HP Unified Functional Testing 11.5 - It is combination of QTP (Quick Test Professional) and Service Test (ST) Tools. Using this tool we can automate GUI, API and Mobile applications functionality.

**Advantages of UFT over QTP**  
  
•    QTP only for Functional and Regression Testing of GUI( Windows based and Web) Applications.  
•    UFT is Combination of QTP and Service Tools, It supports GUI and API Testing.  
•    UFT has brand New IDE as per latest trends in IT Industry  
•    Image / Control based Test Recording   
•    Extended Support for Mobile Testing, write agnostic scripts, which once written can be used on multiple devices and test our scripts on simulators as well as real devices.

UFT is an automation functional testing tool originally from Mercury Interactive which was acquired by Hewlett Packard (HP) in 2006. UFT is primarily used for functional, regression and service testing. Using UFT, you can automate user actions on a web or client based computer application and test the same actions for different users, different data set, on various Windows operating systems and/or different browsers. Automation using UFT if planned and executed in a proper manner can save considerable time and money.

HP Unified Functional Testing uses VB Script as its scripting language. This is the only language that is fully supported by UFT’s IDE. VB Script supports Object Oriented Programming concepts but not polymorphism and inheritance.

Supported Browsers

* **Browsers supported by QTP**Internet Explorer 6, 7, 8, 9, 10, 11, Edge ([with appropriate patches and service packs](https://www.learnqtp.com/uft-qtp-browser-support-matrix/))
* Firefox 3.0.X, 3.5, 3.6 to 56 ([with appropriate patches and service packs](https://www.learnqtp.com/uft-qtp-browser-support-matrix/))
* Google Chrome till version 61 ([with appropriate patches and service packs](https://www.learnqtp.com/uft-qtp-browser-support-matrix/))
* [Safari on Mac OS](https://www.learnqtp.com/uft-12-new-features-in-detail/) v6 – v9 (in Beta mode)

A Raw Example Where UFT can be Used

Let’s consider the scenario where you want to test login for a particular user. To test this scenario manually, you would generally follow these steps –

1. Open web browser
2. Enter the URL of the application.
3. Enter user id & password and then click on “Login” button on login page.
4. Verify that login is successful

Now consider a case where you have to test this login functionality many a times or you have to test the same functionality for large number of different users say a 1000 or more users. Performing this action manually is time consuming & tiresome activity. Now UFT helps you replicate your actions that you do manually such as opening the application, entering user id and password and hitting the “Login” or “Submit” button and verifying whether or not you have logged in successfully. To achieve this, you can write your code for log in scenario in UFT and run the code any number of times with a single click of a button, of course with same or different test data as required.

**What is the Quick Test Professional (QTP) testing process?**

**For any automated tool implementation, the following are the phases/stages of it**. Each one of the stages corresponds to a particular activity and each phase has a definite outcome.

1. Test Automation Feasibility Analysis
2. Appropriate Tool Selection
3. Evaluate the suitable framework
4. Build the Proof of Concept
5. Develop Automation Framework
6. Develop Test Script, Execute and Analyze

**The Quick Test testing process consists of 6 main phases:**

**1. Create your test plan**  
Prior to automating there should be a detailed description of the test including the exact steps to follow, data to be input, and all items to be verified by the test. The verification information should include both data validations and existence or state verifications of objects in the application.

**2. Recording a session on your application**  
As you navigate through your application, QuickTest graphically displays each step you perform in the form of a collapsible icon-based test tree. A step is any user action that causes or makes a change in your site, such as clicking a link or image, or entering data in a form.

**Enhancing your test**  
o Inserting checkpoints into your test lets you search for a specific value of a page, object or text string, which helps you identify whether or not your application is functioning correctly.  
NOTE: Checkpoints can be added to a test as you record it or after the fact via the Active Screen. It is much easier and faster to add the checkpoints during the recording process.  
o Broadening the scope of your test by replacing fixed values with parameters lets you check how your application performs the same operations with multiple sets of data.  
o Adding logic and conditional statements to your test enables you to add sophisticated checks to your test.

**3. Debugging your test**  
If changes were made to the script, you need to debug it to check that it operates smoothly and without interruption.

**4. Running your test on a new version of your application**  
You run a test to check th3e behavior of your application. While running, QuickTest connects to your application and performs each step in your test.

**5. Analyzing the test results**  
You examine the test results to pinpoint defects in your application.

**6. Reporting defects**  
As you encounter failures in the application when analyzing test results, you will create defect reports in Defect Reporting Tool.

# Creating Automation Frameworks with QTP

In this tutorial, we will use QTP (HP Functional Test) to create

* [Data Driven Framework](https://www.guru99.com/creating-automation-frameworks-with-qtp.html#1)
* [Keyword Driven Framework](https://www.guru99.com/creating-automation-frameworks-with-qtp.html#2)
* [Hybrid Framework](https://www.guru99.com/creating-automation-frameworks-with-qtp.html#3)

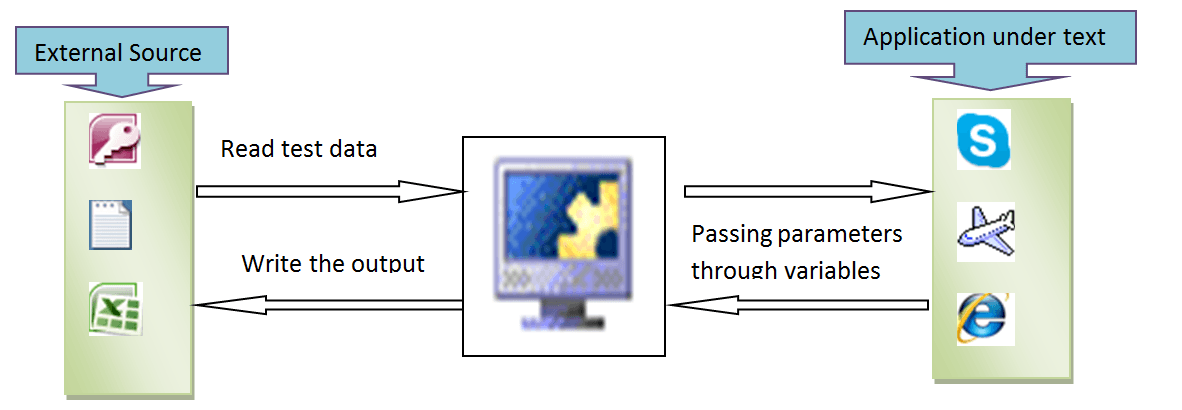
So let’s get started -

## Data Driven Framework

Data Driven Framework is a framework which is driven by the different combinations of input and output data.

One way of passing different combinations of data is by Parameterization. In this method, we use different features of QTP. To know more about Parameterization, please visit [here](https://www.guru99.com/quick-test-professional-qtp-tutorial-12.html).

However in DDF, scripts are written to do parameterization. This kind of framework is useful when the AUT’s functionality must be tested with several inputs and capture the respective outputs. These inputs can be read from an external file such as Database, Excel, Outlook, Text file etc and the respective outputs are written back to the corresponding external source.

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_1.png)

The general steps involved in the data driven framework are:

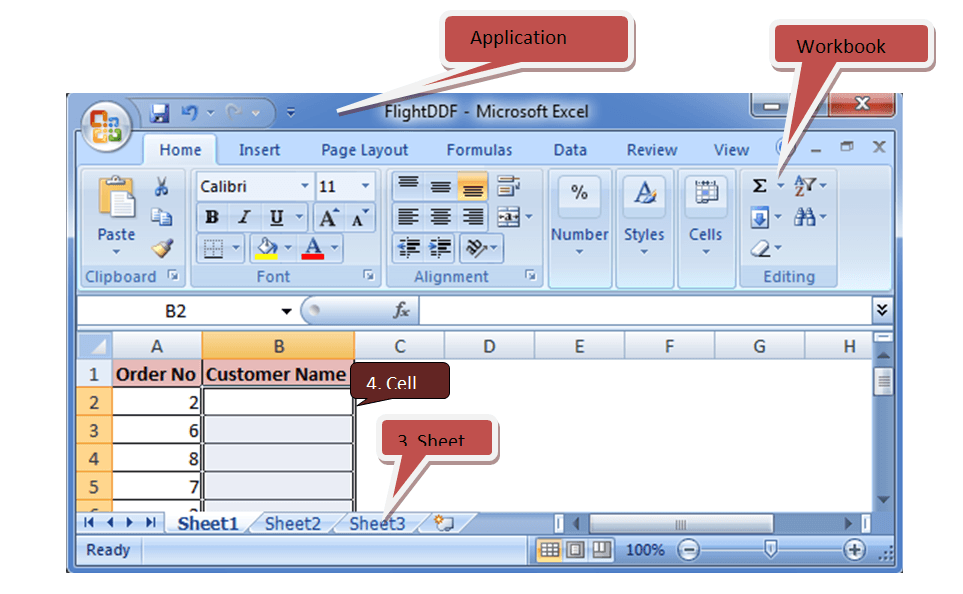
1. Prepare the[Test Case](https://www.guru99.com/test-case.html)for the Application Under Test
2. Add the Objects from AUT to OR
3. Write the scripts based on the Test Case

In this tutorial, we will develop a data driven framework for a sample test case by using Excel as an external source for the Test data.

## Step1) Prepare the Test Case for the Application under Test

**Test Case**: Open the order number and get the customer name for that order. Repeat the same process for different Order numbers

**External Source**: Excel File

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_2.png)

The External source for this sample is an Excel file. The VB script in QTP must be written to open an Excel file in order to read the test data. This can be achieved in a hierarchical manner.

1. An Excel file is first opened an as Application

2. Then the workbook must be opened from the specified location

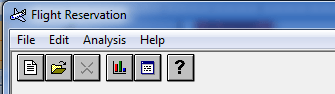
3. The sheet where the test data are present.

4. Finally the cell must be read.

## Step 2)  Add the Objects from AUT to OR

Once the Test case is ready, start adding all the required objects to the repository. In our test case, the objects need to be added are as follows

* “Open Folder” icon in the Flight Reservation application

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_3.png)

* “Order No” Checkbox which can be obtained when the “Open Folder” icon is clicked

[Creating Automation Frameworks with QTP](https://cdn.guru99.com/images/QTP_Article_35/Article_35_4.png)

* The WinEdit box of the Order no (where the numbers are entered)

[Creating Automation Frameworks with QTP](https://cdn.guru99.com/images/QTP_Article_35/Article_35_5.png)

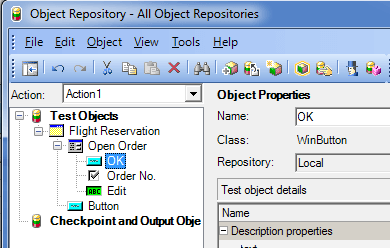
* The OK button

[Creating Automation Frameworks with QTP](https://cdn.guru99.com/images/QTP_Article_35/Article_35_6.png)

* The “Name” field which is a WinEdit box. This field will be populated with a name once the OK button is clicked for a particular Order number.

[Creating Automation Frameworks with QTP](https://cdn.guru99.com/images/QTP_Article_35/Article_35_7.png)

Once all the required objects have been added, the object repository will appear as follows:

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_8.png)

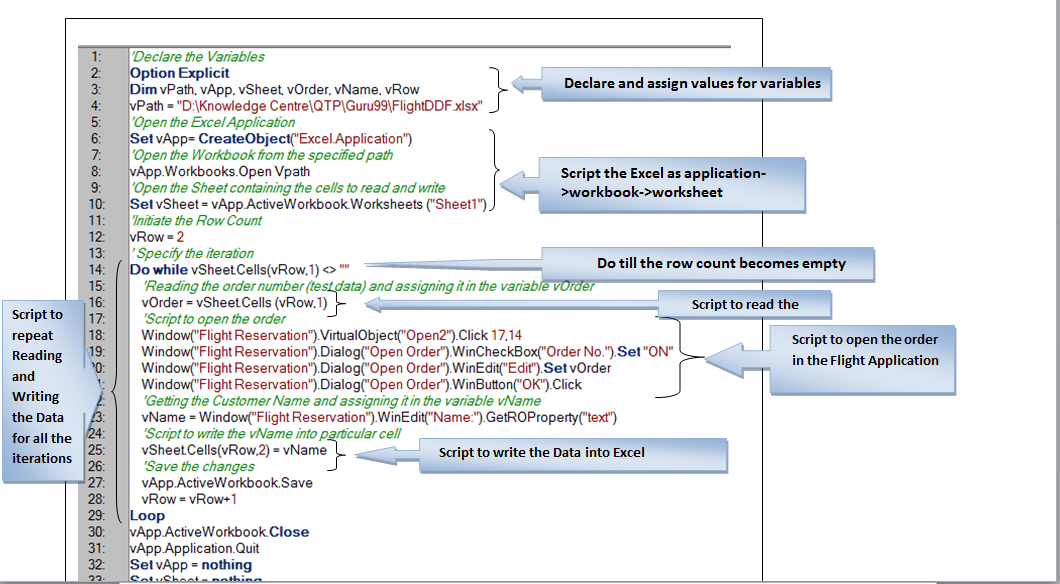
## Step 3)  Write the scripts based on the Test Case

Before running the script, ensure that the Excel file containing the test data has been saved and is closed.

Below script is to read the order number from the Excel and assign it into the application through variable “vOrder” and write the customer name from the application through variable “vName”.

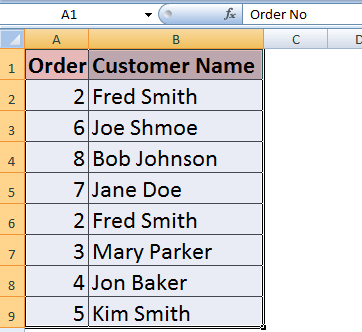
**Excel Name**: FlightDDF.xlsx

**Sheet Name**: Sheet1

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_9.png)

### Output

Once the above script is run, the output can be obtained from the Excel as follows:

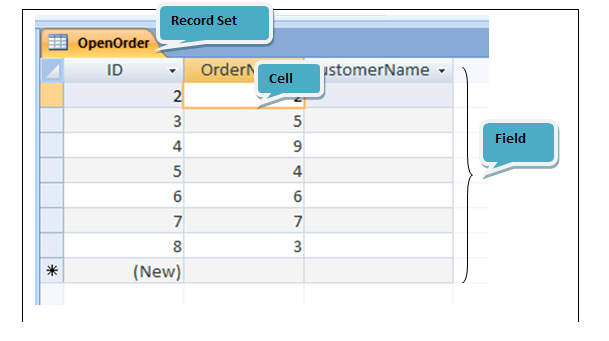
[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_10.png)

The Data Driven Framework can also be developed by writing descriptive programming.

### Using Database as an External Source for DDF

The same test case can be executed if the external source is a Database using following steps

1. Write VbScript  to establish the database connection
2. VBScript to open a record set or a table.
3. VBScript to open desired field
4. The particular cell is read from the field.

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_11.png)

### Script

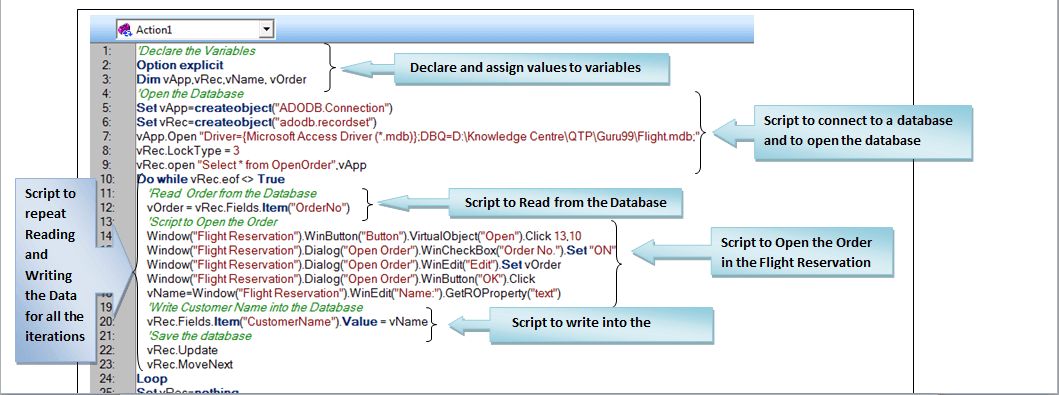
To Establish a Microsoft Database connection

Driver = {Microsoft Access Driver (\*.mdb)}; DBQ = <Path of the database>

**Record Set Name :**OpenOrder

**Fields Name:**OrderNo, CustomerName

P.S: Microsoft Access 2010 cannot be connected using the below script.

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_12.png)

### Output

### [Creating Automation Frameworks with QTP](https://cdn.guru99.com/images/QTP_Article_35/Article_35_13.png)

### Advantages of DDF

* Large number of test data can be read and written in to the external file in a single test
* Loop statement is used to repeat the same steps for several iterations. Hence coding effort is reduced
* Since the data are read and written directly into the external file, there is no need to copy, paste or export data in order to use them
* Test Data can be read from any external file and the outputs can be written into any other external file

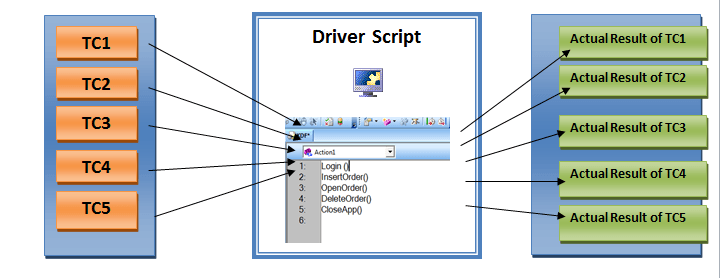
### Disadvantages of DDF

* Scripting knowledge is required to develop this framework
* Sometimes more number or combinations of data from an external source like Database may slow down or even crash the QTP

## Keyword Driven Framework

Keyword Driven Framework is a framework where keywords drive the test. Here **keyword refers to the user defined functions**. In this framework, keywords are created in order to perform a particular test step or a test case. These keywords are then called into the driver test to run several test cases in the same test.

To know about user defined functions in QTP, please visit [here](https://www.guru99.com/quick-test-professional-qtp-tutorial-27.html)

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_14.png)

In general, the frameworks can be developed in three ways in order to run to the test.

1. Record and run the test
2. Add objects to the local repository and write the scripts for all test steps
3. Write descriptive programming for all test steps

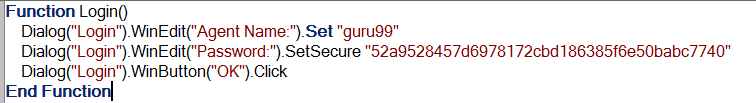
In this tutorial, The KDF is developed by recording and running the test.

Our aim is to run a single test for five different test cases such as login into the application, insert an order, open an order, delete an order and close the application. Hence, we will record the test steps for these test cases and create the functions with keywords Login, InsertOrder, OpenOrder, DeleteOrder and CloseApp respectively.

**Test Case1: Login into the application**

**Keyword:**Login ()

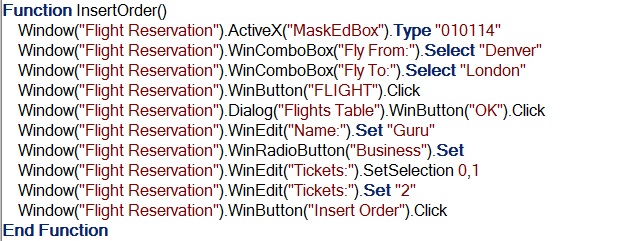
**Recorded Script:**

**[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_15.png)**

**Test Case2: Insert the Order**

**Keyword:**InsertOrder()

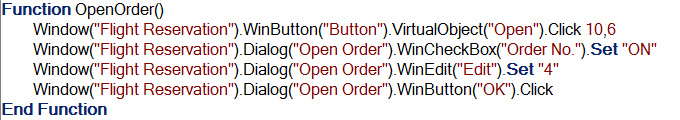
**Recorded Script:**

**[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_16.png)**

**Test Case3: Open the Order**

**Keyword:**OpenOrder()

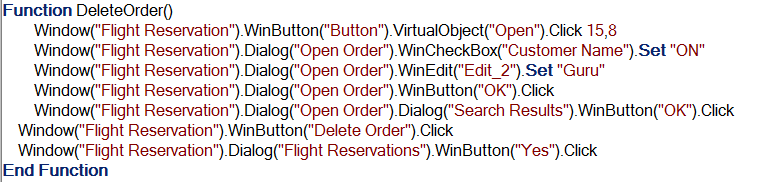
**Recorded Script:**

**[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_17.png)**

**Test Case4: Delete the Order**

**Keyword:**DeleteOrder()

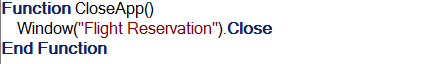
**Recorded Script:**

**[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_18.png)**

**Test Case5: Close the application**

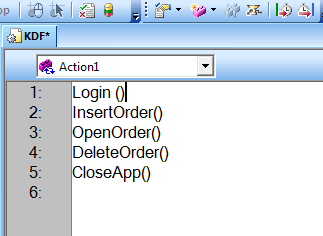
**Keyword:**CloseApp()

**Recorded Script:**

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_19.png)

The functions that are created for different test cases are saved in a function library and are associated with the main test. It is enough to call the keywords for the needed test cases in the main test thereby reducing the size of the driver script in the main test.

The Driver Script for this simple framework looks as:

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_20.png)

By running the above script, the actual result for all the five test cases can be obtained from a single test.

### Advantages

* Any number of test cases can be run on a single test just by calling their respective keywords
* Writing general descriptive programming for all web/ windows objects & calling them as keywords will help in running the same test for different dynamic applications
* Reduces the size of the driver script

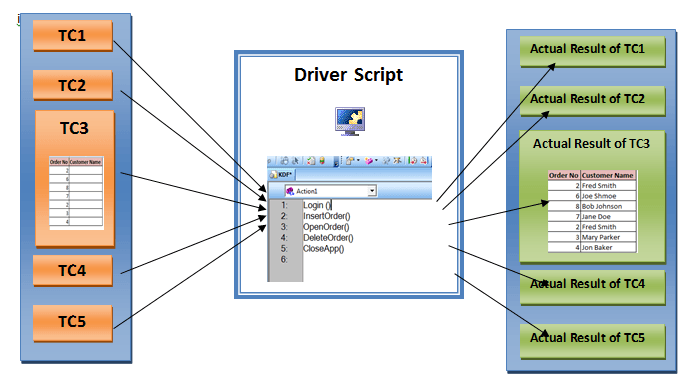
### Disadvantages

* Time taken to develop these frameworks is very high if there are very few number of test cases to run
* Recording the steps is not always used when designing KDF for many applications on a same test.

## Hybrid Framework

Hybrid framework is a combination of Data Driven Framework (DDF) and Keyword Driven Framework (KDF) where several test cases with several inputs can be executed in the same test.

In this article, the same test cases that are used in KDF will be executed in a single test. The Keywords and the scripts for all the test cases are same as in KDF. However, TC3: Open the order has been parameterized. Hence the script for this test case is written to receive the order number from an Excel file and to write the customer name into the excel file

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_21.png)

**Test Case1: Login into the application**

**Keyword:**Login ()

**Test Case2: Insert the Order**

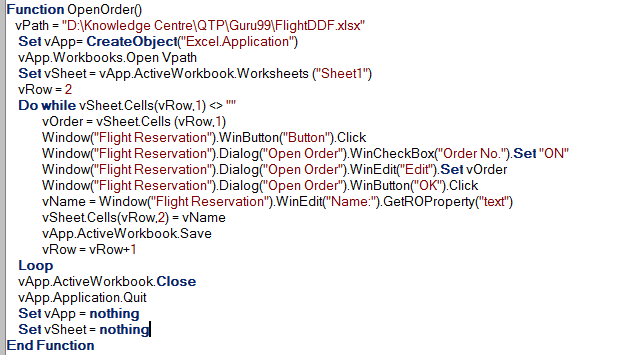
**Keyword:**InsertOrder()

**Test Case3: Open the Order for several order numbers**

**Keyword:**OpenOrder()

**Description:**Here the same script used to develop a DDF is used, thereby achieving the test case for several iterations.

**Script:**

[](https://cdn.guru99.com/images/QTP_Article_35/Article_35_22.png)

**Test Case4: Delete the Order**

**Keyword:**DeleteOrder()

**Test Case5: Close the application**

**Keyword:**CloseApp()

By following this simple method, parameterization of TC3 is achieved. If applicable, all the other test cases can also be parameterized in the same test.

Above example, is a very simple way of designing hybrid framework. The same framework can also be achieved with descriptive programming.

### Advantages

* The time taken to run the test designed with hybrid framework is relatively less compared to other frameworks
* This can be used when we need all the test cases and inputs that are associated with particular test case, in the same test suite.

### Disadvantage

* Clear knowledge on the combining different framework is required.