

Descriptive Programming and UFT



“Take the first step in faith. You don't have to see the whole staircase, just take the first step.”

- Martin Luther King, Jr. quotes

Marriage of UFT and Test Object

When UFT records the test it jot down all the property in object repository. We configure the object identification wizard to instruct UFT to pick the set of properties for specific class of object. But why we need this Object Repository? Because we want to store Object properties in it and UFT know how to pick and store the property during recording.

So, suppose if we can define our own property set and able to manipulate them at run time then we do not need object repository at all. It is like telling UFT to pick the particular object based on the property set which we can provide at run-time. Is it all sounding alien?

Let's first understand the object property and object recognition by UFT

STEP1: Spy the Google Search box and find object properties



[Open IE](#) | [Google.co.uk](#) | [Navigate to UFT](#) | [File](#) | [New](#) | [Test](#) | [Spy](#) | [Object spy](#) | [Spy Search box](#)

1 How UFT Understand

Object hierarchy:

- Browser : Google
 - Page : Google
 - WebTable : WebTable
 - WebEdit : q

2 How Object built by Dev Team

Google

`<input name="q" class="gbqfif" id="gbqfq" spellcheck="false" type="text" autocomplete="off" value=""/>`

Properties	Values
Class Name	WebEdit
abs_x	404
abs_y	373
class	gbqfif
default value	
disabled	0
height	18
html id	gbqfq
html tag	INPUT
innerHTML	
innerText	
kind	singleline
max length	2147483647
name	q
outerhtml	<input name="q" class="gbqfif" id="gbqfq" spellcheck="false" type="text" autocomplete="off" value=""/>
outerhtml	
pattern	
placeholder	
readonly	0
required	False
rows	0
type	text

Let's analyse the "q" in UFT perspective. UFT understand the properties and pick all the property:=value pair that are configured in object identification (UFT | Tools | Object Identification). The Class Name (known as micClass i.e. Mercury Interactive Class in UFT) tells the type of object for example web application can comprise WebEdit, Link, Button, Frame, Page, Browser object etc. The element has "Input" html tag with other properties for example "html id :=gbqfq", "kind: :=singleline", "max length := 2147483647"(character),"type := text".

Now if we can get the intelligence to pick these property by programming then there is no need of UFT object repository for object identification. We can build the description of the object by summing up these property in complex variable and can use/manipulate as per our requirement. So the same "q" object can be identify by programming by using the property set without using the object repository. Remember that we have just expand the object with its property:=value. It does not mean that we can lose the parent objects. To traverse up to the object level we must use proper hierarchy. If you notice the spy figure and the hierarchy below then there is no "WebTable" in hierarchy. Why?

UFT only pick the optimal hierarchy to identify the object uniquely. So if we get n-number of nested object in the hierarchy then UFT may skip few of them which are not necessary to reach object.

1

Object "q" in OR

Browser("Google").Page("Google").WebEdit("q").Set "Quicktestpro.co.uk"

2

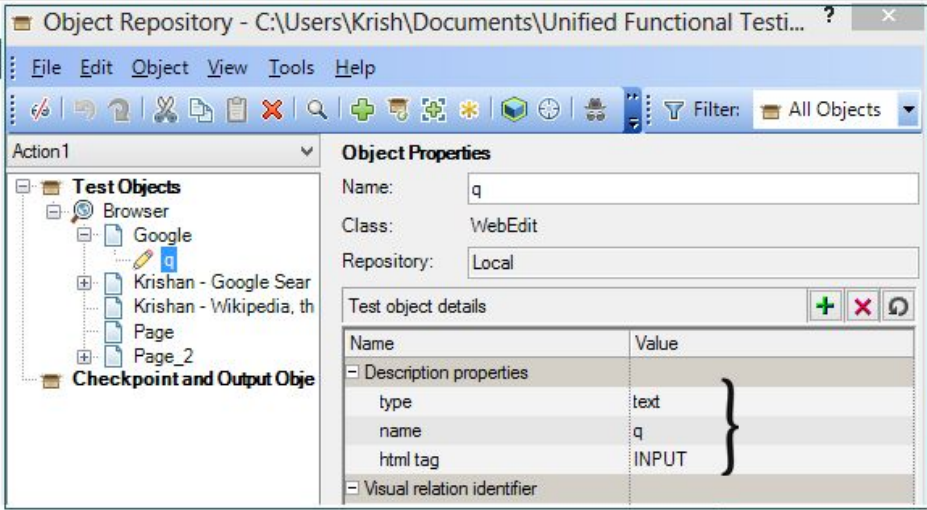
Unpack and describe properties of "q"

Browser("Google").Page("Google").WebEdit("html tag:=INPUT","html id:=gbqfq").Set "Quicktestpro.co.uk"

Remember “property:=value” set value use “:=” instead of “=”. So, do not confuse though it is little non-traditional way to write the pair.

We can describe the object and there will no need of OR to hold that object. But remember that OR still need to keep parent object otherwise

1



1. Object identify by Object Repository

2

.WebEdit("name:=q", "html tag:=INPUT", "type:=text")

2. Object identify by Descriptive way

Let’s now make this whole hierarchy free of object repository. So in same way we can describe all object in hierarchy.

Unplugged OR by describing each object in the hierarchy of test step	
	Browser("Google").Page("Google").WebEdit("q").Set "Quicktestpro.co.uk"
1	Browser("title:=Google")
2	.Page("title:=Google")
3	.WebEdit("name:=q", "html tag:=INPUT", "type:=text")
4	.Set "QuickTestPro.CO.UK"

Now there are three questions:

1. Should we spy each object, unpack all object? Then what is the use of UFT?
2. How many and which “property:= value” pair we should write to identify correct object?
3. Can we use descriptive programming for all the objects including browser and page so there will no need of object repository?

We will answer all these question but let's explore little bit more on programming side. Instead of providing the "property:=value" pair if we make an object comprises all the "property:=value" pair then we need not to unpack all properties. But that is exactly UFT was doing i.e. packing all property in one object "q" then? Yes, here we will have the control over the property set. How? Let's see the following example:

Exercise – Create a Script without Object Repository

STEP1: Google Search without recording and object repository using HTML



[Open UFT](#) | [File](#) | [New](#) | [DPProgramming1](#) | [Record](#) | [IE](#) | [Google.co.uk](#) | ["Nidhi"](#) | [Search](#) | [Close browser](#) | [Modify code as given below](#) | [Run](#)

	SystemUtil.Run "iexplore.exe", "http://www.google.co.uk"
	Browser("title:=Google").Page("title:=Google").WebEdit("name:=q", "type:=text").Set "Great Nidhi"
	Browser("title:=Google").Page("title:=Google").WebButton("name:=btnG", "type:=submit").Click
	Browser("title:=Google").CloseAllTabs

STEP2: Gmail automation by descriptive programming



[Open UFT](#) | [File](#) | [New](#) | [DPProgramming1](#) | [write following code](#) | [Run](#)

```
Systemutil.CloseProcessByName "iexplore.exe"
Systemutil.Run "iexplore.exe", "http://www.gmail.com"
UID = InputBox("Please provide user ID")
PWD = InputBox("Please provide Password")
```

```
Browser("title:=Gmail: Email from Google").Page("title:=Gmail: Email from Google").WebEdit("name:=Email").Set UID
```

```
Browser("title:=Gmail: Email from Google").Page("title:=Gmail: Email from Google").WebEdit("name:=Passwd").Set PWD
```

```
Browser("title:=Gmail: Email from Google").Page("title:=Gmail: Email from Google").WebButton("name:=Sign in").Click
Systemutil.CloseProcessByName "iexplore.exe"
```







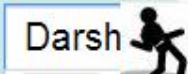
Concept of Exercise – Create a Script without Object Repository

In STEP1, we have created a script with describing single object i.e. "q" Google text box and search button. In This case we still need all parent hierarchy in OR. In STEP2, we are completely removing OR and describing all the objects including Browser and Page object.

But still, unpacking all properties and writing them in object makes no sense. It is same as reinventing wheel every time. Let's explore little bit more about it.


We can create a description of the object i.e. our own object with certain set of properties. VBScript provide Description.Create to create an object initialize with null value. This blank object can be assign with multiple set of “property:value” pair. Remember the syntax of providing the value to the property of object.

ObjectName(<PropertyName>).Value = <Value of property>

STEP 1	Set oWebEdit = Description.Create	NULL 	
STEP 2	oWebEdit ("name"). Value = "q"		
STEP 3	oWebEdit ("type"). Value = "text"		
STEP 4	Browser("X").Page("Y").WebEdit(oWebEdit).Set "Darsh"		

Now when we create a description of the object and pass into the hierarchy then UFT search the similar object in the application with matching properties and if it succeed, then it perform the action on the identified object (STEP 4). Remember the rule for hirerchy – If we use descriptive programming in any object then the hirerchy after that object must use descriptive objects. So if we are using description object for Page object then remaining hierarchy i.e. WebEdit must also use descriptive object.

Problem -1 what will happen when we create description with property that match more than one object for example if there are four WebEdit box and the following code will try to identify all the 4 object

	Test Script	Application	
	Set oWebEdit = Description.Create oWebEdit("type").value = "text"	<div><div>Aric</div><div>Siddhant</div><div>Darsh</div><div>?</div><div>Pari</div><div>Aadhya</div></div>	
	Description match with all WebEdits		

Exercise – Feel the power of being Zeus – Be a creator and describe the object

STEP1: Create a description of object properties.

 [Open UFT | New | Test | DynamicWriteDP | Write the following code](#)

```
Systemutil.CloseProcessByName "iexplore.exe"
Systemutil.Run "iexplore.exe","http://www.Google.com"
```

```
Set objWebEdit=Description.Create
objWebEdit ("micclass").value="WebEdit"
objWebEdit ("name").value="q"
objWebEdit ("html tag").value="INPUT"
objWebEdit ("html id").value="gbqfq"
```

STEP2: Record the Script of Google Search



Go to DynamicWriteDP | Record after last line | Google.com | "Lavi" | Click Search button | Close IE | Stop | Save

STEP3: Pass the Descriptive object in step



Go to DynamicWriteDP | Modify line-A from line-B | Run

A	Browser("Google").Page("Google_2").WebEdit("q"). Set "lavi"	Remove OR object
B	Browser("Google").Page("Google_2").WebEdit(objWebEdit). Set "lavi"	No quote for variable objWebEdit

STEP4: Create description for other object



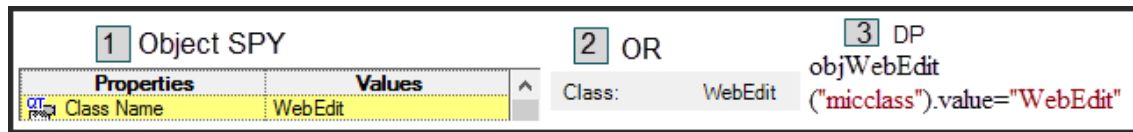
Go to DynamicWriteDP | Create Page Object by writing code from first line | Run

```
Set objPage=Description.Create
objPage ("micclass").value="Page"
objPage ("title").value="Google"
```

A	Browser("Google").Page("Google_2").WebEdit(objWebEdit). Set "Lavi"
B	Browser("Google").Page(objPage).WebEdit(objWebEdit). Set "Lavi"

Concept of Exercise – Feel the power of being Zeus – Be a creator and describe the object

In STEP1, we are creating an object with web edit description. Create method returns a property object collection in which we can add "properties:=value" in order to create the description of an object during run-time. Now if you notice a new term micclass in description. If you spy the object or verify the property in OR then also there will no micclass. micclass stands for Mercury Interactive Class. Web test objects do not support the Class Name identification property so micclass can be treated as a class property to create test object description.



So now the question is that which kind of object description we can create and what will be micclass for them.

STEP1: Explore all the identification properties and micclass of Web objects



Navigate to UFT | File | New | Test | ObjectIdentificationConfiguration | Tools | Object Identification | Web | Generate Script | Save in "C:\Test\myObjectIdentificationConfiguration.vbs" | Right click on "myObjectIdentificationConfiguration.vbs" | Edit | find "Web objects" by CTR + F | Explore all the object properties and micclass of the objects.

These are the frequently used object with their micclass.

	Web object	Micclass
1	Web Browser	Browser
2	Page	Page
3	Edit box	WebEdit
4	Image	Image
5	Link	Link
6	Web Element	WebElement
7	Button	WebButton
8	Checkbox	WebCheckBox
9	Combobox/DropDown	WebList
10	Table	WebTable

In STEP2, we have recorded the script. We need parent object in OR to reach upto our DP WebEdit object. In STEP3, we are getting rid of UFT recorded "q" object with our bundled objWebEdit object. Remember "q" was also a collection of property recorded by UFT and logically same as objWebEdit. It is just we are stealing the UFT power to build our own description to match the similar object during run-time. In STEP4, we are creating new objPage property object. Remember, just giving name of objPage doesn't give it right to contain the page property. It is just a variable name and can be anything, even oTomHanks. We need to fill this object with appropriate "property: =value" pair. Also you can only unbox the hierarchy i.e. if you any descriptive object in hierarchy then all other object in hierarchy in test-step must use descriptive object.

Create method in Description. Create statement returns property object.

	Peroperty Object/Method	Method and Property description
1	Add Method	Adds the property from another Properties object to our Properties object

2	Count Method	Returns the number of property items in the Properties collection.
3	Item Method	Accesses a specific property item in a Properties collection, to set or retrieve its value.
4	Remove Method	Removes the specified property item from the Properties collection.
5	Name Property	Sets or retrieves the name of an existing property or adds a new property item if item does not exist
6	RegularExpression	Indicates whether the value of a property item in the Properties collection is a regular expression.
7	Value Property	Sets or retrieves or add the value of an property item in the Properties collection

We can correlate the above method in the example below:

```

Set oDesc = Description.Create
7 oDesc("prop_1").Value = "val.1"
  oDesc("prop_2").Value = "val.2"
3 oDesc.Item("xyz").Value = "userName"
6 oDesc("PropName").RegularExpression = False ' Default True
1 oDesc.Add OtherCollection(0)
2 For i = 0 To oDesc.Count - 1
5   objDescription = "Name: " & oDesc(i).Name & "Value: " & oDesc(i).Value
Next
  oDesc.Remove "prop_1"

```

We have analyze two ways of Descriptive programming. One requires string in test steps while other require physical object by Description.Create. Following are the few key difference between them:

Mars Vs Venus	Step based DP (Static)	Object based DP (Dynamic)
Statement	.WebEdit("name:=q", "html tag:=INPUT", "type:=text")	Set objWebEdit=Description.Create
Way of working	String used to describe Web Object	Physical Object creation
Test Script	Readability issue due to long statement	Increase line of code due to property assignment
System resource	Require no additional memory	Due to new object, It consume system memory
When?	object requires in few steps	When object requires in many statements

Let's talk about business – Descriptive Programming Advantages

We have seen the concept of Descriptive Programming (DP). DP brings many benefits. The important question is how effectively we are using it. Let's explore common example of DP that will give the

idea about the control over the test application by using DP. But let's first explore the important aspect of UFT i.e. "ChildObject".

Child Object returns all the child object of parent. The Statement Page("X").ChildObject will return all childobject of the page (except WebElement).

STEP1: Get all Child Object of QuickTestPro.CO.UK



Navigate to UFT | File | New | Test | ChildObject | Record | IE | QuickTestPro.Co.UK | Click on any link | Close IE | Save | Modify script accordingly | Run

```
SystemUtil.Run "iexplore.exe"  
Browser("Browser").Navigate "http://quicktestpro.co.uk/"  
Set ohelloKids = Browser("Browser").Page("Quick Test Pro").ChildObjects
```

```
'All Collection have count property so don't afraid to use it  
MsgBox ohelloKids.Count
```

```
For i = 0 to ohelloKids.Count - 1  
ohelloKids(i).Highlight  
Next
```

```
Browser("Browser").Page("Quick Test Pro").Sync  
Browser("Browser").CloseAllTabs
```

ChildObject returns all the child of the WebPage. But WebPage comprises various type of webObjects for example WebEdit, Link, Image etc. So what if we want to access a specific kind of kid from our Allkids? We can pass our kid description in ChildObject method and it will return all child object matching with the specified criteria.

STEP1: Get all Link on QuickTestPro.CO.UK by ChildObject



Navigate to UFT | File | New | Test | LinkChildObject | Record | IE | QuickTestPro.Co.UK | Click on any link | Close IE | Save | Modify script accordingly | Run

1	Set oLinks = Description .Create
2	oLinks("micclass").Value = "Link"
3	SystemUtil.Run "iexplore.exe"
4	Browser("Browser").Navigate "http://quicktestpro.co.uk/"
5	Set ohelloKids = Browser("Browser").Page("Quick Test Pro").ChildObjects(oLinks)
7	<i>'All Collection have count property so don't afraid to use it</i>
9	MsgBox ohelloKids.Count
10	For i = 0 to ohelloKids.Count - 1
11	ohelloKids(i).Highlight
12	print ohelloKids(i).GetRoProperty("text")
13	Next

In STEP1, we have created a description of our object (kid) and passed in ChildObject. It returns all the kid matched with the specified description. Line-1 & Line-2 provide description of all link object. We can refine and narrow our description by specifying the more properties. We passed the description in childobject in Line-5 to return all the link objects on webpage during run-time. We skipped the object repository and now directly speaking with webpage. All collection provides the count property. So, we will get the number of links present on page at Line-9. Now remember that ohelloKids is a collection of individual links i.e. each link has its own property values. Line-11 gives us the opportunity to perform any action on any link while Line-12 returns the run-time (i.e. link on webpage not in OR) link property. We can employ if-else conditional logic at line-12 to perform any action on desired link. For Example

```
If ohelloKids(i).GetRoProperty("text") = "TechnoPreacher" Then
    ohelloKids(i).Click
End if
```

Now we can build a reusable function that can return all the objects based on their micclass. We can devise our own logic to perform action on specific object.

STEP1: Let's Describe Everything – Manage test programmatically



[Navigate to UFT](#) | [File](#) | [New](#) | [Test](#) | [LinkChildObject](#) | [Record](#) | [IE](#) | [QuickTestPro.Co.UK](#) |
[Click on any link](#) | [Close IE](#) | [Save](#) | [Modify script accordingly](#) | [Run](#)

1	Set oBrowser = Description .Create
2	oBrowser("micclass").value = "Browser"
3	oBrowser("title").value = "Google"
4	Set oPage1 = Description .Create
5	oPage1("micclass").value = "Page"
6	oPage1("title").value = "Google"
7	SystemUtil.Run "iexplore.exe", "Google.com"
8	wait 3
9	Set oPage = Browser(oBrowser).Page(oPage1)
10	MsgBox "Number of Edits: " & objAllEdit(oPage).Count
11	MsgBox "Number of Buttons: " & objAllButton(oPage).Count
12	MsgBox "Number of Links: " & objAllLinks(oPage).Count
13	MsgBox "Number of Images: " & objAllImage(oPage).Count
14	Function objAllEdit(Page)
15	Set objAllEdit = MasterFunction(Page, "WebEdit")
16	End Function
17	Function objAllButton(Page)
18	Set objAllButton = MasterFunction(Page, "WebButton")
19	End Function
20	Function objAllLinks(Page)
21	Set objAllLinks = MasterFunction(Page, "Link")

22	End Function
23	Function objAllImage(Page)
24	Set objAllImage = MasterFunction(Page, "Image")
25	End Function
26	SystemUtil.CloseProcessByName "iexplore.exe"
27	Function MasterFunction(Page, MicClass)
28	Set Desc = Description .Create()
29	Desc("micclass").Value = MicClass
30	Set MasterFunction = Page.ChildObjects(Desc)
31	End Function

In this code, we wrote reusable function that returns all object based on their class (micclass in UFT).

Let's dig this code. We are regrencyng oPage object with Google page.

Jumping tide: LINE- 10 → Line- 14 → Line 27 → Line 15 → Line 10

We are passing the page object in line 10. N line 14, we again called the master function with page and "WebEdit" micclass. In master function, we are creating description with micclass and returning all ChildObject of page of specific class.

Remember: In VBS, we return value from function by assigning value in function name

A = x

Function x

X = 10 'return 10 to A, calling statement

End Function

Debug yourself: Insert breakpoint and debug the function.

Jumping nottuB: Line 11 → Line- 17 → Line 27 → Line 18 → Line 11

Jumping kniL : Line 12 → Line- 20 → Line 27 → Line 21 → Line 12

Jumping egamI : Line 13 → Line- 23 → Line 27 → Line 24 → Line 13

Ordinal Identifier and Descriptive programming

We deal with collection of object while using descriptive programming. Most of the time we require to identify the object based on its index or It becomes imperative to perform action based on the index or creation time property for example second editbox or third browser etc.

Example1:

set oEditDesc = Description.Create()

oEditDesc ("Name").Value = "q"

oEditDesc ("Index").Value = "0" ' First Object starts with zero index

Browser ("title:=Google").Page("title:=Google ").WebEdit(oEditDesc).Set "Richa"

Or

Browser ("title:=Google").Page("title:=Google ").WebEdit("name:=q", "index:=0").Set "Richa"

Example2:

```

SystemUtil.Run "iexplore.exe" 'Browser with CreationTime 0
SystemUtil.Run "iexplore.exe" 'Browser with CreationTime 1
Browser("CreationTime:=0").Navigate "http://QuickTestPro.CO.UK"
Browser("CreationTime:=1").Navigate "http://KnowledgeInbox.com"

```

Confusion Eliminator: Considerations for Descriptive programming

Let's make code little more complex: Variable in DP statement

We can enter a variable name as the property value if you want to find an object based on property values we retrieve during a run session. For example:

```

nameVar="q"
Browser("Google").Page("Google").Webedit("name:=" & nameVar)

```

Regular Expression

UFT evaluates all property values in programmatic descriptions as regular expressions. Therefore, if you want to enter a value that contains a special regular expression character (such as *, ?, or +), use the \ (backslash) character to instruct UFT to treat the special characters as literal characters

Reuse DP String

String based DP annoy with it's the lengthy code across the script. We can rectify it by assigning the object with description in variable.

```

Set MyWObject = Window("Text:=Myfile.txt - Notepad")
MyWObject.WinEdit("AttachedText:=Find what:").Set "Yogi"
MyWObject.WinButton("Caption:=Find next").Click

```

Object Spy and Copying properties

Object spy provides "Copy Identification Properties to Clipboard" so we need not to spy each and every time to jot down the object property and build our description in script.

Let's Explore Where Descriptive Truth Prevails

Similar Action on Same Type of Object:

Suppose you have online government form web application with 50 text field and you want to fill the data in textbox. Now adding all 20 the Object repository would not be a good programming approach. We can create description and manage all textbox in few lines of code without OR interaction.

Big OR

When we run script, OR loads in RAM. RAM optimal utilization is precious for better performance. If the size of Object repository increases too much then it decreases the performance of QTP while recognizing an object. Descriptive Programming can solve this problem.

Dynamic Object

This might be the most important aspect to implement descriptive programming. Suppose you place an order on amazon or ebay and it generate different order number each time. Or Welcome <user_name> dynamic object after login. The dynamic object like these can be handle by descriptive programming.

Shared and Conflicting Object Repository Updates

Suppose we have one script that modify OR and other script that need to read old values of the OR. This is conflict situation and end-up in "dirty-read" or "dirty-write". To avoid this we can implement DP in this situation.

Miso-Object Repositorist

Do you hate object repository. Do you have OR Phobia? We have solution for you but tell us why should you do not like OR?

1. Web application in development stage and object cannot capture for OR.
2. Suppose same object appear on whole application. Suppose we have 30 page application and same 5 links and 3 button appear on each page. Adding all will not be the good way to resolve the issue.

All in those case where we want to minimize OR interaction, Descriptive Programing come for rescue. It provide the greater controller on the application during run-time and save OR overhead.