Recent

Topics

Tutorials

Highlights

Settings

Feedback (http://community.

Sign Out

PREV
Loading and cr

NEXT
ary f…

Settings

10 days left in your trial.
Subscribe.

Feedback
(http://community.safaribooksonline.cor

Sign Out

Drawing a rectangle on the screen with Win32 API methods (Extern)

In this recipe, you will also see an example of how to use the `Extern` reserved object to define references to methods in external DLLs suc as those of the Win32 API. These methods can then be loaded and executed during runtime. We have already seen an example of this ir the *Reading values from an INI file* recipe in Chapter 1, *Data-driven Tests*. Here, we will learn how to implement a function that draws a rectangle on the screen with the color of your choice. This is useful tc mark areas on the screen that are of interest (especially when the tes fails) and hence, makes the report analysis task more efficient.

Getting ready

From the **File** menu, navigate to **New** | **Test**, or use the *Ctrl* + *N* shortcut.

To complete this recipe, we need the global `Extern` object, which, witl proper use, provides UFT with access to the methods of an external **Dynamic Link Library** (**DLL**). We will define a variable and assign it a reference to the global `Extern` object (this is done to avoid persistence, as `Extern` is a reserved object that is not released from memory until UFT closes):

```
Dim oExternObj

Set oExternObj = Extern
```

Then, we will declare the methods required to accomplish our task; in this case, to draw a rectangle on the screen:

```
With oExternObj
    .Declare micHwnd, "GetDesktopWindow", "User32.DLL", "GetDesktopWindow
    .Declare micULong, "GetWindowDC", "User32.DLL", "GetWindowDC",micHwnd
    .Declare micInteger, "ReleaseDC", "User32.DLL", "ReleaseDC", micHwnd,
    .Declare micULong, "CreatePen", "Gdi32.DLL", "CreatePen", micInteger,
    .Declare micInteger, "SetROP2", "Gdi32.DLL", "SetROP2", micULong, mic
    .Declare micULong, "SelectObject", "Gdi32.DLL", "SelectObject", micUL
    .Declare micULong, "DeleteObject", "Gdi32.DLL", "DeleteObject", micUL
    .Declare micULong, "GetStockObject", "Gdi32.DLL", "GetStockObject", m
    .Declare micULong, "Rectangle", "Gdi32.DLL", "Rectangle", micULong, m
End With
```

How to do it…

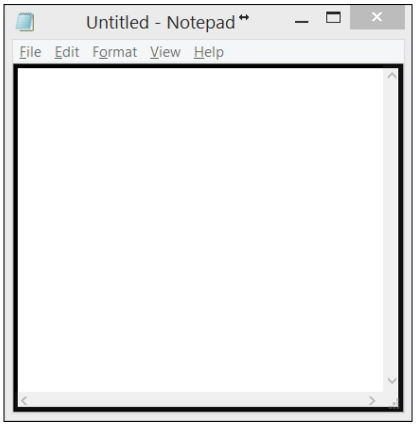
After we define the connection to the DLL with its returned value and arguments, we will write a function that accepts the following arguments, namely, `TestObject` and a reference to the `Extern` object named `oExternLocal`:

```
Function DrawRect(ByRef TestObject, ByVal oExternLocal)
    Dim YTop, XLeft, YBottom, XRight
    Dim hDC, hPen

    'Get object coordinates
    With TestObject
        XLeft = .GetROProperty("abs_x")
        YTop = .GetROProperty("abs_y")
        YBottom = YTop+.GetROProperty("height")-1
        XRight = XLeft+.GetROProperty("width")-1
    End With

    With oExternLocal
        ' Get the Desktop DC
        hDC = .GetWindowDC(.GetDesktopWindow)
        ' Create a five pixels wide Pen
        hPen = .CreatePen(6, 5, RGB(0, 0, 0)) ' PS_INSIDEFRAME, 3 , RGB(0
        .SetROP2 hDC, 6 ' hDC, R2_NOT
        .SelectObject hDC, hPen
        ' Use an empty fill
        .SelectObject hDC, .GetStockObject (5) ' NULL_BRUSH

        ' Draw the rectangle
         .Rectangle hDC, XLeft, YTop, XRight, YBottom

        ' CleanUp
        .ReleaseDC .GetDesktopWindow, hDC
        .DeleteObject hPen
    End With

    Set oExtern = Nothing
End Function
```

To utilize the function, use the following code:

```
DrawRect Window("Notepad").WinEditor("Edit"), oExternObj
```

As a result of running the test, the **Notepad** window would look similar to the following screenshot for a brief time (it is possible to extend it by adding a `Wait` command before releasing the drawing context and pen objects):



How it works…

First, we use the variable `oExternObj` as a reference (shallow copy) to the `Extern` reserved object to avoid persistence, that is, to ensure that the declared external methods do not remain in memory. Otherwise, we will need to close and reopen UFT to reset the `Extern` object.

Second, we call the function `DrawRect` and pass `TestObject` to be highlighted (in this case, the Notepad `WinEditor`) and the `oExternObj` variable.

Third, the function `DrawRect` calculates the boundaries of the given `TestObject`, and calls the relevant methods from the external Win32 API via the `oExternLocal` object to set the required resources (`Pen`, `Drawing Context`, and so on). It then uses the `Rectangle` method to actually draw a five-pixel-wide rectangle around `TestObject`. Finally, it releases the resources.