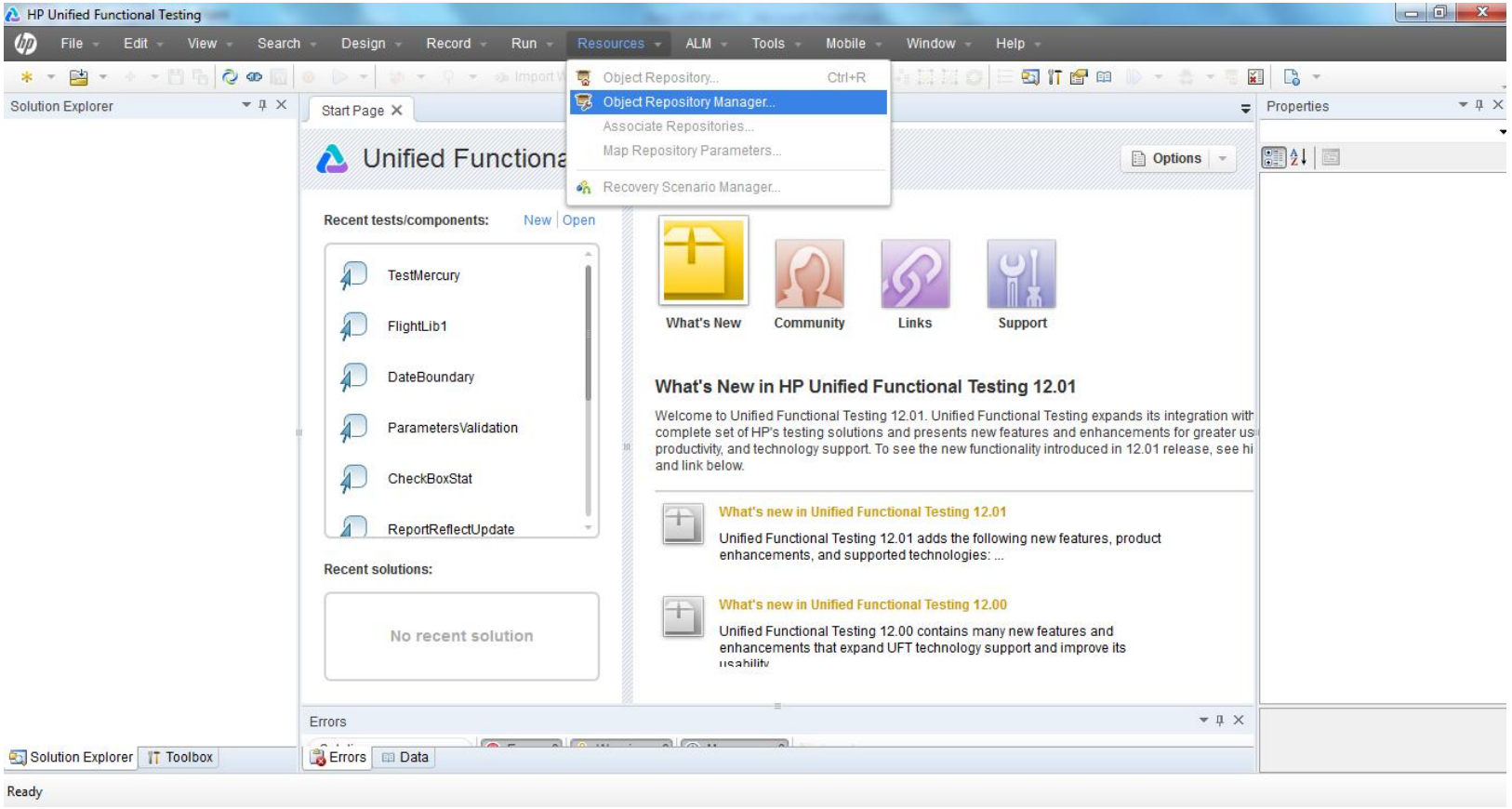
Using Record menu user can do recording on any open application.

User Interface of UFT : Run Menu



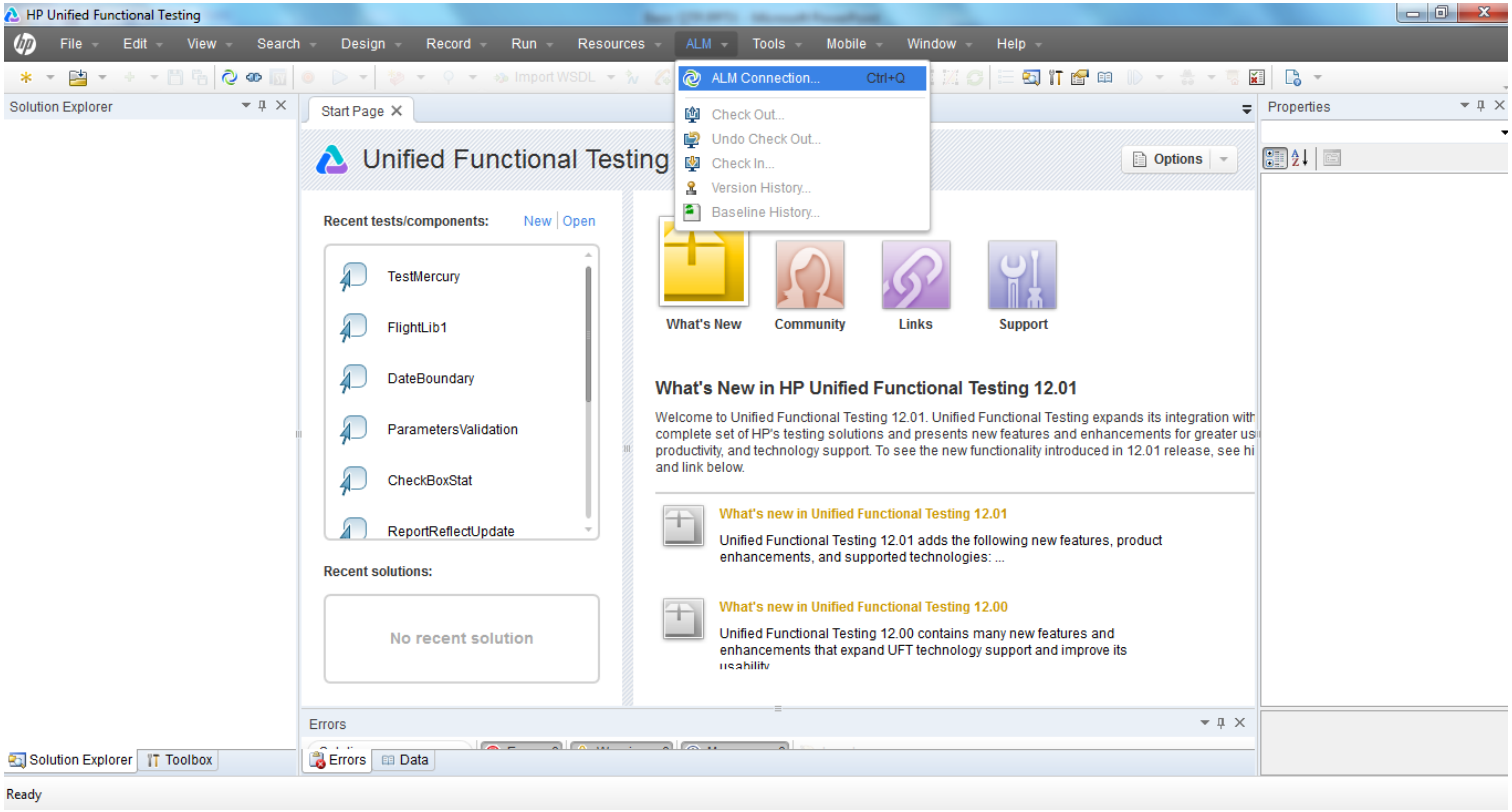
Using Run menu basically user can execute the test script.

User Interface of UFT : Resources Menu



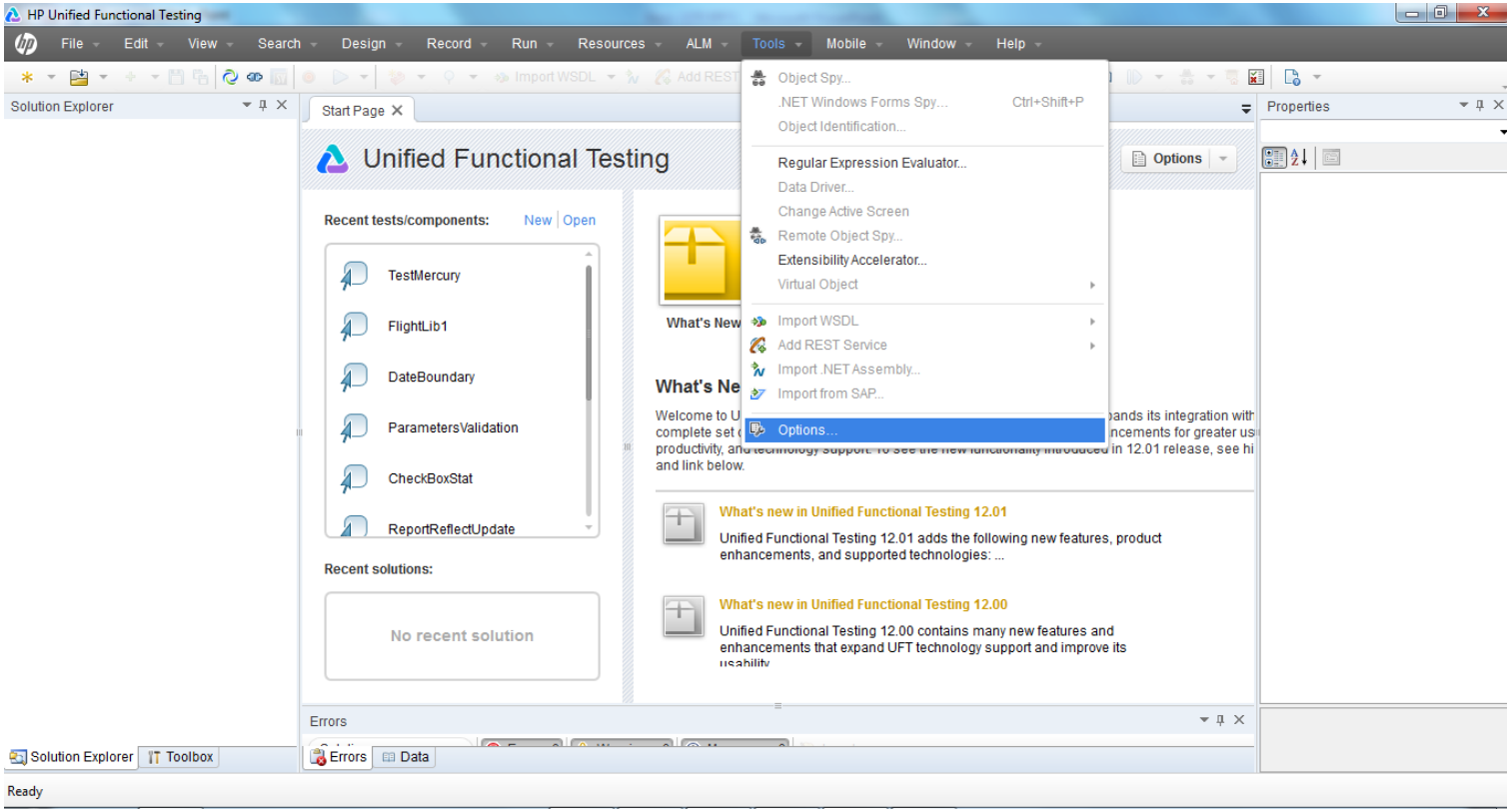
Using Resource menu user manage/create/update object repository.

User Interface of UFT : ALM Menu



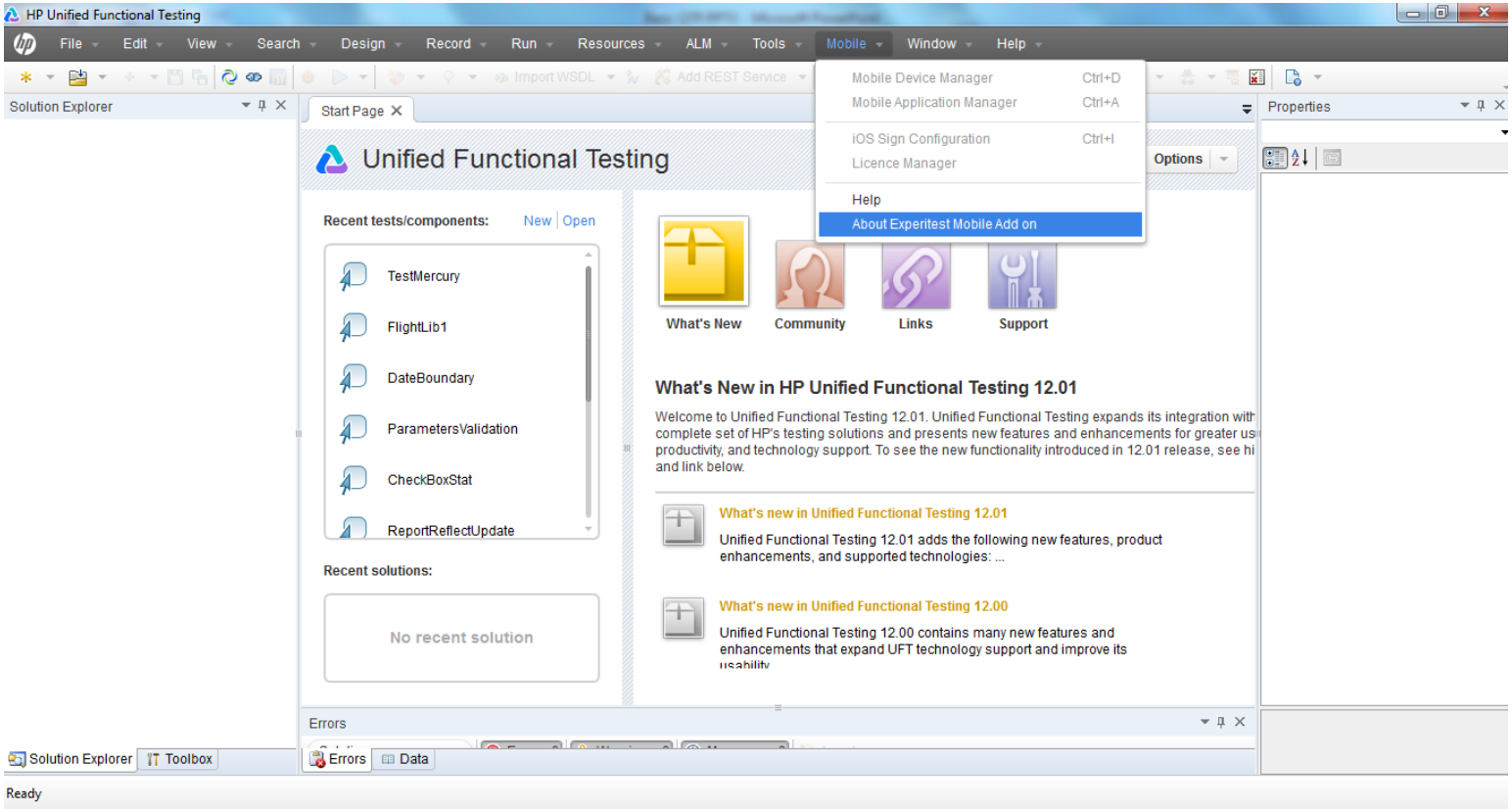
Using ALM menu user can connect to ALM application.

User Interface of UFT : Tools Menu



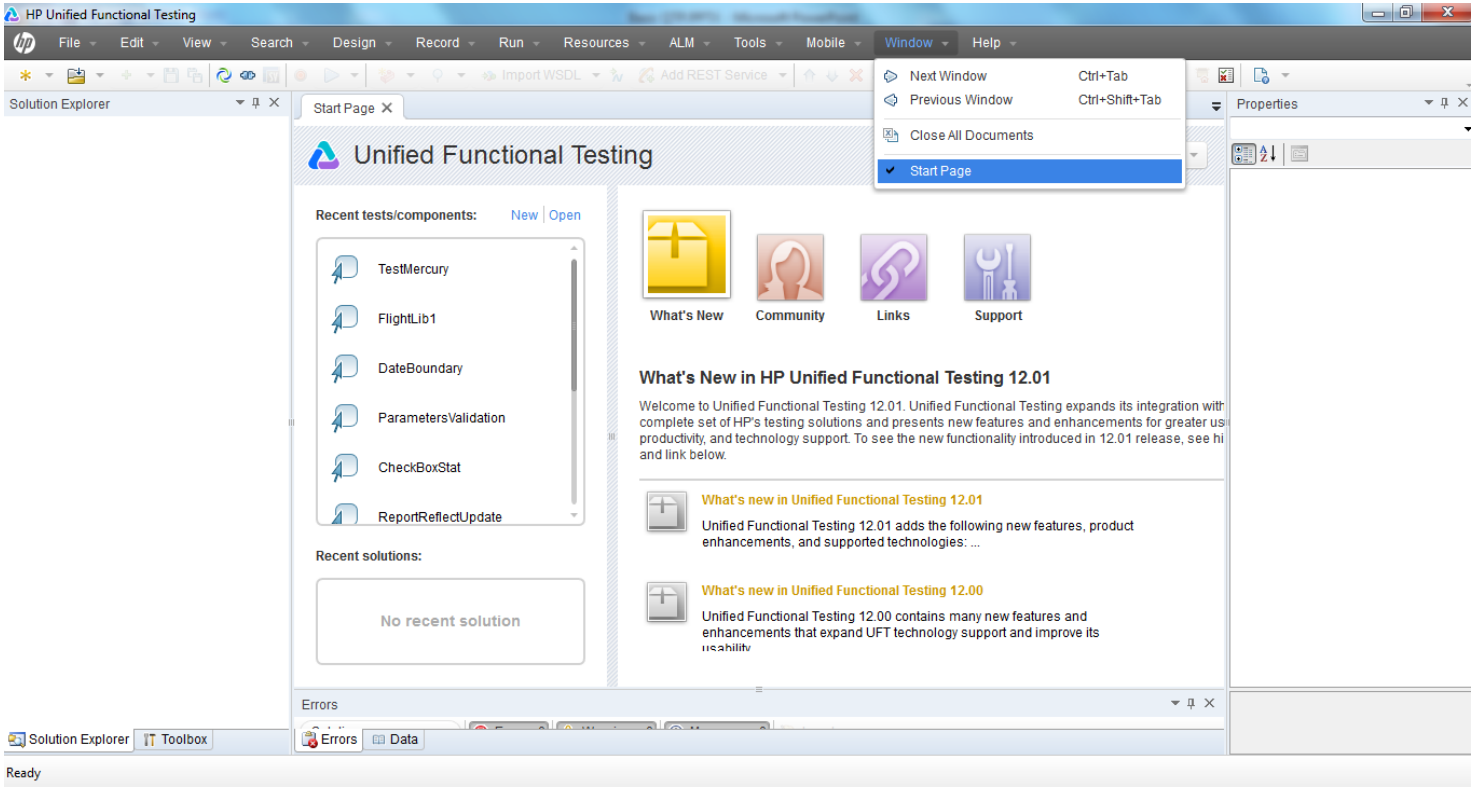
Using Tools menu user can efficiently automate the application.

User Interface of UFT : Mobile Menu



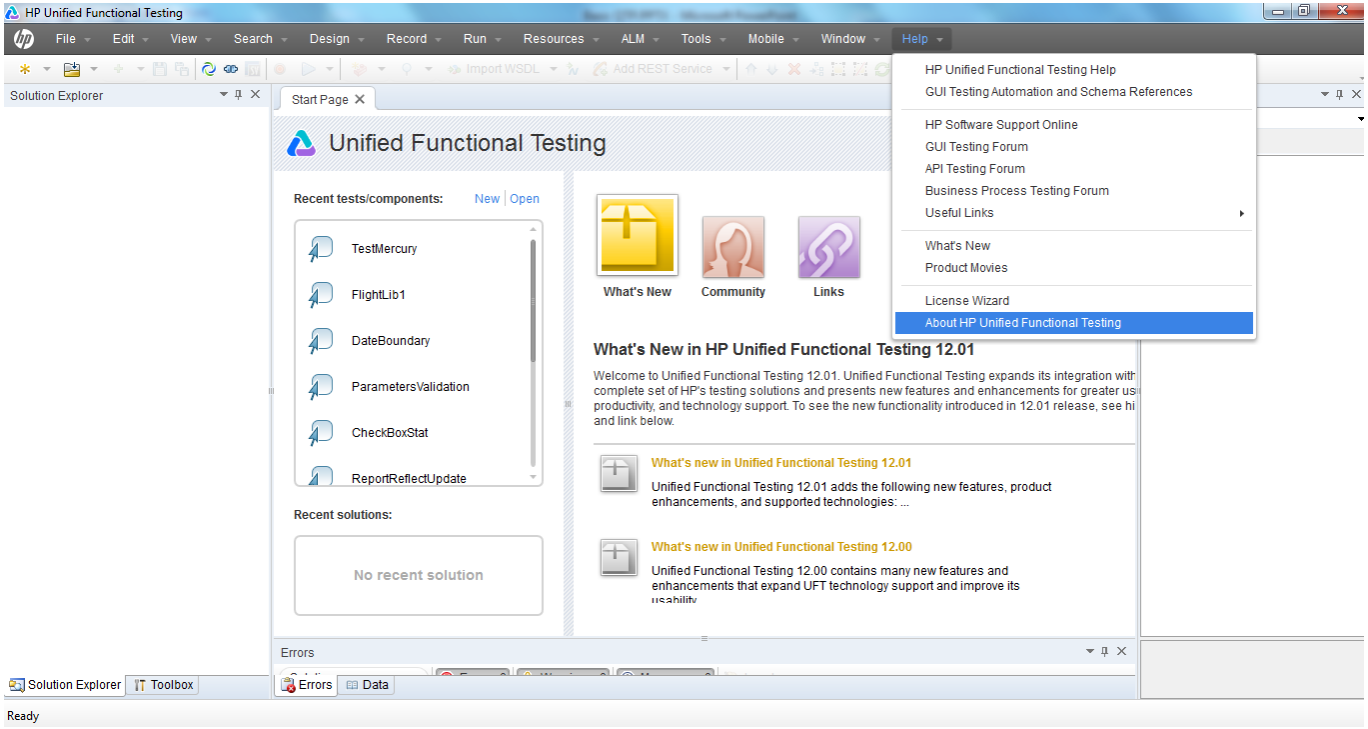
New feature of UFT for mobile application. Using Mobile menu user can automate mobile application.

User Interface of UFT : Window Menu



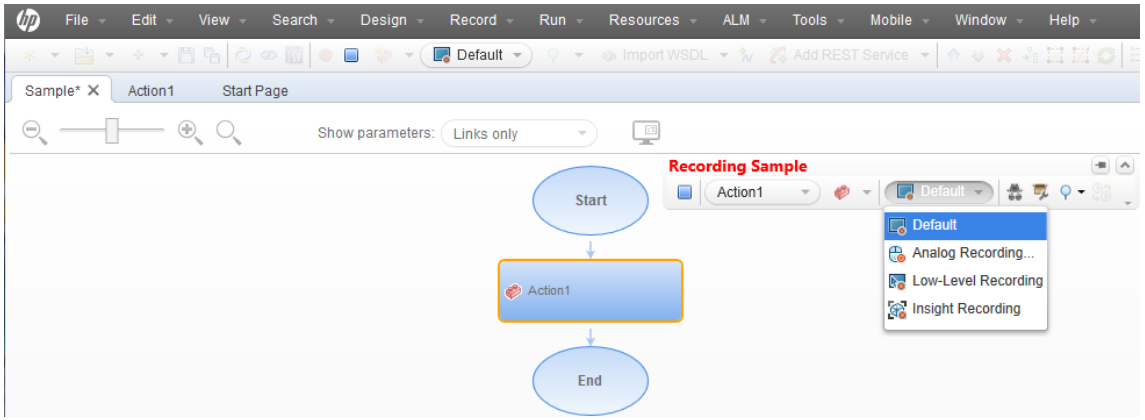
Window menu will help user to navigate from one window to another window.

User Interface of UFT : Help Menu



Help menu will provide user guide to user to understand How to use UFT effectively.

Recording modes



Normal Recording:- Is default mode of recording. It recognizes objects in the application regardless of their location on the screen. It records the objects and actions performed on them.

Analog Recording:- Enables you to record the exact mouse and keyboard operations you perform in relation to either the screen or the application window. In this recording mode, UFT records and tracks every movement of the mouse as you drag the mouse around a screen or window.

Low-Level Recording:- Enables you to record on any object in your application, whether or not UFT recognizes the specific object or the specific operation. This mode records at the object level and records all run-time objects as Window or WinObject test objects. Use low-level recording for recording in an environment or on an object not recognized by UFT. You can also use low-level recording if the exact coordinates of the object are important for your test or component.

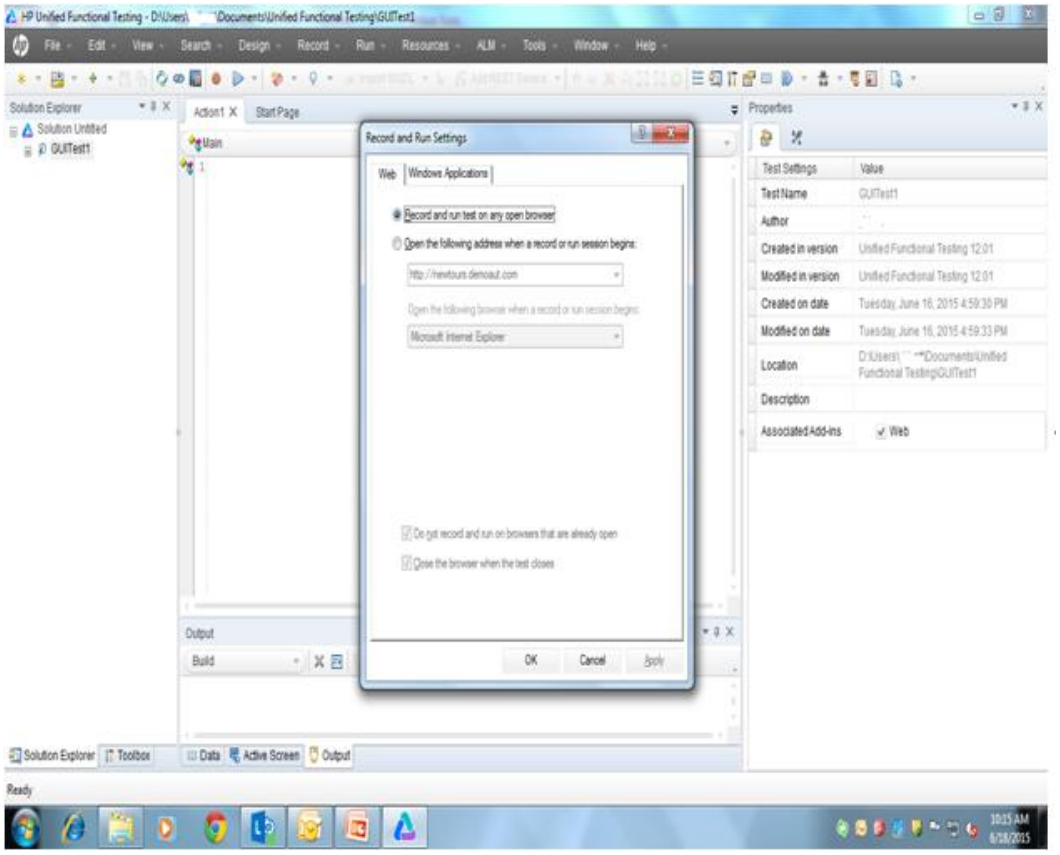
Insight Recording : - UFT records operation based on its appearance and NOT based on its native properties.

Recording a Test

1. Click on Record(F6) button to start the recording. It launches Record and Run Settings which has settings for each selected add-ins.

2. Perform certain steps

3. Click Stop(F4) button to stop the recording

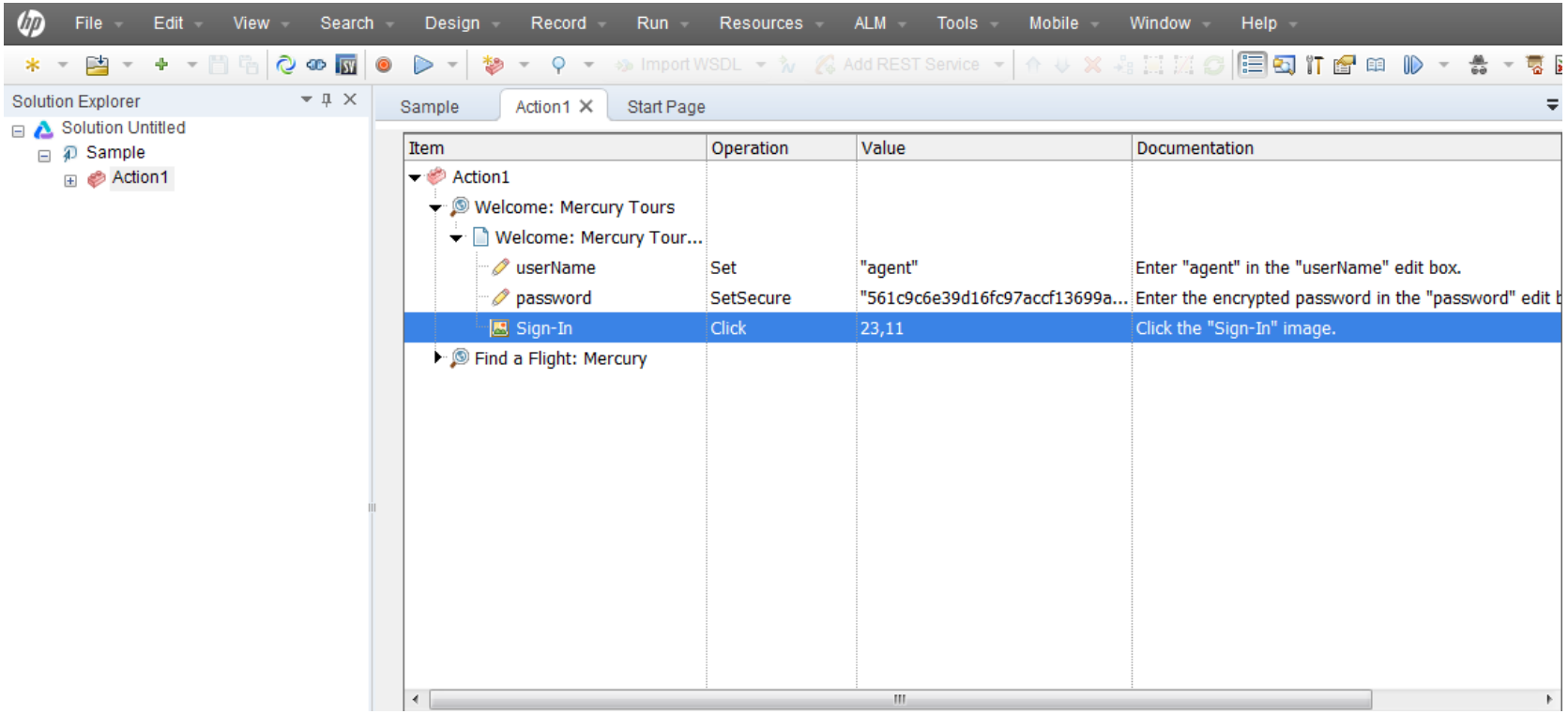


Views of the scripts

The recorded test is displayed in two views :

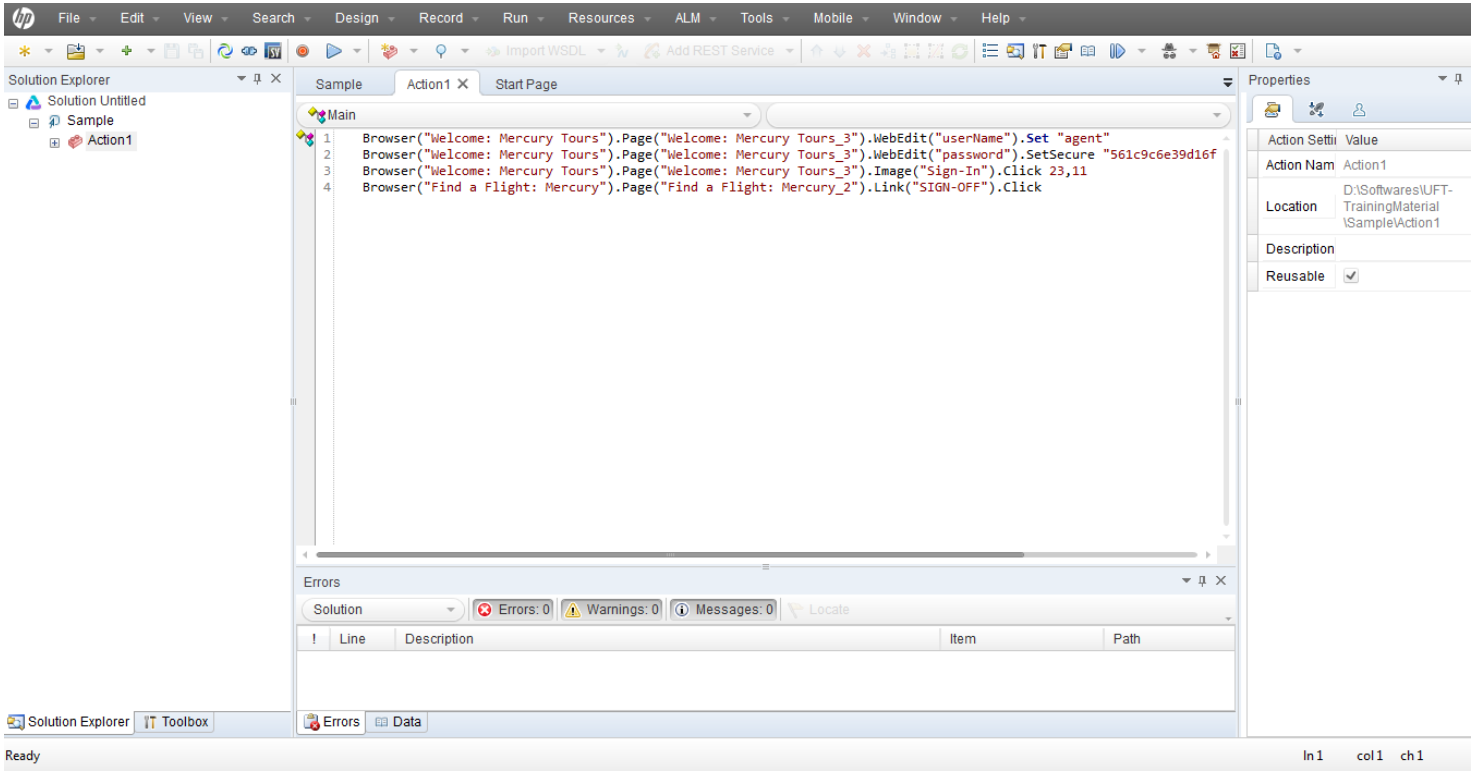
1. ***Keyword View*** which shows the test in a keyword driven way, which is modular and has documentation to explain each step explicitly.
2. ***Expert View*** shows the underlying VB script code corresponding to each of the operation performed while recording.

Keyword view of the recorded script



The Keyword View is comprised of a **table-like view** where **Each step** is a **separate row** in the table and **Each column** represents different parts of the steps.

Expert View of the recorded script



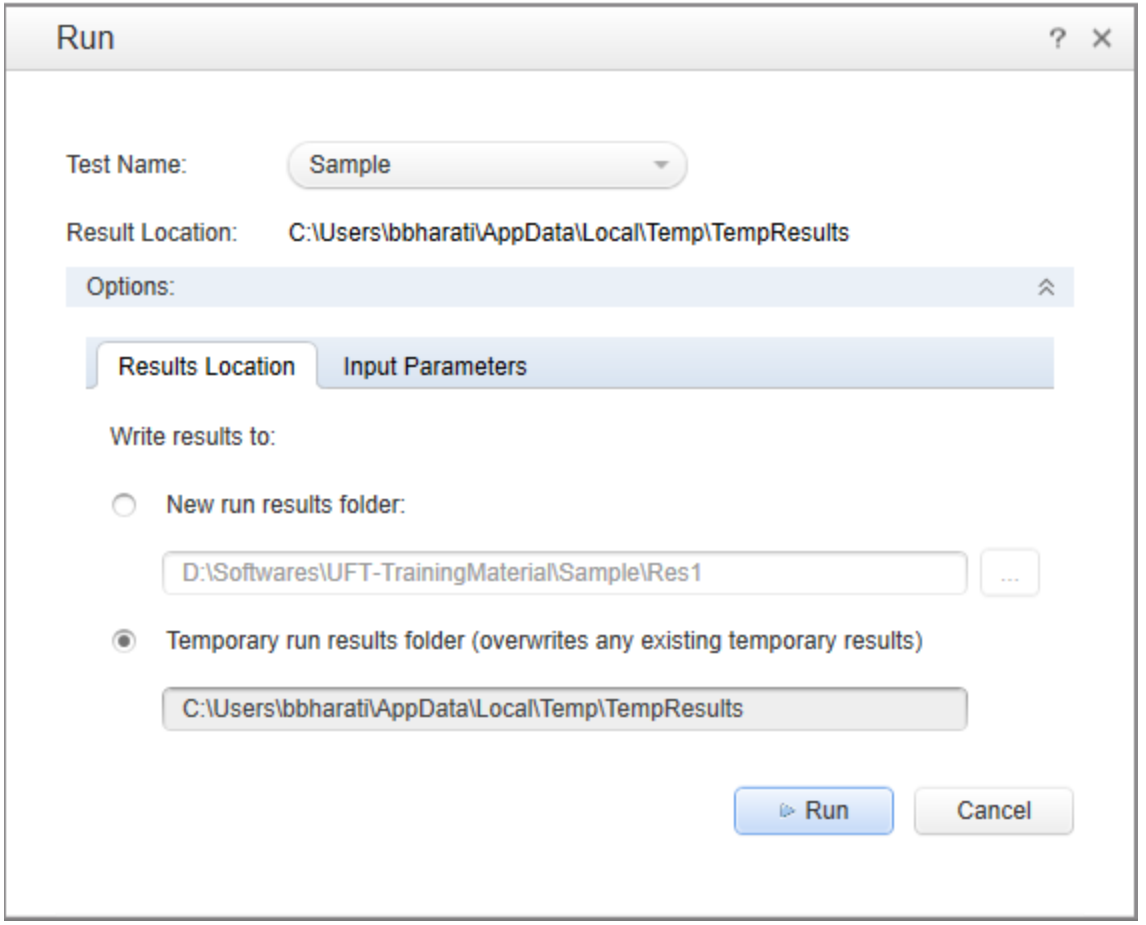
In the Expert View , each line represents a Test Step in VB Script.

Running the test

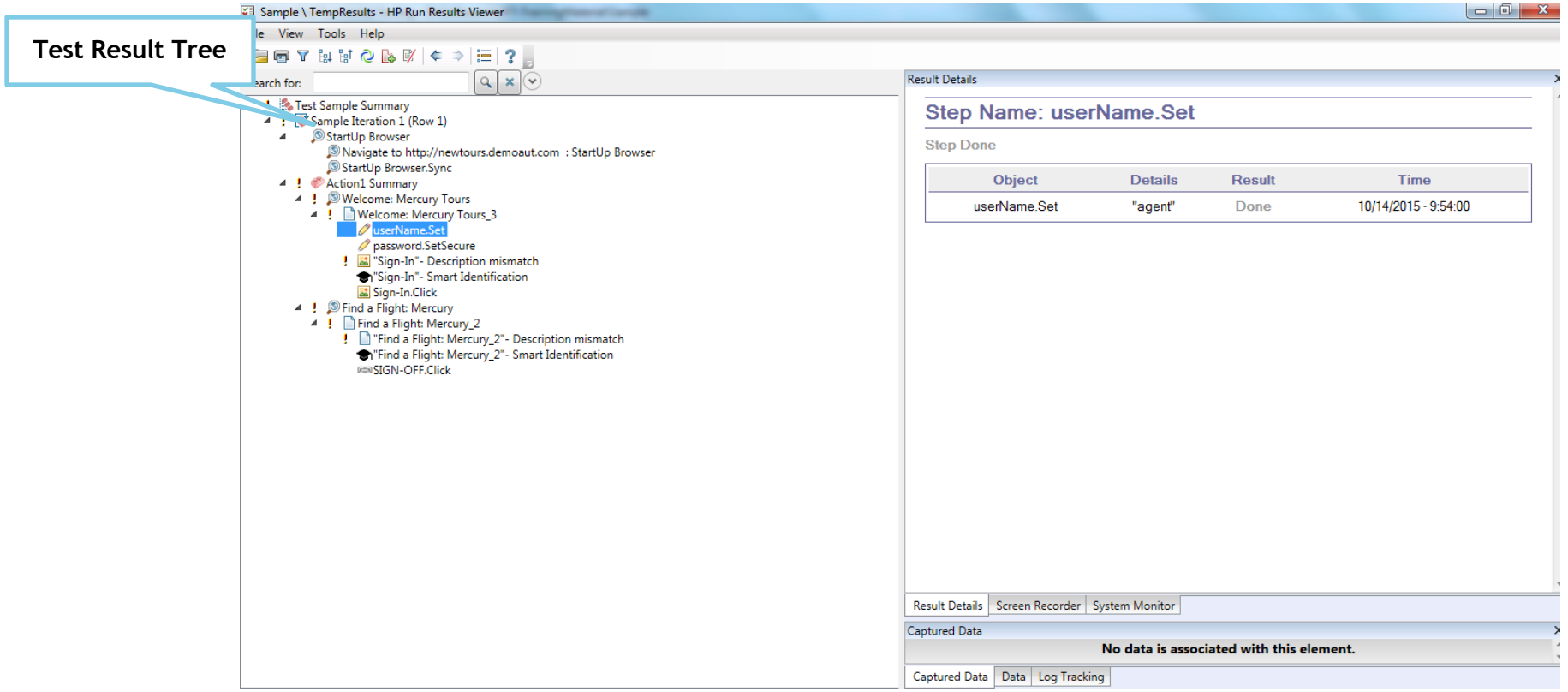
1. Click on Run(F5) button to start the execution.

2. Run dialog is displayed

3. For Result folder user need to select either “**New Run result folder**” Option or “**Temporary run result folder…**” Option



Test Result Window



The Test Results Window gives us sufficient information to show the steps passed, failed etc.

Parameterization

- Parameterization allows creating maintainable scripts which can run with different set of data.
- The data is not hard-coded in the script and are replaced with parameters in order to execute the script with different data.

Types of Parameterization

1. ***Data Table parameters*** → It helps to create a data-driven test (or action) that runs several times using the data in the data sheet. UFT substitutes the constant value with a different value from the Data Table for each iteration.
2. ***Environment Variables*** → It helps in using data from other sources like environment variables.
3. ***Random Number variables*** → The random number input generates random numbers and uses them as input value for a parameter. By default, the random number ranges between 0 and 100. A different random number is generated every time the parameter is called for every iteration or for every Test run.
4. ***Test/Action Parameter*** → Action parameters enable you to transfer input values from your test to a top-level action, from a parent action to a nested action, or from an action to a sibling action that occurs later in the test . For e.g. Msgbox Parameter("Name")

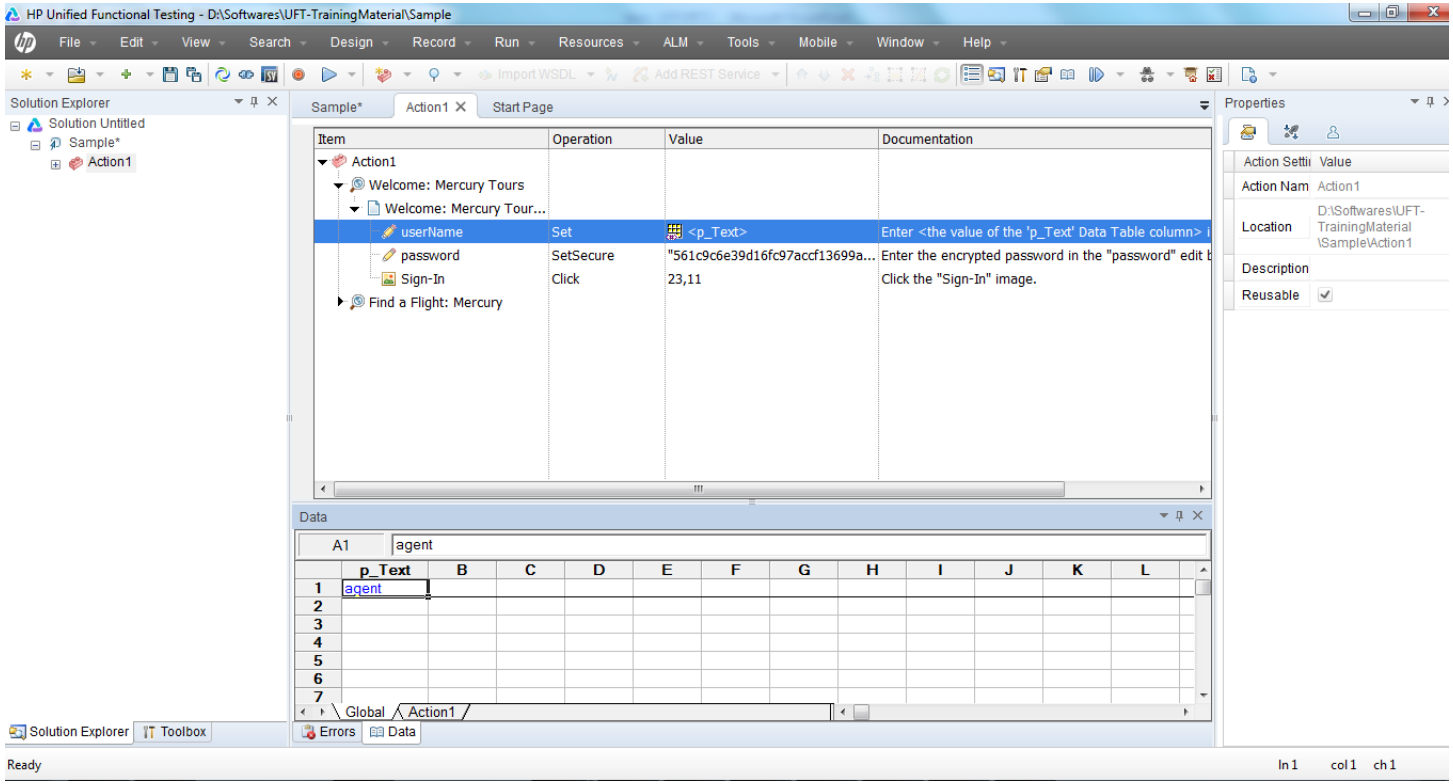
   Test Parameter is same as action parameter only the difference is that it will be available to all the actions in the test in which it is defined.

   For e.g. TestArgs("Name") = Value

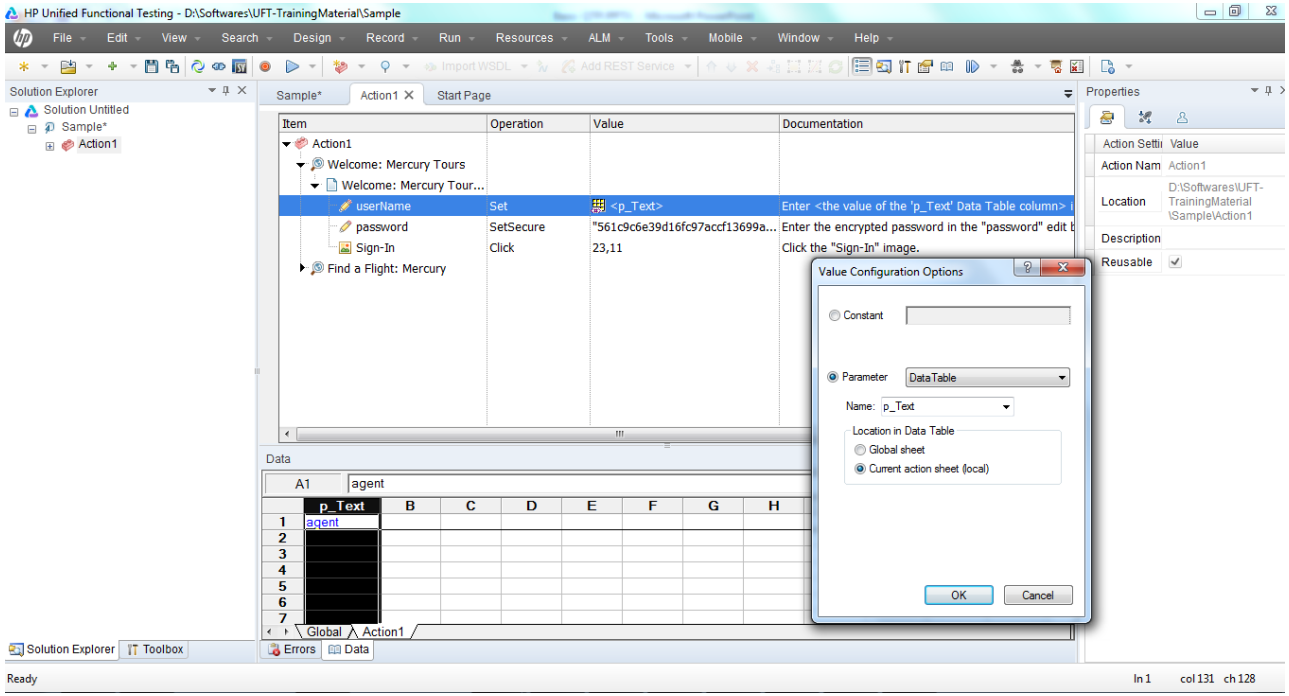
Parameterization in Global sheet of Datatable

Code generated in Expert View:

✓ Browser("Welcome: Mercury Tours").Page("Welcome: Mercury Tours_3").WebEdit("userName").Set DataTable("p_Text", dtGlobalSheet)



Parameterization in Global and Local sheet of Data table

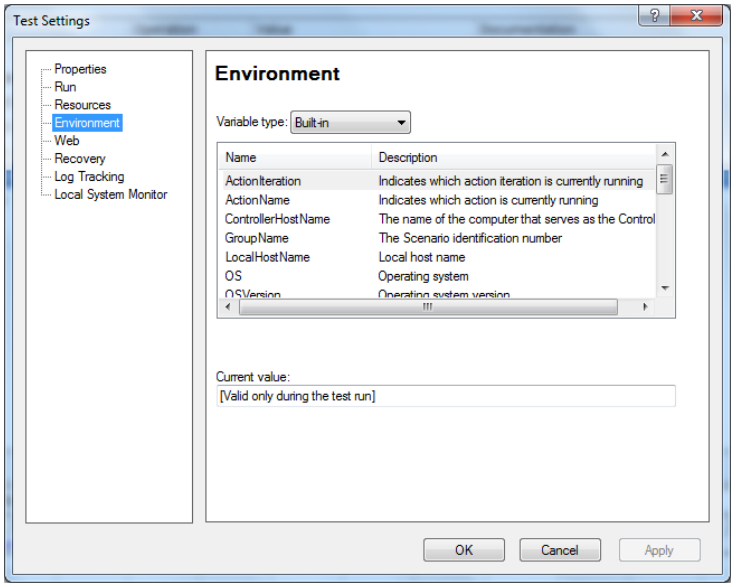


Parameterized both the hard coded values in scripts

✓For e.g. Search is parameterized and derived from Global sheet.

✓Similarly it can be derived from Local sheet too.

Environment Variables



Types of Environment Variables

1) Built-in – These are provided by UFT. Built-in variable list can be accessed from File->Settings->Environment option.

   E.g. sOS = Environment("OS")

2) User Defined – Tester can define own environment variables. There are two types of User defined variables

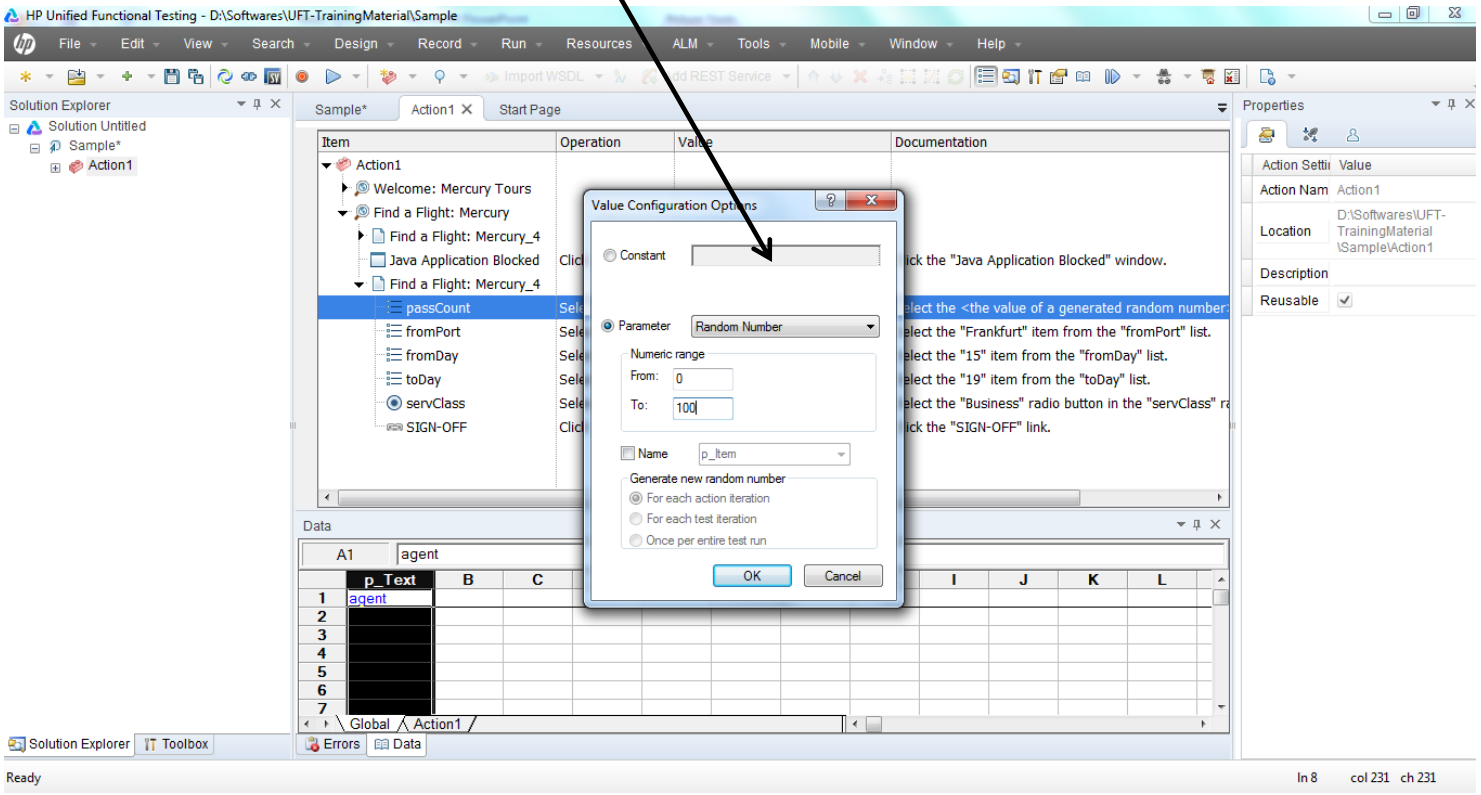
- Internal: These are the variables that are defined in the test.

  E.g. Environment. Value("Path") = "C:\Test"

- External : These are the variables that predefined in the external environment file (xml) and can be used in the test

Random numbers

Clicking on ‘Configure the value’ shows ‘Value Configuration options’ dialog



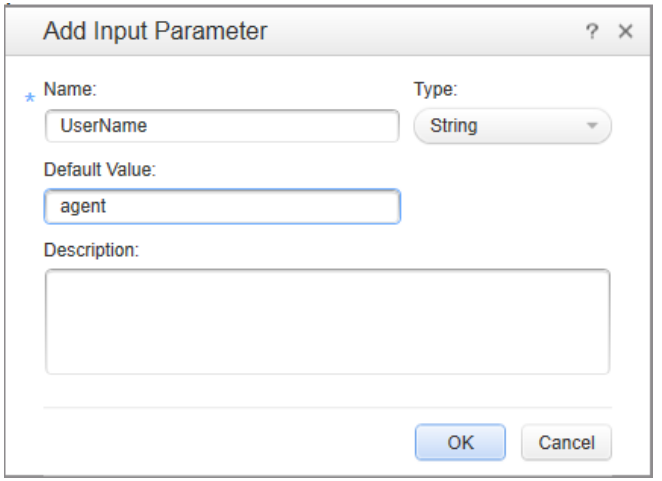
Test Parameters on Test Settings dialog

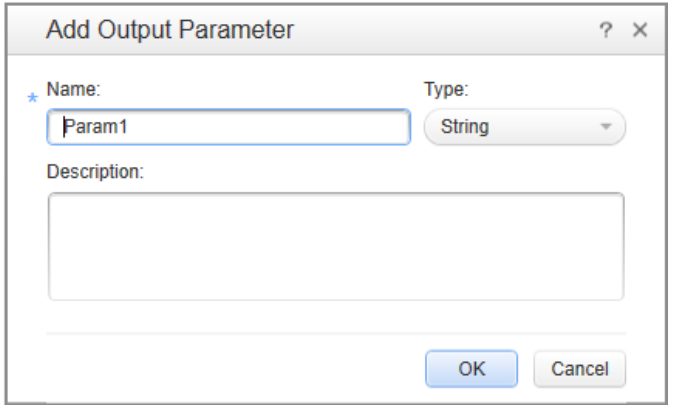
❑ Code to insert the value using Parameter:

Browser("Welcome: Mercury Tours").Page("Welcome: Mercury Tours_5").WebEdit("userName").Set Parameter("UserName")

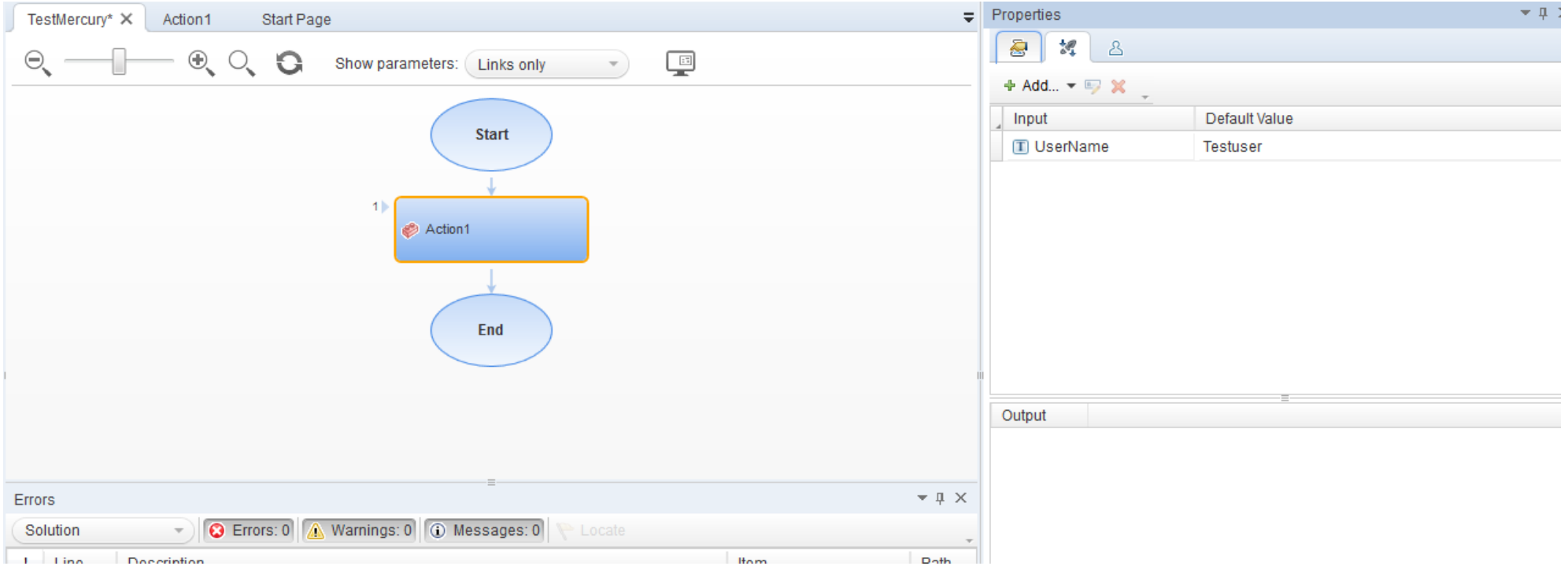
❑ Code to display the value in 'Username' Test Parameter:

Msgbox Parameter("UserName")





Action Parameters on Action Properties dialog



- Code to display the value in 'Username' Action Parameter:

```
Msgbox Parameter("Username")
```

- Code to change the Action Parameter:

```
Parameter("Username") = "Testuser"
```

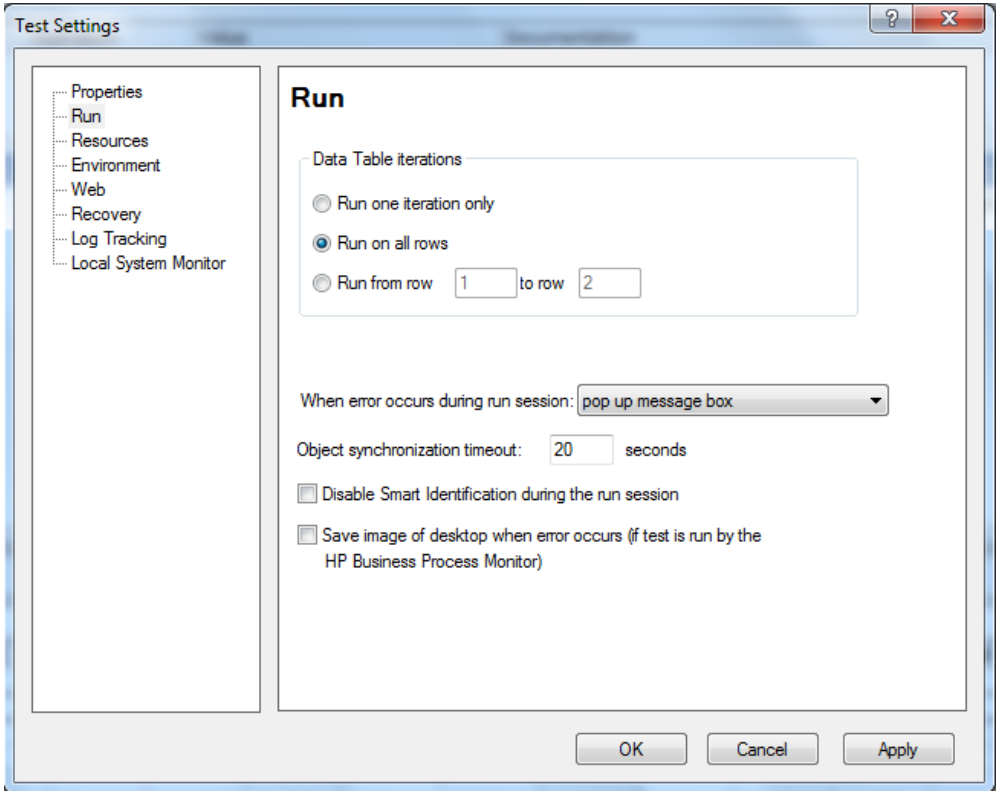
Test Settings for executing all the iterations

Data Table iterations Specifies the iterations for the test. Select an option:

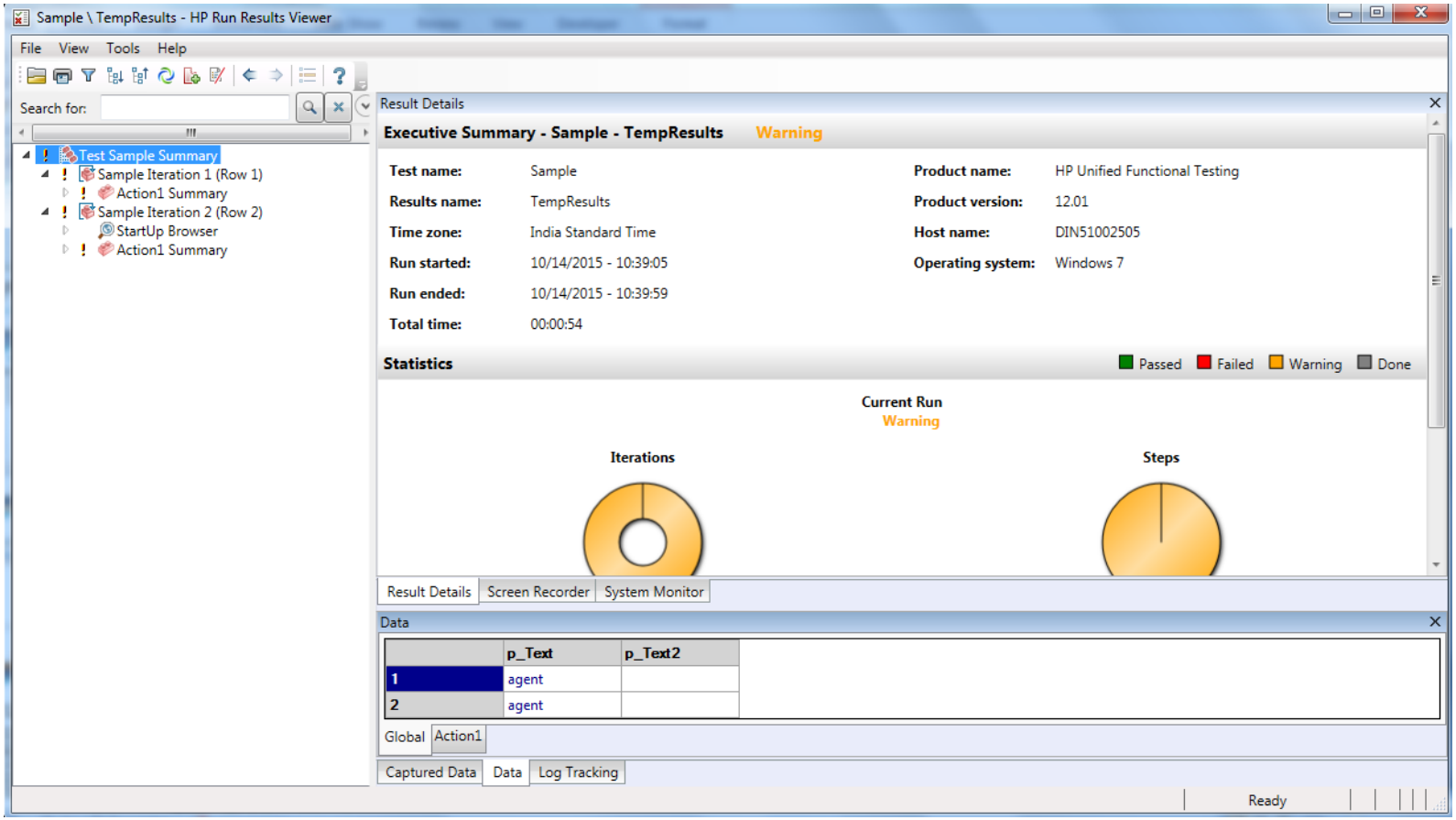
•**Run one iteration only**:- Runs the test only once, using only the first row in the global data table.

•**Run on all rows**:- Runs the test with iterations using all rows in the global data table.

•**Run from row __ to row __:**- Runs the test with iterations using the values in the global data table for the specified row range.



Result File with multiple iterations



Introduction to Actions

UFT is made up of logical units called actions.

Actions help in making the test modular and increase the reusability of test script.

A given test can have multiple actions, with each action containing a logical sequence of steps.

Actions can be of three kinds. They are :

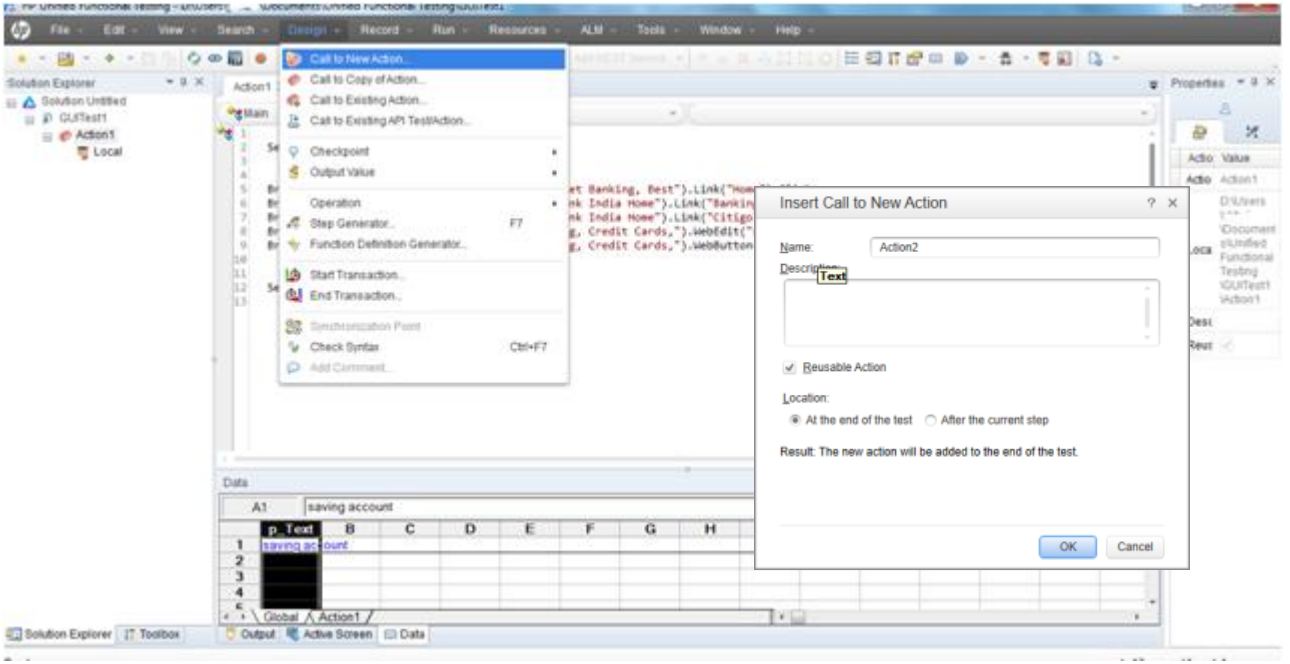
- ***Non reusable action*** : An action which cannot be called by other actions. By default all actions which get created are Non-reusable actions.
- ***Reusable actions*** : An action that can be called multiple times by the same test or any other test
- ***External action*** : A reusable action which can be called from a different test

Creating new Actions

1) Select Insert->Call to New Action menu

2) Insert Call to New Action dialog is displayed

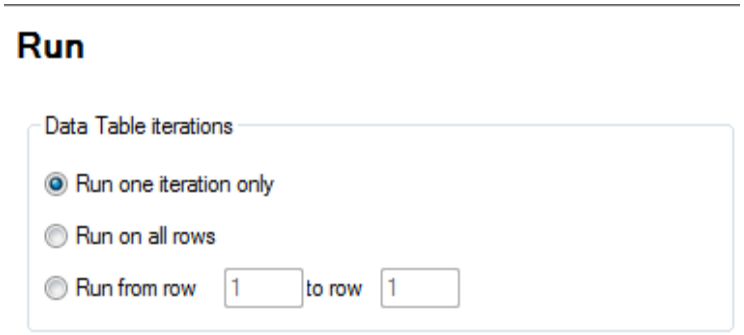
3) Enter Action name and check the checkbox in case Reusable Action needs to be created



Action call properties

Action call properties determine the number of iterations the called action can perform. The properties set is applicable only to particular action call.

The options available are as follows:



1) Run one iteration only→ Runs the called action only once

2) Run on all rows→ Runs the called action with the number of iterations according to the number of rows in the action's Data Table.

3) Run from row __ to row __→ Runs the called action with the number of iterations according to the specified row range.

Inserting Calls to Existing Action

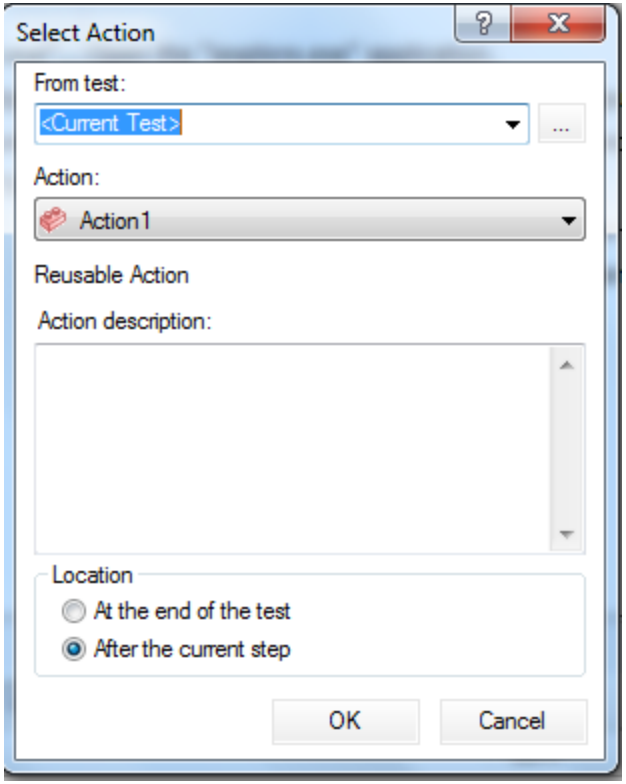
The Steps to call an external action are as follows :

1) Go to Design → Call to existing action, this opens a Select Action dialog where we can select the test from which the action needs to be called.

2) Once the test is selected the reusable actions available in that test are visible. Select the Action which is required.

Code in Expert View:

RunAction "Action1", oneIteration



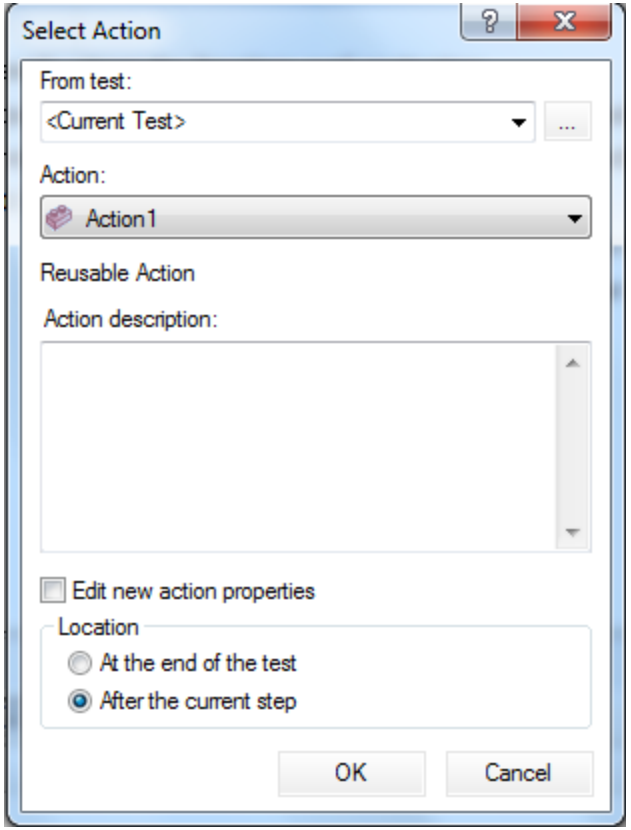
Inserting Calls to Copy of Action

The following steps are used for inserting calls to copies of Actions:

1) Go to Design → Call to Copy of Action, this opens a selection dialog where we can select the test from which the action needs to be called.

2) Once the test is selected all the actions available in that test are visible. Select the Action which is required. Choose where to put the action.

3) The called action can be modified in the calling test without affecting the actual action.



Objects & Object Identification

- Objects are a representation of every item found in an application.
- Objects are visual(e.g. Button and Text) and non-visual (e.g. Dictionary, Reporter) elements. Each object has its properties and methods.
- UFT learns objects of the application based on their properties.
- UFT stores the object data along with properties in the object repository.

Object Identification:

- UFT uses default Object Identification properties : Mandatory & Assistive to learn objects stored in Object Repository.
- If successful identification was not possible then go for Smart Identification using Base filter properties & Optional base filter properties.
- Ordinal Identifiers like (index , location or creation time) can be used if Smart Identification is disabled.

Introduction of Object Repository

Object repository is the collection of the objects which UFT has identified during the recording.

- The objects are represented in a tree view.
- The object repository can be accessed by the following path:
  - Resources → Object Repository Manager
  - Click on the object repository toolbar button
  - Keyboard keys (Ctrl+R)
- Even when steps containing a test object are deleted from the test or component, the objects remain in the object repository.

Types of Object Repository

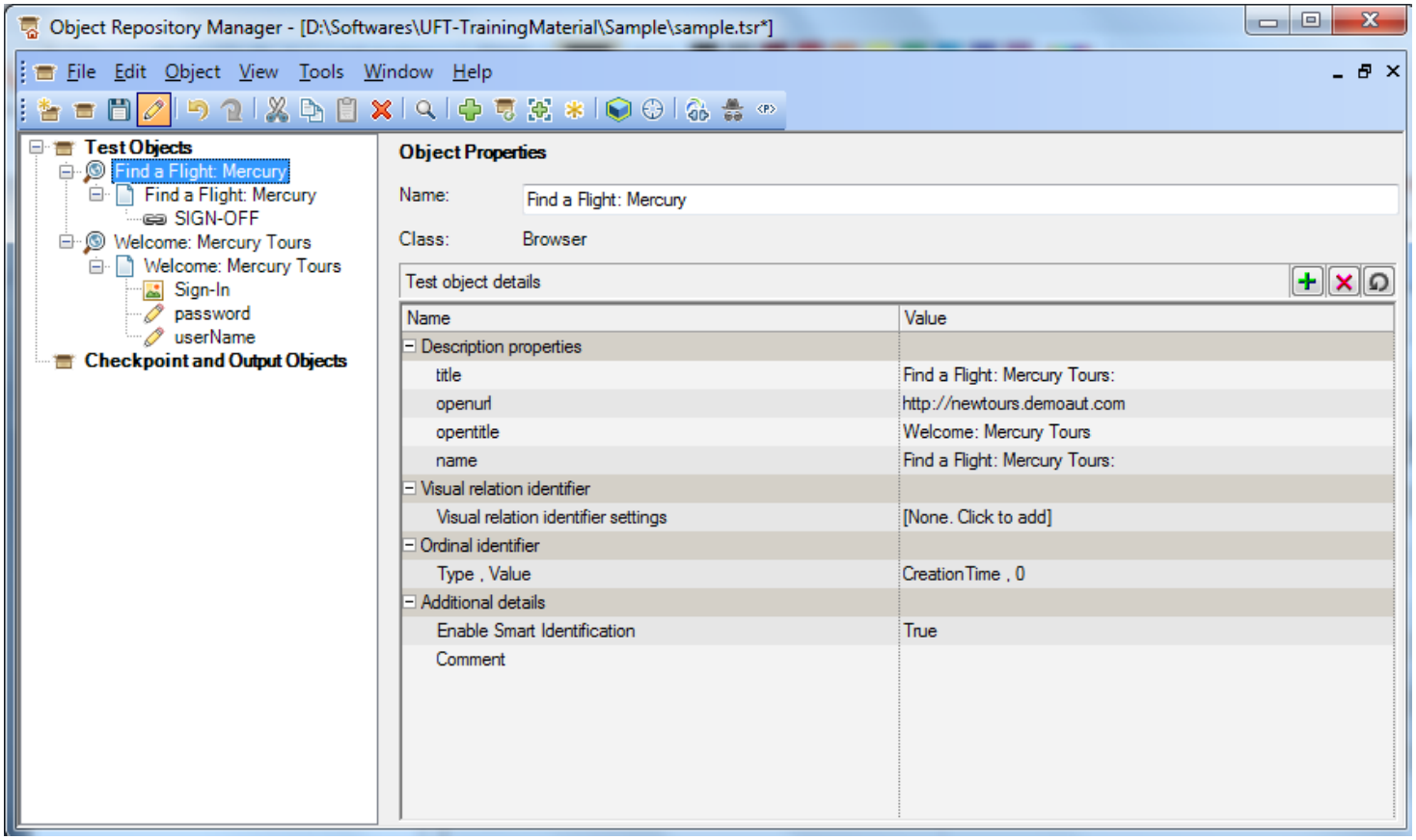
Local Object Repository

- UFT uses a separate object repository for each action.
- When you save your test, all of the local object repositories are automatically saved with the test (as part of each action within the test).

Shared Object Repository

- UFT uses the same object repository for different actions.
- You can export the local objects to a shared object repository .

Test Objects in Object Repository

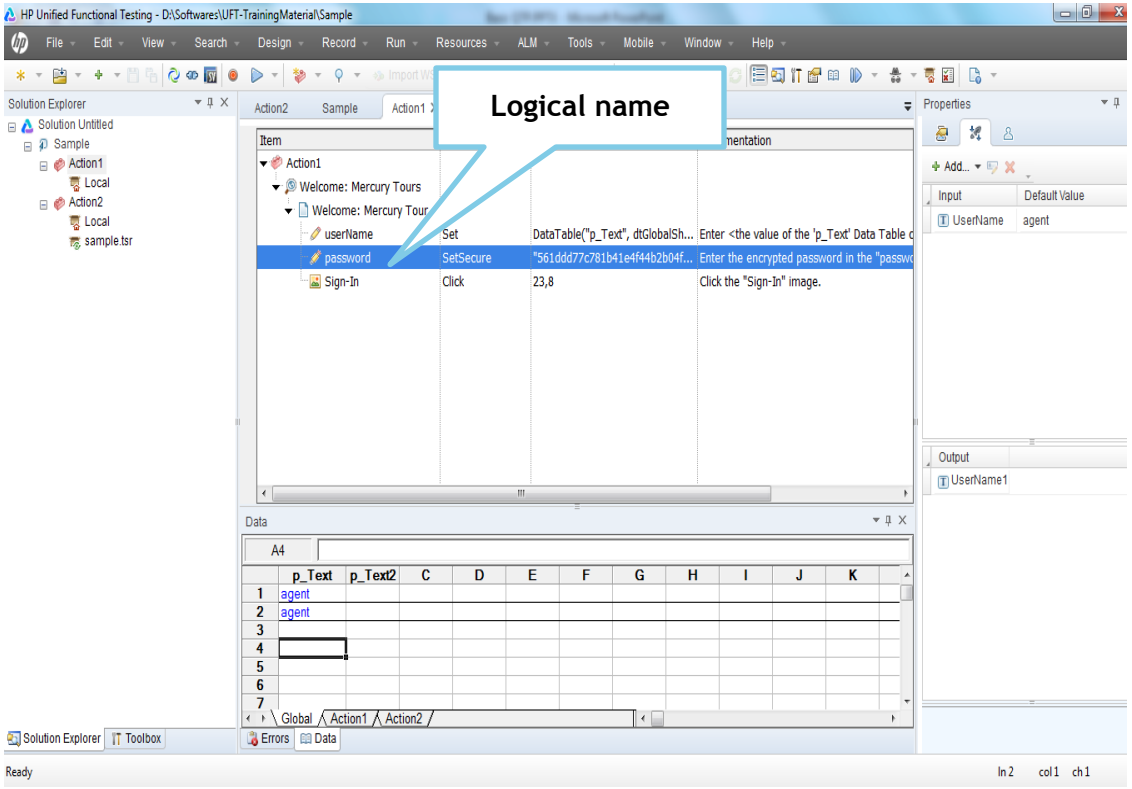


Logical Name of an Object

- After reading the class and properties of an object, UFT assigns a logical name to the object

- UFT refers to the recorded objects by using its logical name

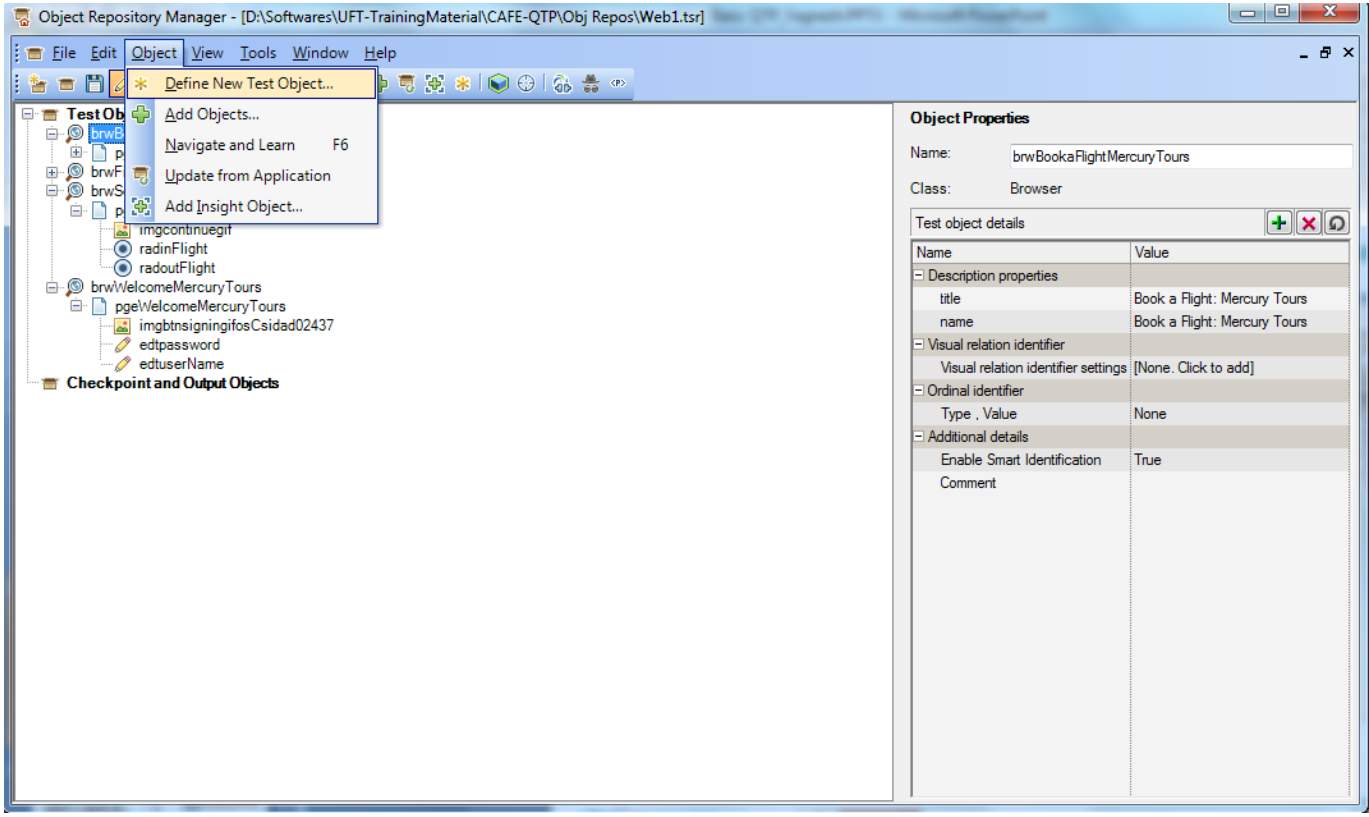
- The logical name can be edited and used in UFT scripts



Object Repository Manager - Define New Test Object

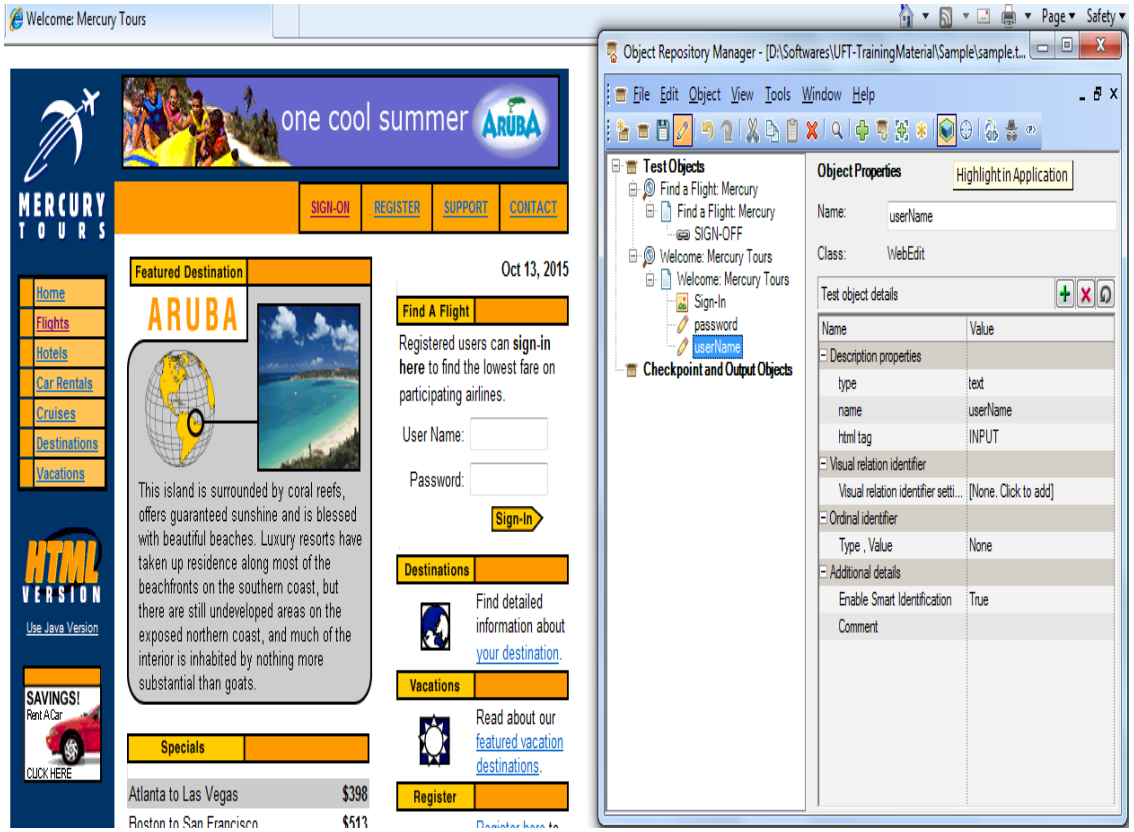
Define New Test Objects helps in adding Test object that do not yet exist in application. This enables to prepare an object repository and build tests or components for application before the application is ready for testing.

Objects can be added to Local or Shared Object Repository



Object Repository Manager - Highlight in Application

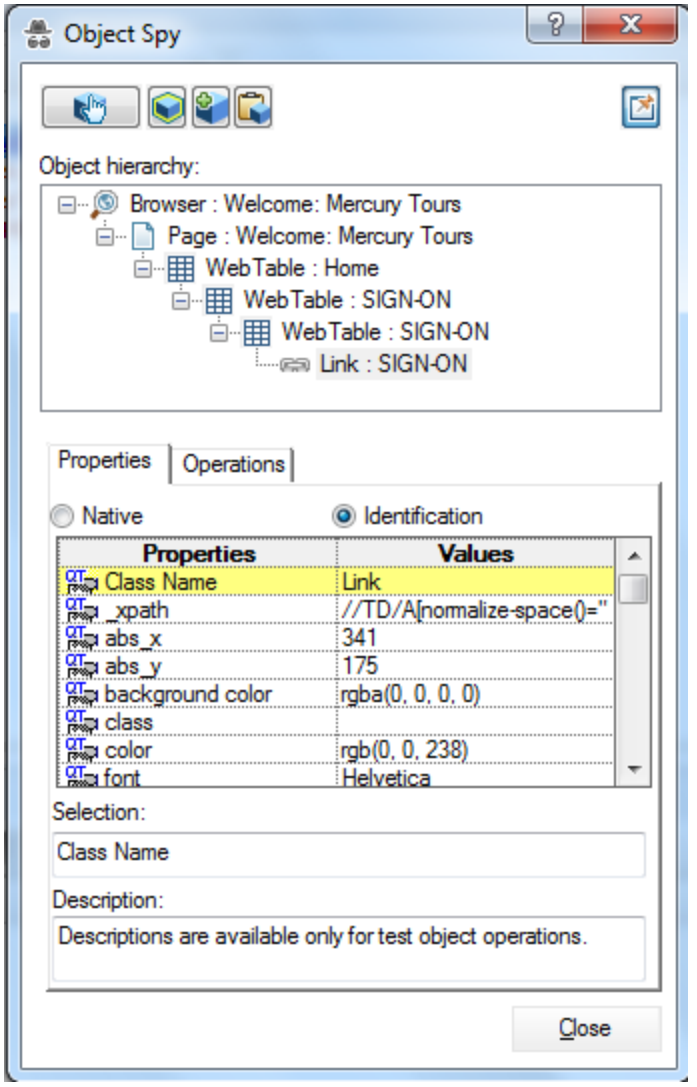
Select a test object in object repository and highlight it in the application. When highlight in Application is selected, UFT indicates the selected object's location in the application by temporarily showing a frame around the object and causing it to flash briefly.
The application must be open to the correct context so that the object is visible.



Object Repository Manager - Object Spy

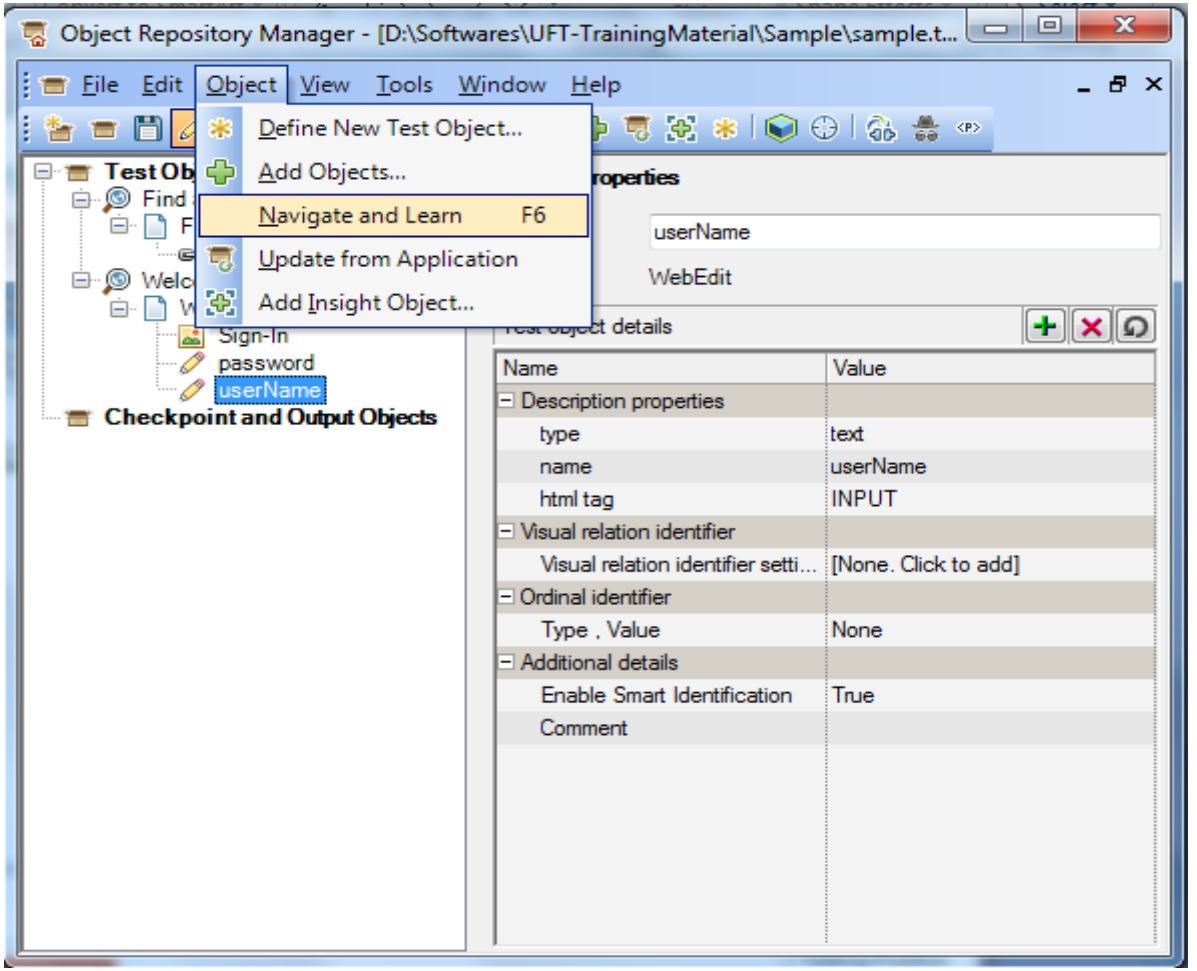
It enables to view the native properties and operations of any object in an open application, as well as the test object hierarchy, identification properties, and operations of the test object that UFT uses to represent that object.

This helps in finding the current object properties of any test object.



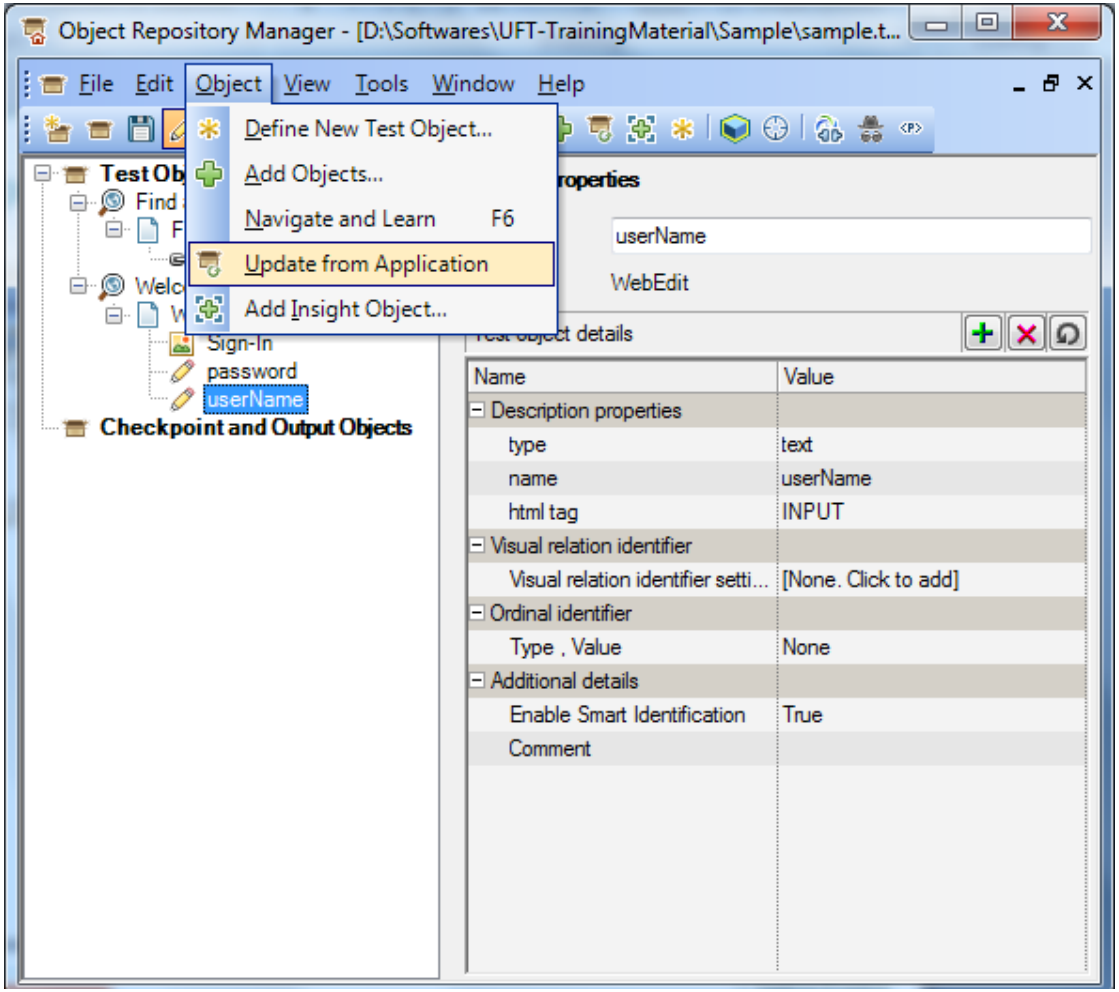
Object Repository Manager - Navigate and Learn

The Navigate and Learn option enables to add multiple test objects to a shared object repository based on defined filter while navigating through the application.



Object Repository Manager - Update from Application

As the application changes, it is important to update individual test object properties from the object in the application using the Update from Application option



Introduction to Synchronization

When we run test it might happen that the time taken by the application to respond is more and the images might not be loaded in which case UFT will report an error as it will not be able to find the object during run time. In such cases we need to synchronize our tests.

We can synchronize using the following options –

1. Insert a synchronization point
2. Use EXIST or WAIT statements
3. Modify the default amount of time that UFT waits for a Web page to load i.e. Modify object synchronization timeout value

Synchronization can be done for following tasks

- For a progress bar to reach 100%
- For a status message to be displayed
- For a property change of an object
- For a window or pop-up message to be displayed

Adding Synchronization Point in Recording mode

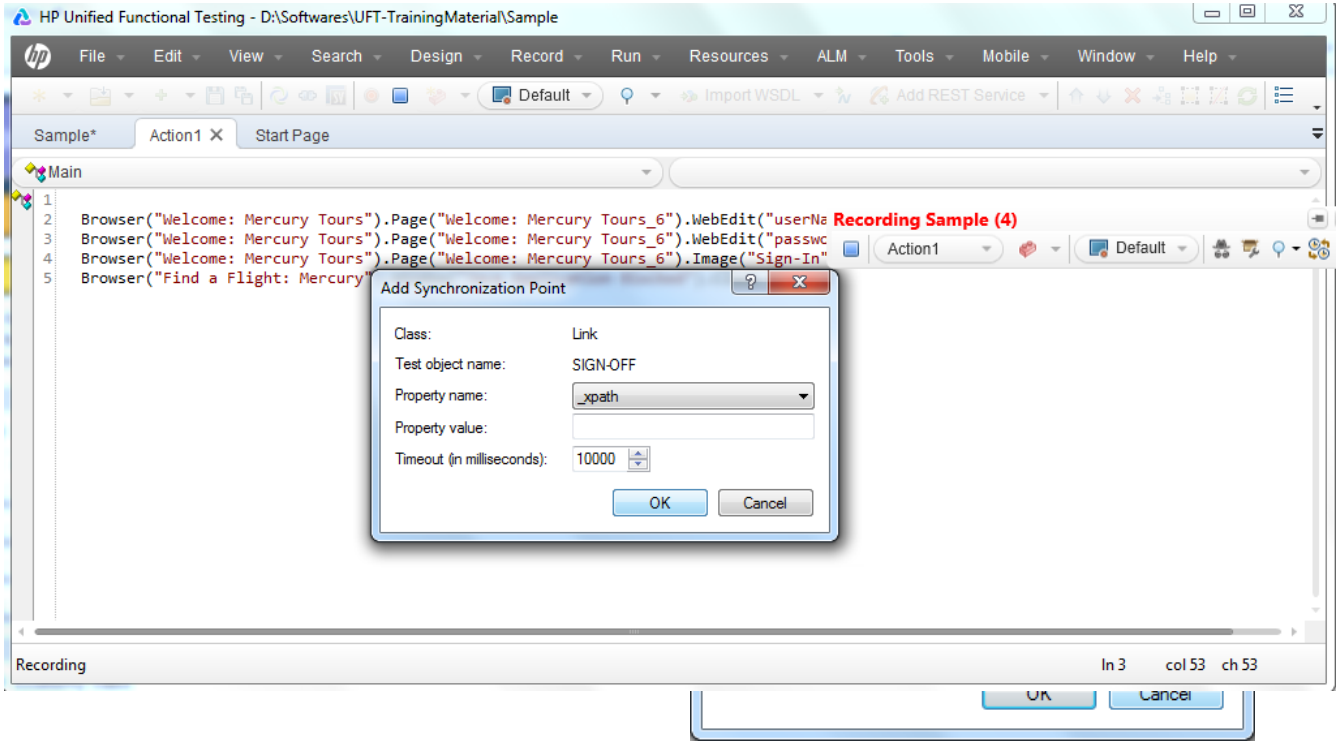
Insert while recording by selecting Design → Synchronization point

Click on the control to be synchronized.

Add Synchronization Point dialog is displayed.

Code snippet

```
Browser("Find a Flight: Mercury").Page("Find a Flight: Mercury_5").Link("SIGN-OFF").WaitProperty "_xpath", '', 10000
```



Synchronization in Expert View

Synchronization – Using Wait

The timing problem can be handled by adding a Wait statement in the script instead of inserting a synchronization point.

Consider the same script, a wait statement is included to instruct the tool to wait for 2 seconds.

Example

```
Browser("Welcome: Mercury Tours").Page("Welcome: Mercury Tours").Image("Sign-In").Click

Wait (2)

Browser("Find a Flight: Mercury").Page("Find a Flight: Mercury").Link("SIGN-OFF").Click
```

Synchronization in Expert View (continue..)

Synchronization - Using Exist

Using this function we can check for the existence of an object or a window and continue with the script based on the result.

Syntax: Object.Exist(Timeout in seconds)

Object can be any GUI object or Window

The function returns a Boolean value. True value is returned in case object exists else False is returned.

Time Out – time for which the object's existence should be checked

Example:

```
If Window("Flight Reservation").WinButton("Update Order").Exist(10) Then

        ……..

End if
```

Transactions

A transaction represents the process in your application that we are interested in measuring. By defining a transaction we can measure how long it takes to run a section of a test script.

Need for Transactions:

Transactions can be used to measure the performance of the script
By analyzing the output of the Transaction we can optimize the script in certain areas

Defining a transaction:

You define transactions within your test by enclosing the appropriate sections of the test with start and end transaction statements. During the test run, the "StartTransaction" step signals the beginning of the time measurement.
The time measurement continues until the "EndTransaction" step is reached.

Creating Transaction

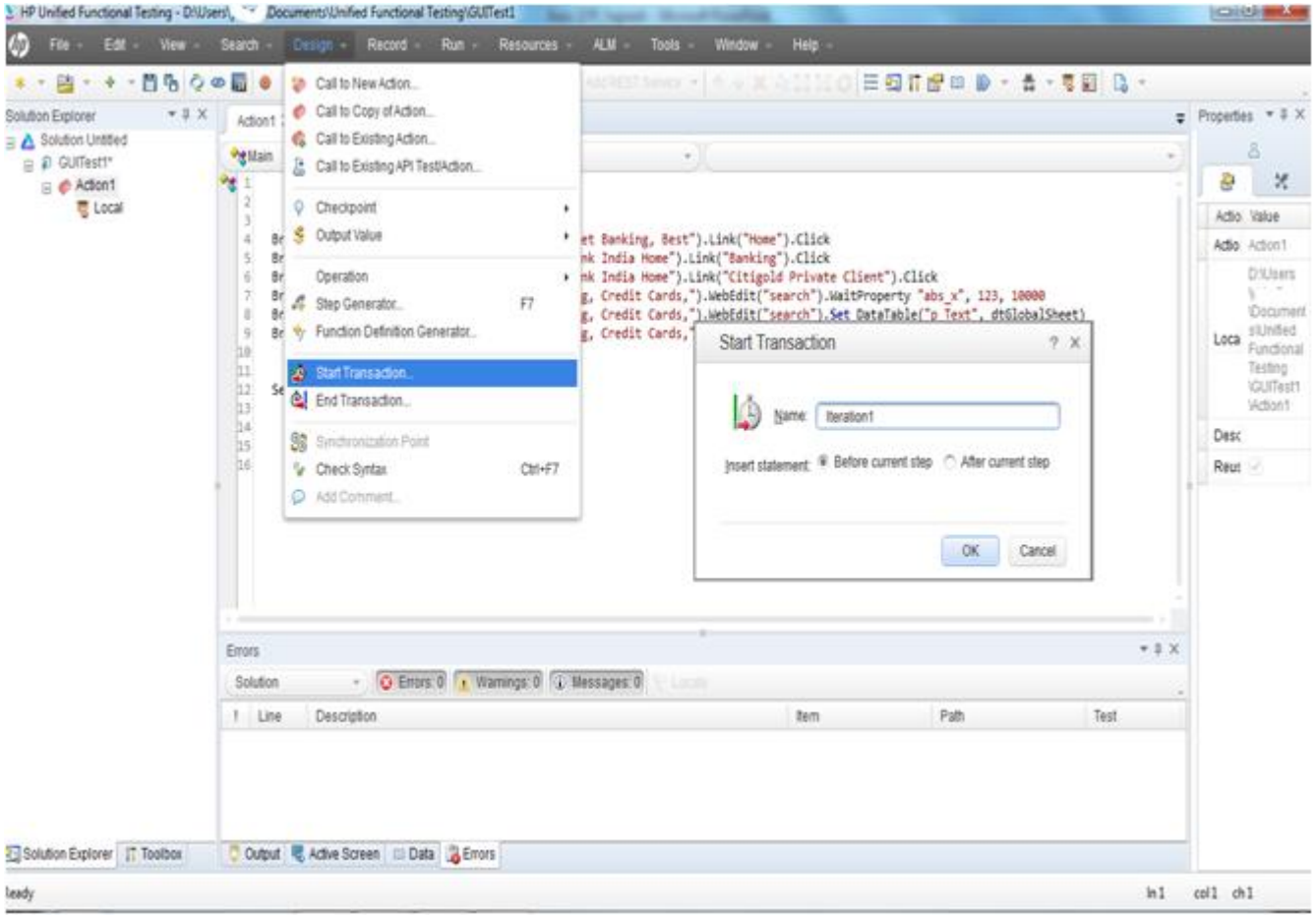
Select Insert->Start Transaction menu.

Start Transaction dialog is displayed. Enter valid name for the transaction.

Click OK button

Code snippet:

Services.StartTransaction "Iteration1" "



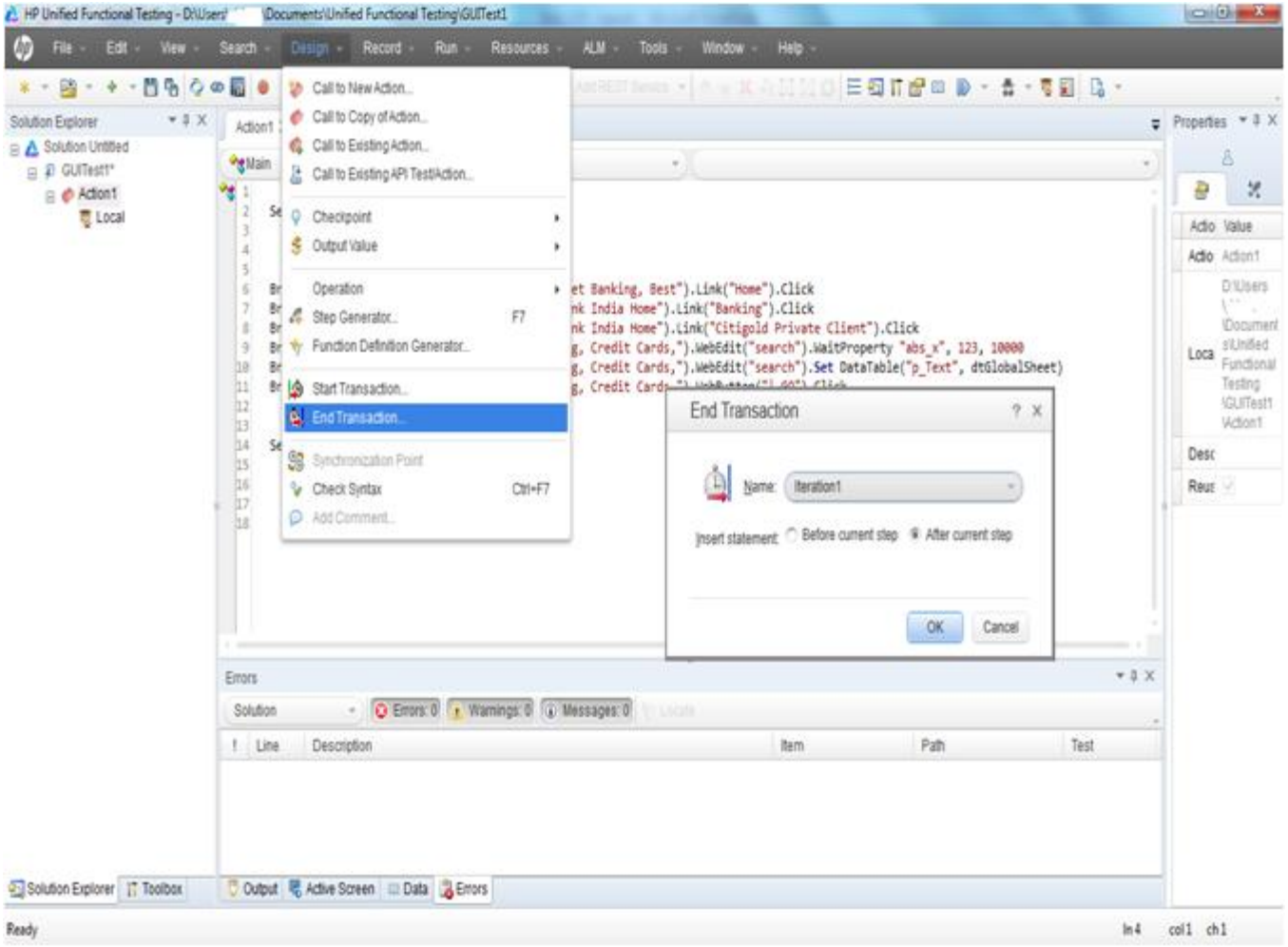
End Transaction

Select Insert->End Transaction menu.

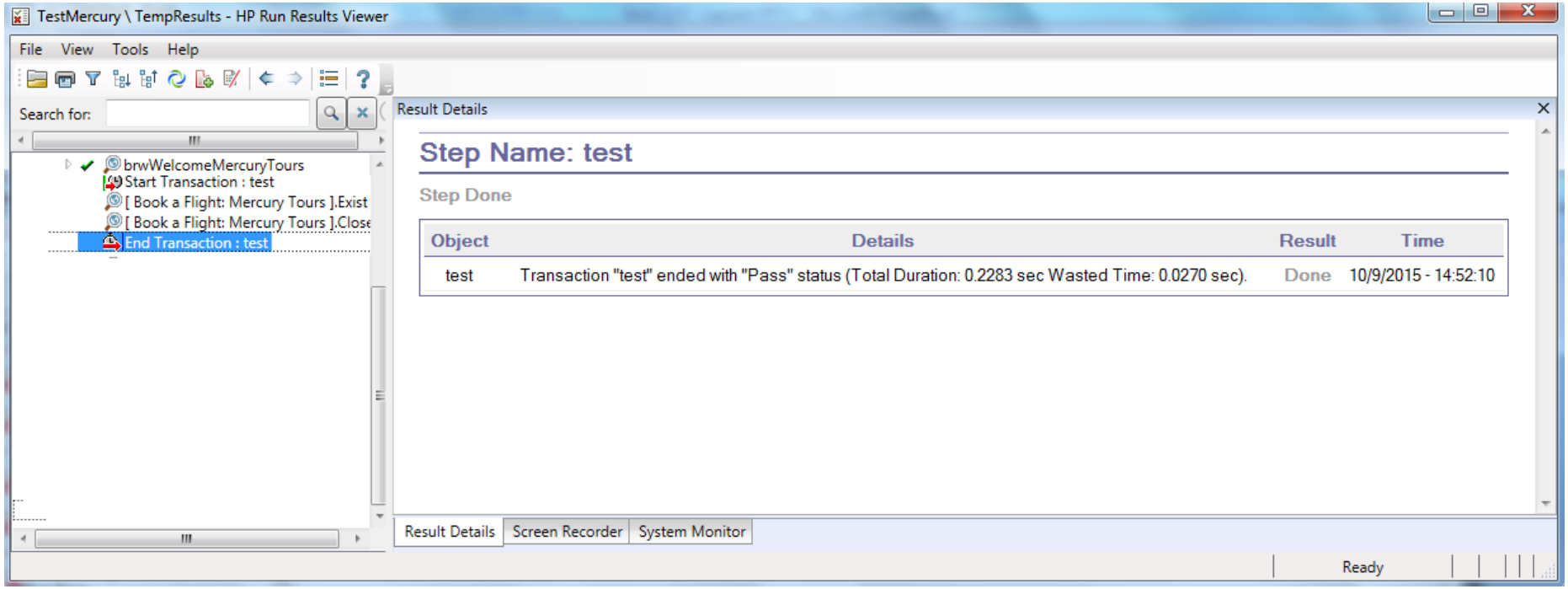
End Transaction dialog is displayed. Select the transaction.

Click OK button

Code snippet:
Services.EndTransaction "Iteration1"



Transaction execution in Result pane



Introduction to Checkpoints

It is a step in UFT that compares two values and then results are reported.

A checkpoint in UFT is a verification point that will compare the current value for a property with the expected value for that property.

It also compares the actual and expected results and if the two values passes, the checkpoint pass else if mismatch checkpoint fails.

Types of Checkpoint

Check Point Type	Description	Example
Standard Checkpoint	Checks property values of an object's properties	Check that a radio button is selected.
Table Checkpoint	Checks information in a table	Check that the value in a table cell is correct.
Page checkpoint	Checks the characteristics of a Web page	Check how long a Web page takes to load or if a Web page contains broken links.
Text / Text Area Checkpoint	Checks that a text string is displayed in the appropriate place in a Web page or application window	Check whether the expected text string is displayed in the expected location on a Web page or dialog box.
Accessibility Checkpoint	Checks compliance with World Wide Web Consortium (W3C) instructions and guidelines for Web-based technology and information systems	Check if the images on the web page include ALT properties required by the W3C Web content Accessibility Guidelines.

Types of Checkpoint (continued)

Check Point Type	Description	Example
Bitmap Checkpoint	Checks the bitmap of an image or a full web page. It does a pixel by pixel comparison between actual and expected bitmaps.	Check that a Web page (or any portion of it) is displayed as expected.
Database Checkpoint	Checks the contents of databases accessed by an application or Web site	Check that the value in a database query is correct.
XML Checkpoint	Checks the data content of XML documents	XML file checkpoints are used to check a specified XML file; XML application checkpoints are used to check an XML document within a Web page.

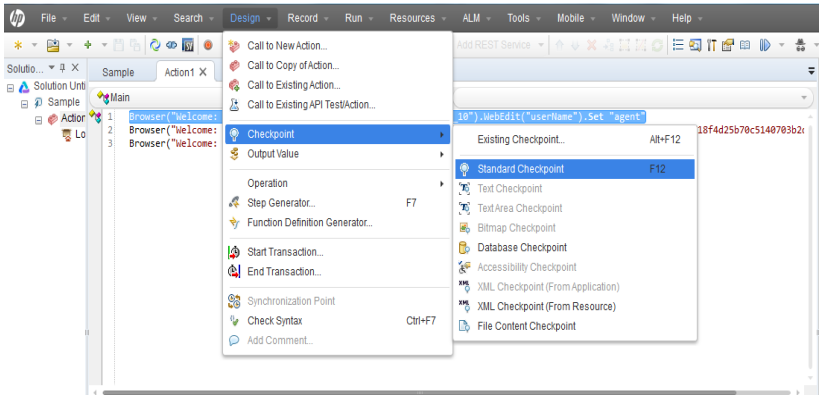
Inserting a Checkpoint

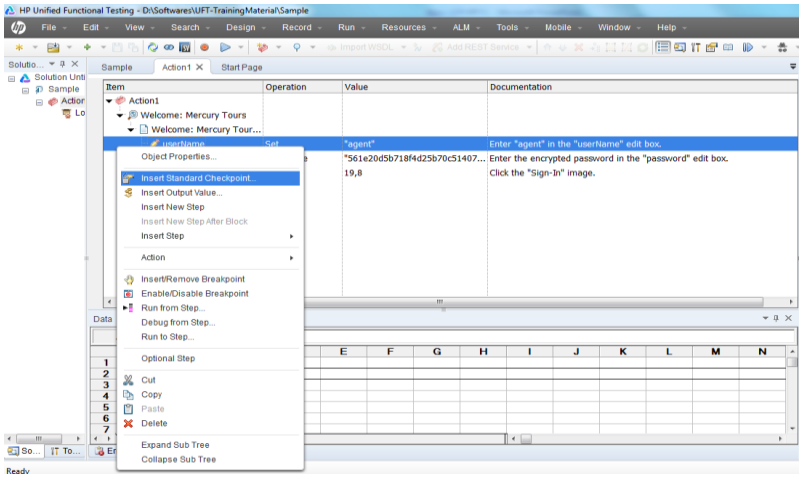
The checkpoint can be inserted by the followings ways:

Select Design Menu → Select Checkpoint and select the type of checkpoint that needs to be added.

Or

In the Keyword view, Right click on the step where the checkpoint is required and select Insert standard checkpoint

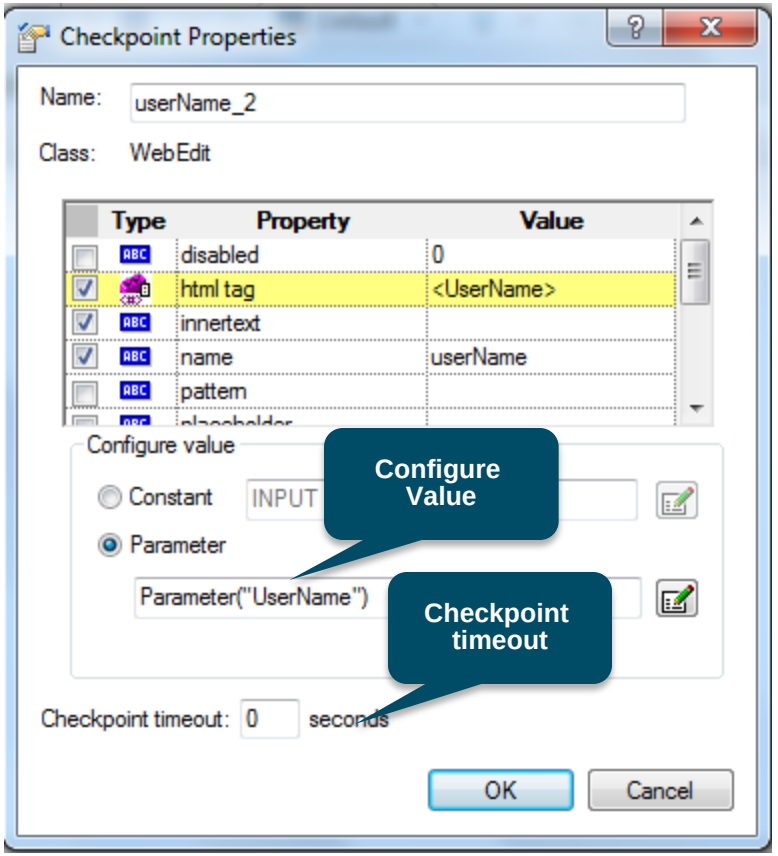




Checkpoint properties dialog

Configure Value→ The value for each property can be configured. The values can be constant or parameterized.

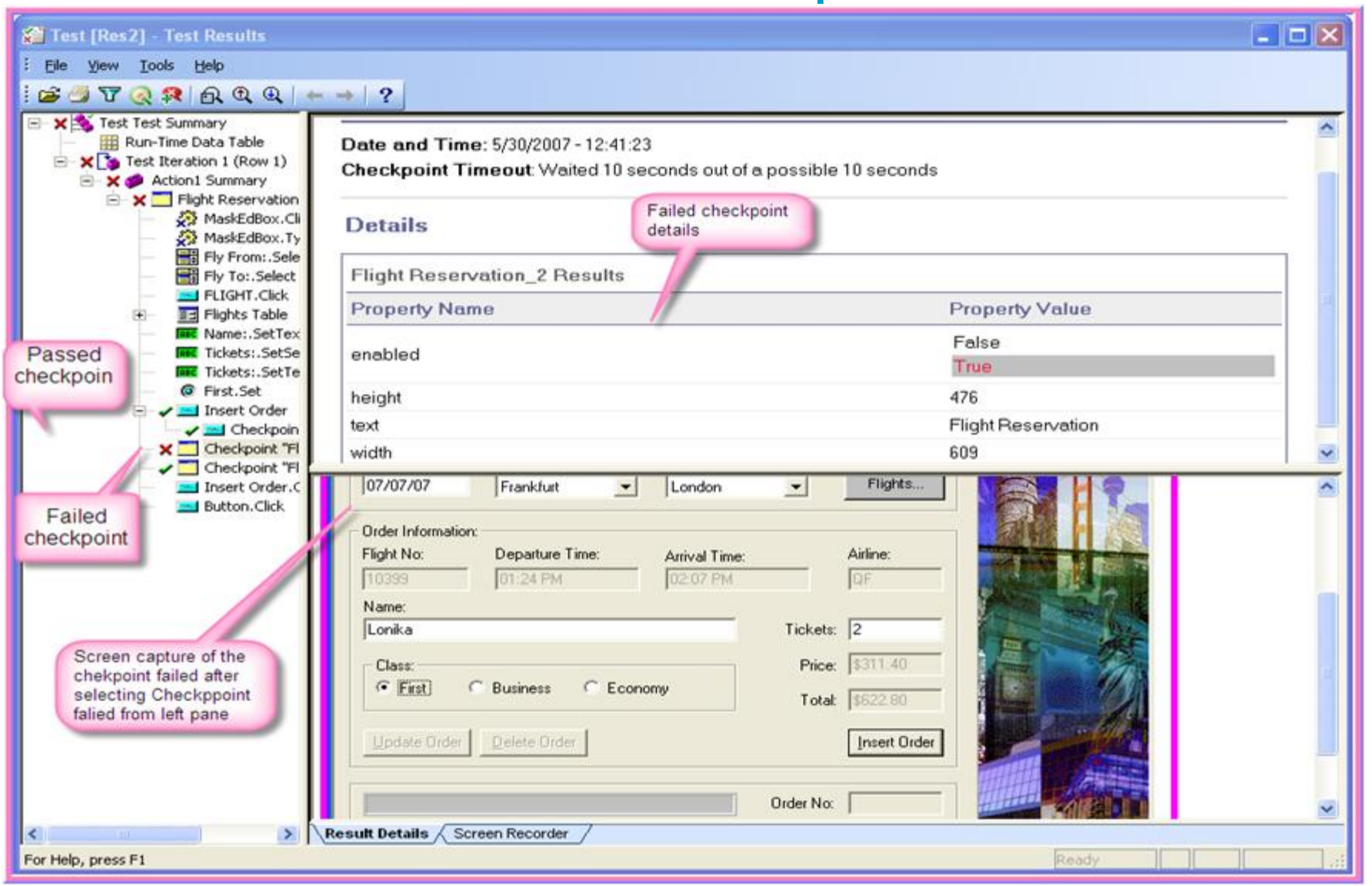
Checkpoint timeout → specifies the time interval during which UFT attempts to perform the checkpoint successfully.



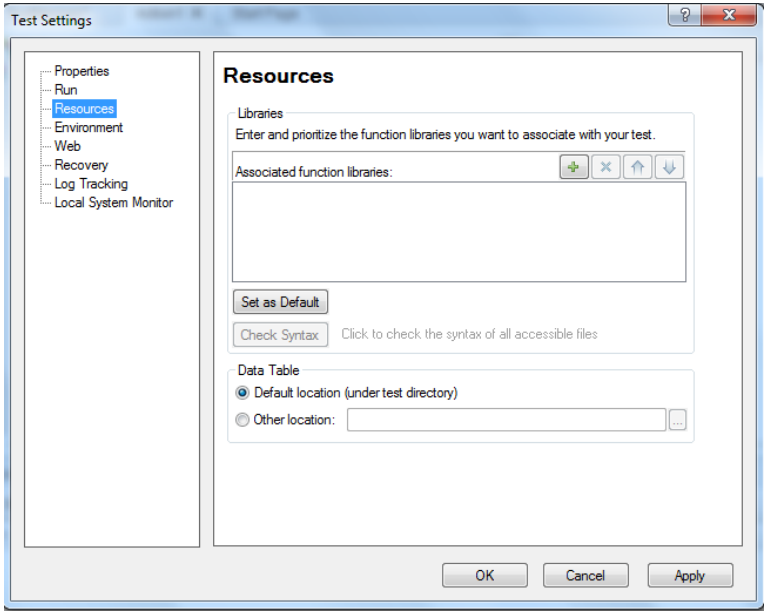
Addition of properties on Checkpoint Properties dialog

1. Select Insert → Standard Checkpoint menu.
2. The checkpoint properties dialog box opens
3. If a Checkpoint needs to be placed for a property. Check that property in the Checkpoint Property window.
4. Modify the expected values as required and Click Ok button to add the checkpoint in the script.

Results of Standard Checkpoint

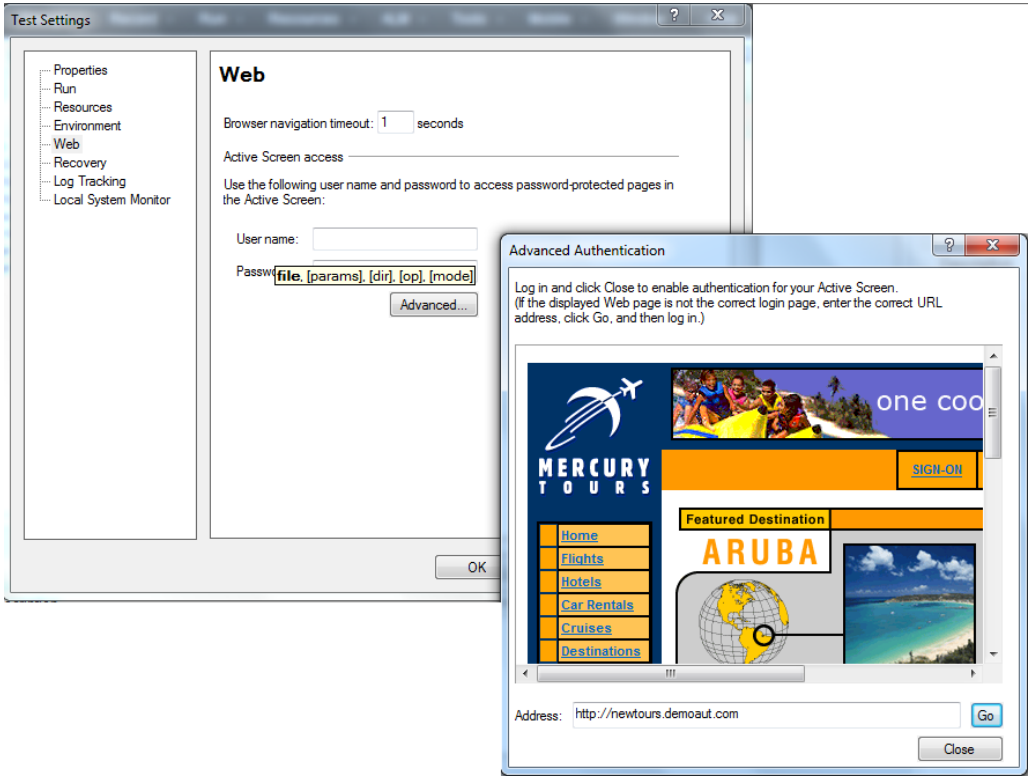


Test Settings - Resources tab



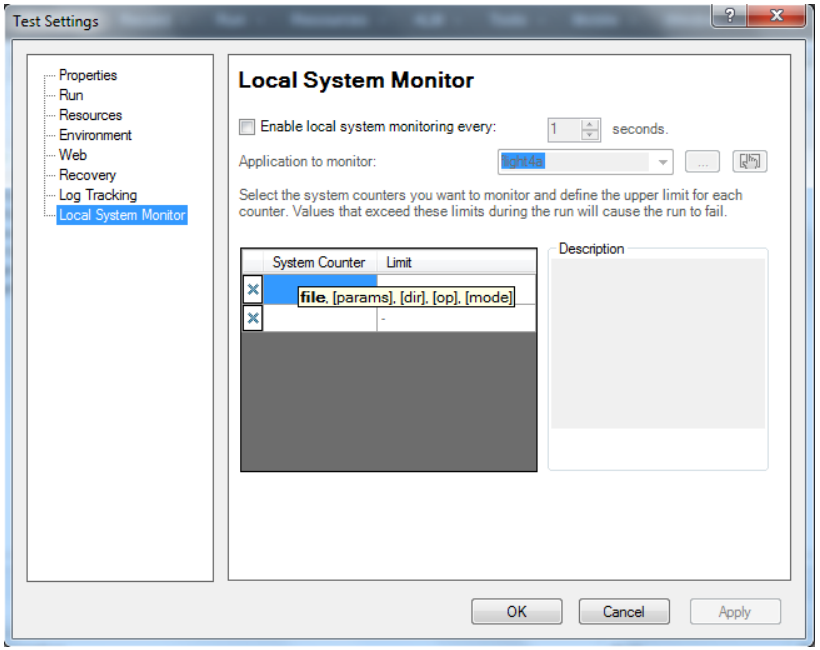
This pane displays the list of resources associated with your test or component (component associations are made via the component's application area).

Test Settings - Web tab



On this tab, we can set how long to wait for browser navigations and can also specify the Active Screen access information to use with password-protected resources in the captured Active Screen page.

Test Settings - Local System Monitor

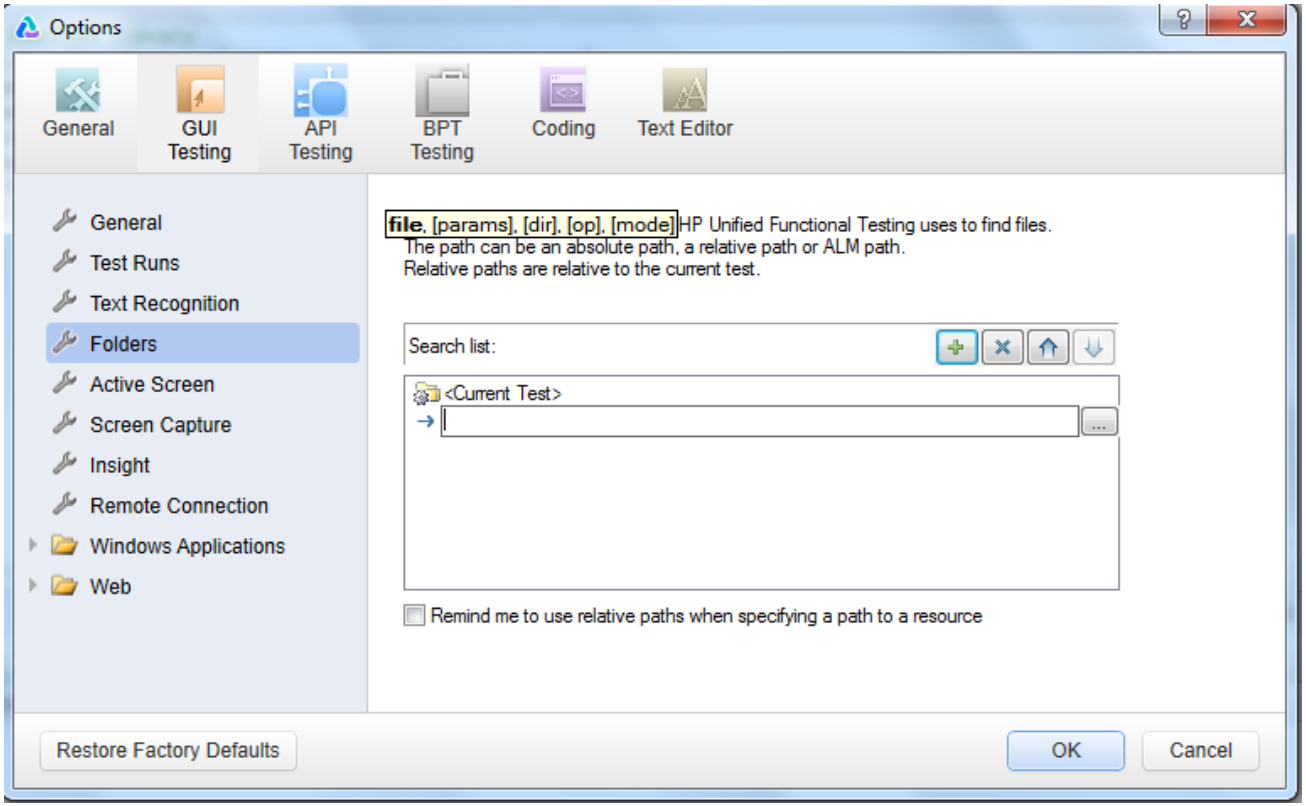


LSM enables to activate system monitoring and we can define the system counters to be tracked during a run session.

The Local System Monitor data that is captured during a test run is displayed in the Test Results window

Tools Options - Folder tab

The Folders pane enables to enter the folders (search paths) in which UFT searches for tests, components, actions, or resource files that are specified as relative paths in dialog boxes and steps. If the same file name exists in more than one folder, UFT uses the first instance it finds.



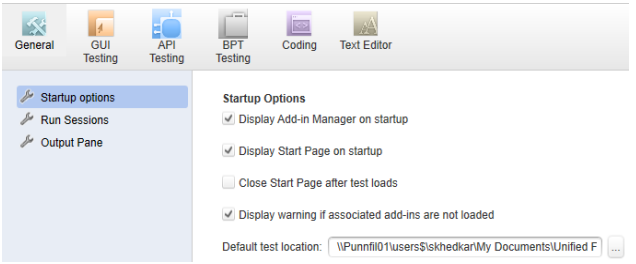
Tools Option - General tab

➢General tab

This tab enables you to define general options for UFT.

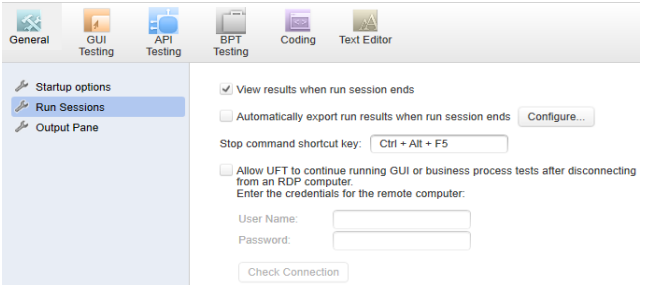
➢General tab > Startup Options

This pane enables you to select what to show when UFT opens.



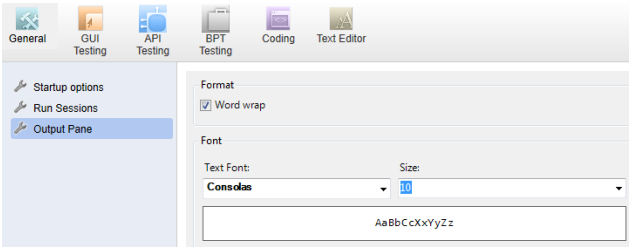
➢General tab > Run Sessions

This pane enables you to determine global UFT run settings that are applicable to both GUI testing and API testing.



➢General tab > Output Pane

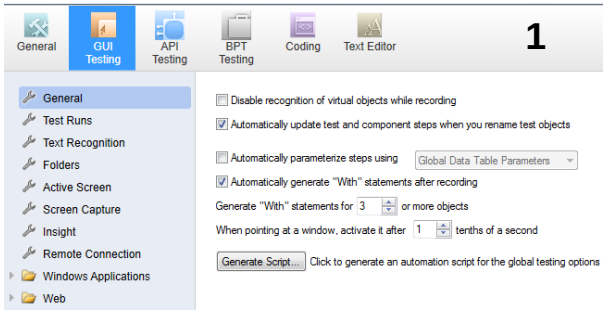
This pane enables you to define display options for the Output pane.



Tools Option - GUI Testing tab

➢GUI Testing tab

This tab enables you to modify global testing options for GUI tests and components.

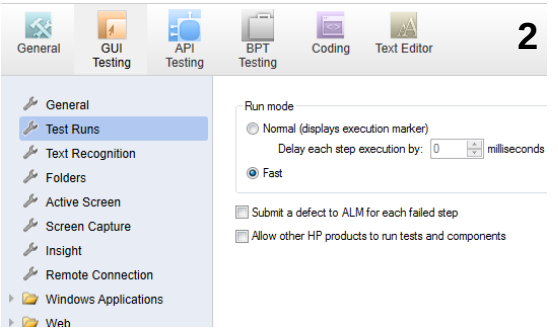


➢GUI Testing tab > General

This pane contains general options for working with GUI tests and components.

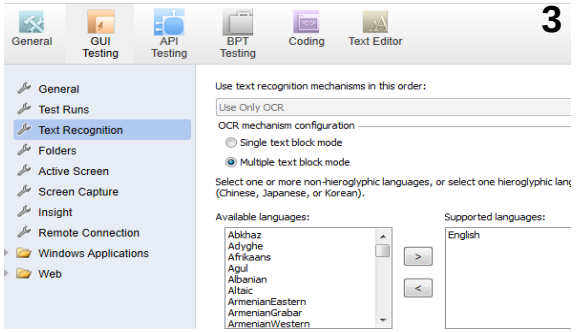
➢GUI Testing tab > Test Runs

This pane enables you to determine how UFT runs GUI tests and components



➢GUI Testing tab > Text Recognition

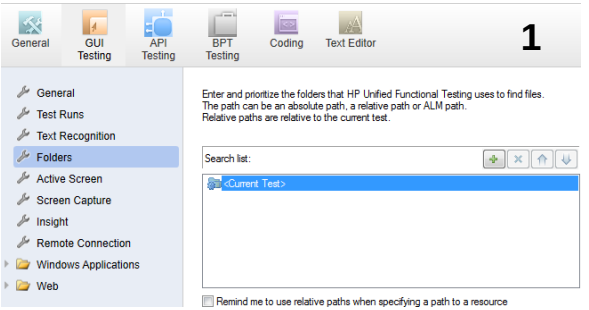
This pane enables you to configure how UFT identifies text in your application. You can use this pane to modify the default text capture mechanism, OCR (optical character recognition) mechanism mode, and the language dictionaries the OCR mechanism uses to identify text.



Tools Option - GUI Testing tab

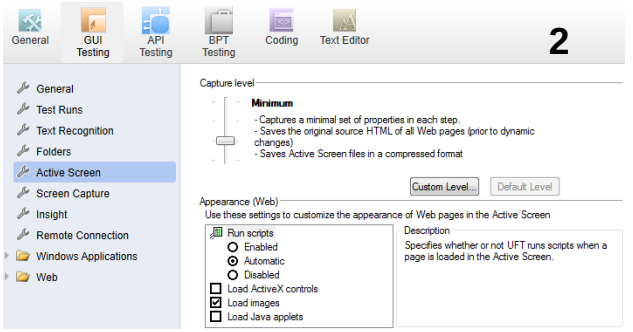
➢GUI Testing tab > Folders

This pane enables you to enter the folders (search paths) in which UFT searches for tests, components, actions, or resource files that are specified as relative paths in dialog boxes and steps.



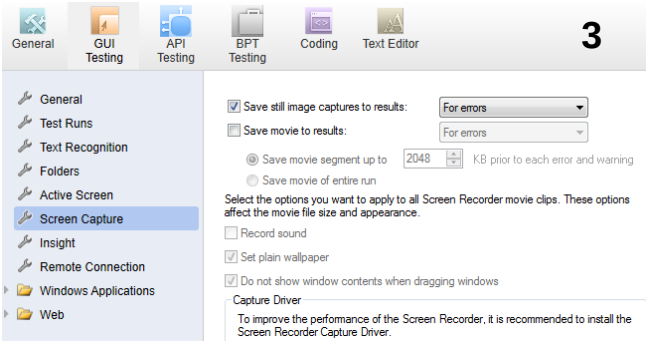
➢GUI Testing tab > Active Screen

This pane enables you to specify which information UFT saves and displays in the Active Screen while recording and running tests.



➢GUI Testing tab > Screen Capture

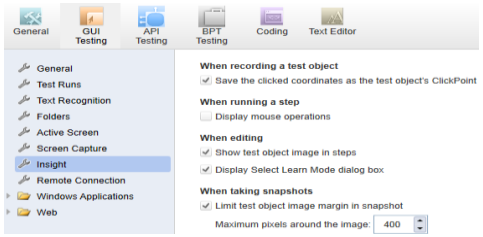
This pane enables you to control how UFT captures images and movies of the application being tested.



Tools Option - GUI Testing tab

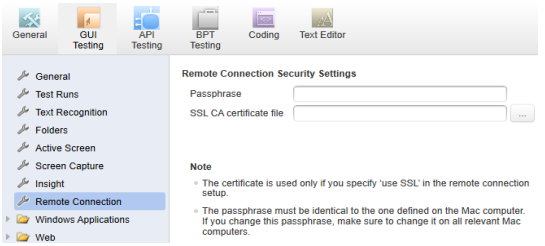
➢GUI Testing tab > Insight

This pane enables you to define options that customize how UFT handles Insight test objects when creating test object, and during record and run sessions.



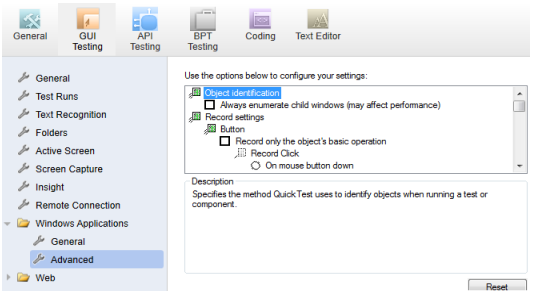
➢GUI Testing tab >Remote Connection

This pane enables you to define options that control the security of UFT's communications over remote connections. This is relevant when setting up a remote connection to a Mac computer during a run session, or when using the Remote Connection button in the UFT toolbar.



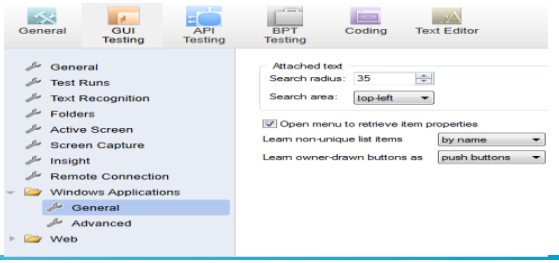
➢GUI Testing tab > Windows Applications > General

This pane enables you to configure how UFT records and runs tests and business components for Windows-based applications.



➢GUI Testing tab > Windows Applications > Advanced

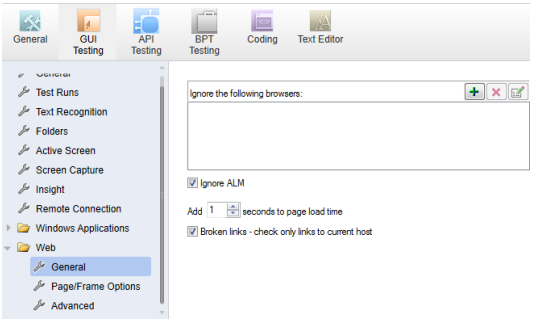
This pane enables you to modify how UFT records and runs tests or business components on Windows-based applications.



Tools Option - GUI Testing tab

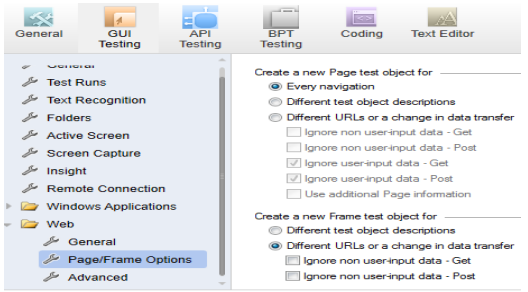
➢GUI Testing tab > Web > General

This pane enables you to determine how UFT behaves when recording and running tests or business components on Web sites.



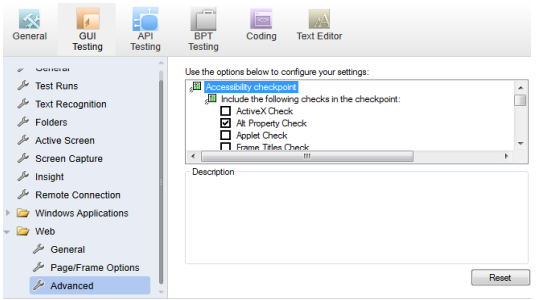
➢GUI Testing tab > Web > Page/Frame Options

This pane enables you to modify how UFT records Page and Frame objects.



➢GUI Testing tab > Web > Advanced

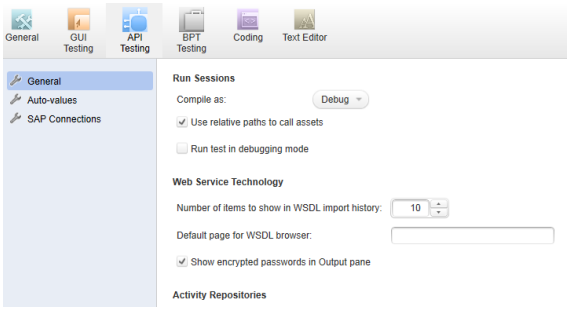
This pane enables you to modify how UFT records and runs tests and business components on Web sites.



Tools Option – API Testing tab

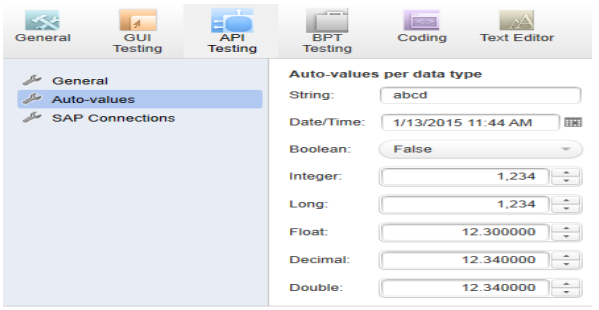
➢API Testing tab >General

This pane enables you to set the test execution mode, the default location for importing WSDL files, and the repository locations for activity sharing.



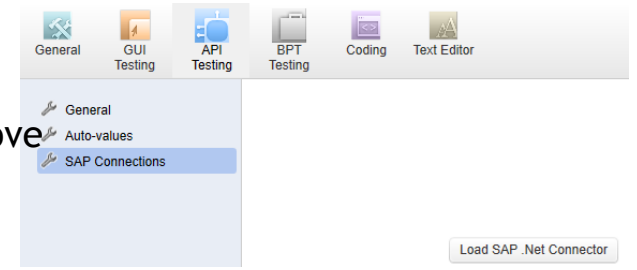
➢API Testing tab >Auto-Values

This pane enables you to provide default values for populating activity properties.

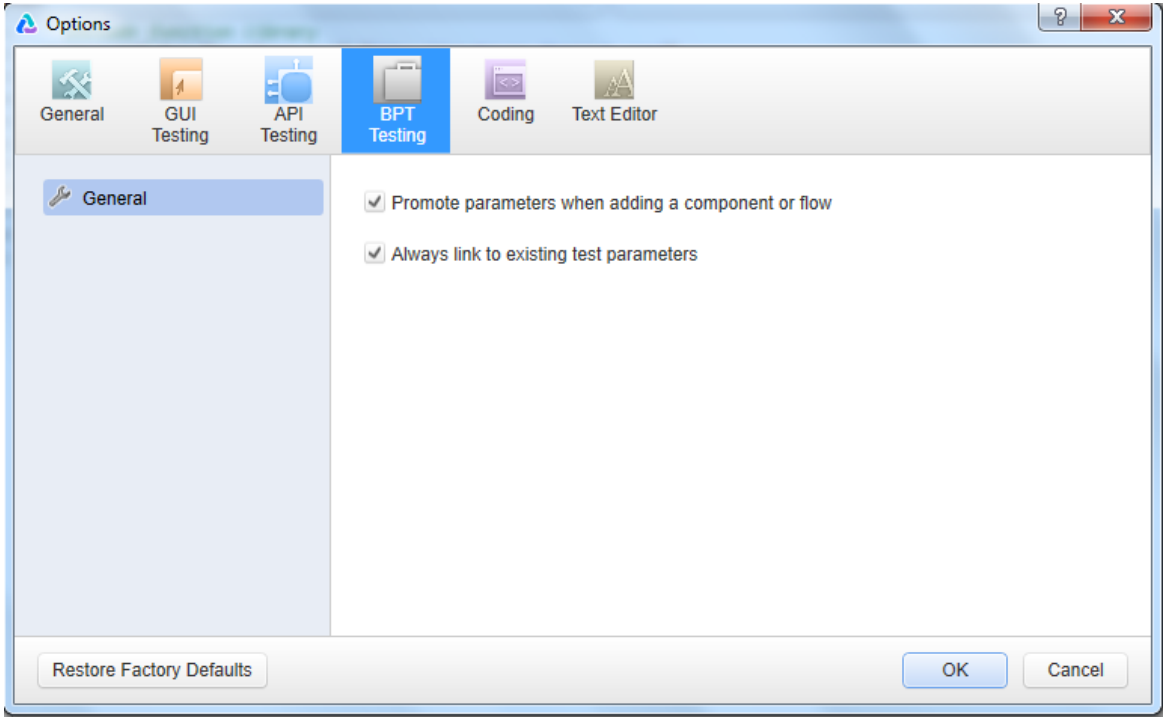


➢API Testing tab >SAP Connections

This pane enables you to manage SAP connections. You can add and remove connections, add authentication information, and check the connection.



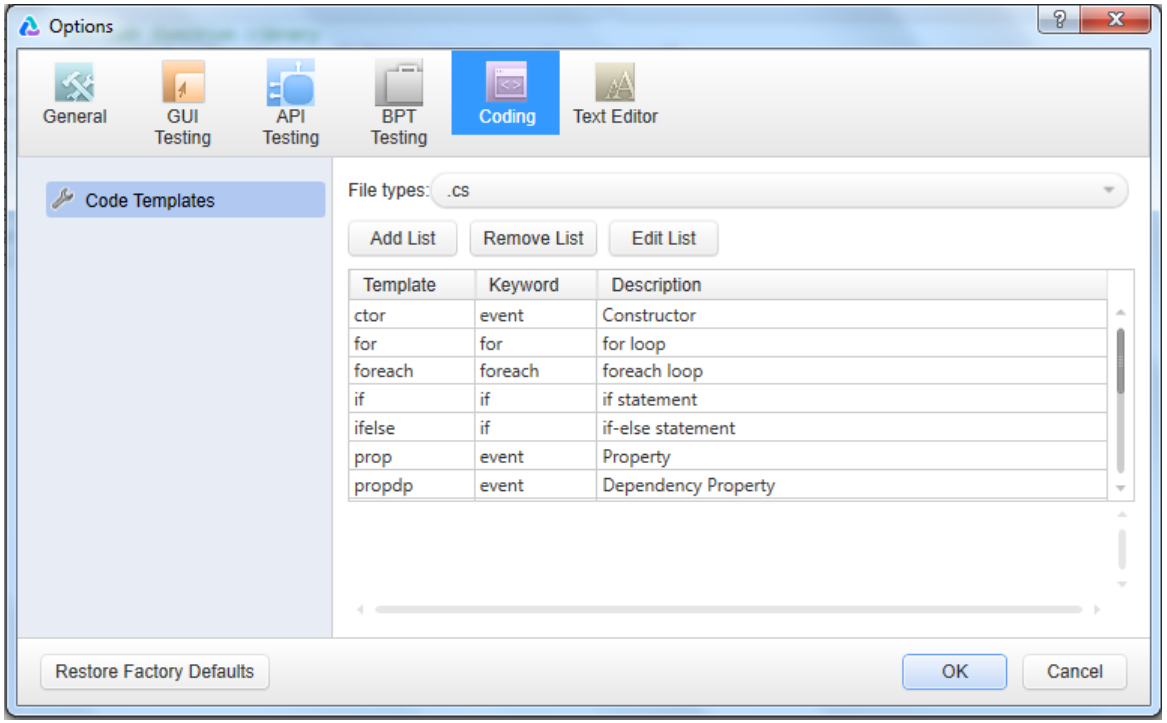
Tools Option - BPT Testing tab



➢Options > BPT Testing

This tab enables you to set options for parameter use of a BPT test and its components in UFT.

Tools Option - Coding tab



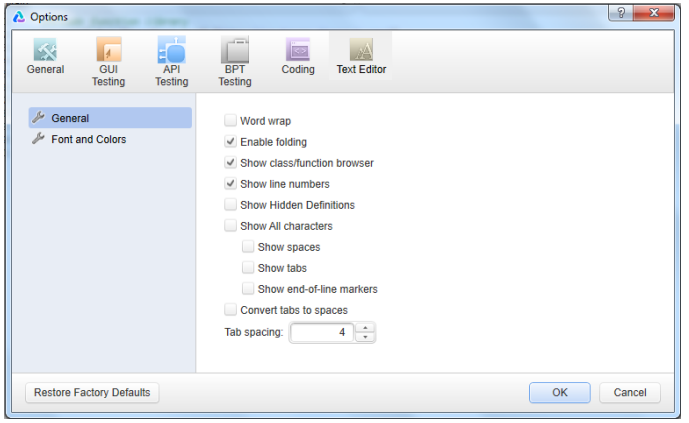
➢Options > Coding

This pane enables you to create and edit templates of pre-designed code snippets or blocks of text used for automatic code completion. You can create different lists of templates for different file types.

Tools Option - Text tab

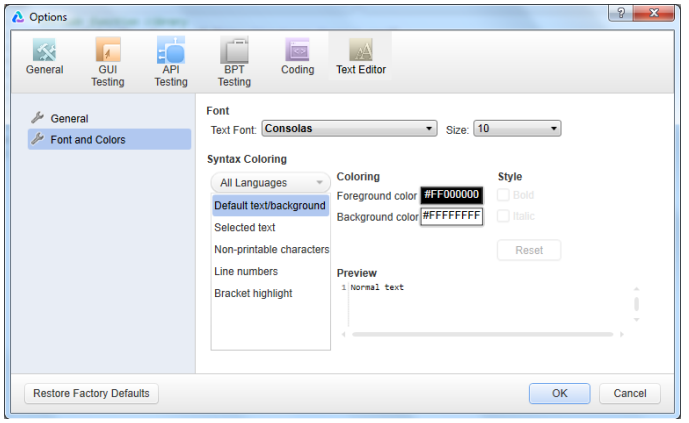
➢Text Editor tab > General

This pane enables you to set general preferences for editing code and text files.



➢Text Editor tab > Fonts and Colors

This pane enables you to specify color preferences for the text you are editing. You can set unique coloring rules for each code element.



UFT (HP Unified Functional Testing)

HP Unified Functional Testing - It is combination of UFT (Unified Functional Testing) and Service Test (ST)/API Tools. Using this tool we can automate GUI, API and Mobile applications functionality.

New Features :-

- It supports to works with multiple tests at the same time.
- It supports to works for edit more than one action at the same time.
- File content Checkpoint for checking file content
- File content output value for capturing file content
- GUI Support for QT and adobe Flex applications

UFT (HP Unified Functional Testing) Continue…..

Advantages of UFT over QTP: -

- QTP only for Functional and regression testing of GUI (Web application and Window based) application.
- UFT is combination of QTP and Service tools , it supports GUI and API Testing.
- Image and control based test recording.
- Extended support for mobile testing, write anagogic script, which once written can be used on multiple devices and test our scripts or simulators as well as real devices.

10/19/2015 1:00 PM© 2007 Capgemini - All rights reserved
77

Thank You ☺