GUI Scripts with KiXforms, Part 3

Exposing some unique controls not normally available via Script

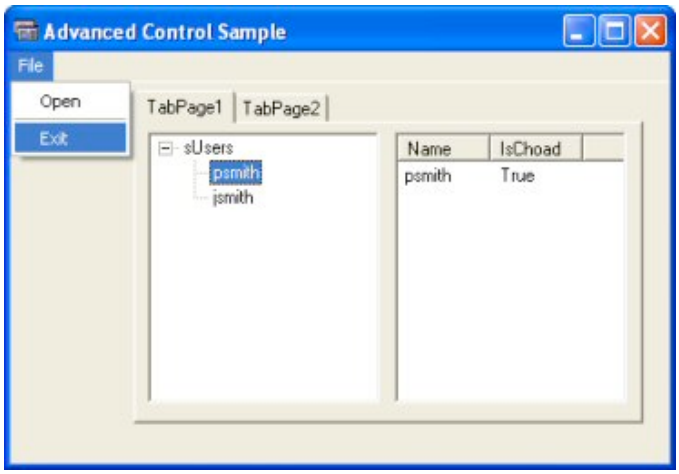
by Bob Kelly
January 2007

We covered what KiXforms is all about (part 1) and how to use it in your scripts (part 2), now we finally get into some things you normally cannot do with a script. Specifically, I'm talking about MenuBars, TabControls and Listviews. There are several more controls offered as part of KiXforms, but I'm working to keep this short and simple so we will focus on these.

Again, we need to create an object variable that references the KiXforms DLL. The Type Library name is "KiXtart.System" so we begin by setting this object variable:

```
Set System = CreateObject("KiXtart.System")
```

To shorten the code you will see me using an alternate way of specifying the height and width properties (size) and the top and left properties (location). Either method will work, but this method results in less lines of code. I think that with the very basics behind us (see part 2) the logic behind what is being done here should still be clear. Well again start with the form object:




```
Set Form1 = System.Form()
Form1.Size =
System.Size(435, 300)
Form1.Text = "Advanced
Control Sample"
```

Now that we have our form, let's add the menu bar. Like the controls that follow, we will first create the object, set its properties and then, add an item to that control and set its properties. In this case, we add the MainMenu object set the top level menu item to read "file" and then add other items (open, a separator and exit) to appear beneath this item. Finally, we tell the form (Form1) that this is the menu it is to use. It may sound a little confusing, but follow along and you'll see it is not so bad:

```
Set MainMenu1 = System.MainMenu()
Set FileMenu = MainMenu1.MenuItems.Add("File")
FileMenu.Text = "File"
Set OpenMenuItem = FileMenu.MenuItems.Add("Open")
OpenMenuItem.Text = "Open"
Set MenuItem1 = FileMenu.MenuItems.Add("-")
```

```
MenuItem1.Text = "-"
Set ExitMenuItem = FileMenu.MenuItems.Add("Exit")
ExitMenuItem.Text = "Exit"
Form1.Menu = MainMenu1
```

To wire those entries to events, you could say ExitMenuItem.OnClick = "Wscript.Exit" using the OnClick event to instruct which function should be executed when the item is clicked by the user. We will put our ListView and TreeView in a tab control, so let's create that next…

```
Set TabControl1 = Form1.Controls.TabControl
TabControl1.Size = System.Size(330, 215)
TabControl1.Location = System.Point(80, 5)
Set TabPage1 = TabControl1.Controls.TabPage
TabPage1.Size = System.Size(322, 189)
TabPage1.Text = "TabPage1"
Set TabPage2 = TabControl1.Controls.TabPage
TabPage2.Size = System.Size(192, 74)
TabPage2.Text = "TabPage2"
```

As you can see, we created two tabs and set their size, location and the text to be displayed on each tab. Next, let's take care of the TreeView code by creating the object and adding a root node (Node0). We set its text and we then add the sub-nodes to it by adding them to the Node0 object instead of to the TreeView object directly (which would result in more root nodes):

```
Set TreeView1 = TabPage1.Controls.TreeView
TreeView1.Size = System.Size(150, 175)
TreeView1.Location = System.Point(5, 5)
Set Node0 = TreeView1.Nodes.Add()
Node0.Text = "sUsers"
Set Node2 = Node0.Nodes.Add()
Set Node1 = Node0.Nodes.Add()
Node2.Text = "psmith"
Node1.Text = "jsmith"
```

Finally, we have the ListView: you'll find the logic behind how to set up each item is very similar and the included documentation is a critical resource. Keep in mind that while the code samples are written in KiXtart. it is still a very valuable resource to learn what objects, properties and methods are available to your VBScript. For the ListView, we first add the columns and set their names before adding a sample item (the text for the first column) and a sub-item (the text for the next column).

```
Set ListView1 = TabPage1.Controls.ListView
ListView1.Size = System.Size(150, 175)
ListView1.Location = System.Point(165, 5)
Set sName = ListView1.Columns.Add()
sName.Text = "Name"
Set IsChoad = ListView1.Columns.Add()
IsChoad.Text = "IsChoad"
Set Item1 = ListView1.Items.Add()
Item1.Text = "psmith"
Set SubItem1= Item1.SubItems(1)
SubItem1.Text = "True"
```

And as in our previous example, we now need to get the script in a loop where it will display while looking for any events that may be triggered. In this example we are really focusing on the controls and not their events, so in actuality this form will not do anything but look impressive.

```
Form1.Visible = "True"
Do While Form1.Visible
      Execute(Form1.DoEvents)
Loop
```

I hope these samples and descriptions have helped you to see some new potential for your scripts. While one might well argue that if you want to do all of this, you should learn a programming language, many like to stick with what they know. If you don't have the time to learn a full blown programming language this gets you there with little effort!

Bob Kelly is president and co-founder of AdminScriptEditor.com, home to an integrated suite of scripting tools and a shared library of scripts and language help. He has authored books on scripting and desktop administration as well as several white papers. Bob also owns and operates AppDeploy.com, where he writes and produces videos on topics related to software deployment. You can contact Bob about "GUI Scripts with KiXforms" at bkelly@adminscripteditor.com.