Chapter 43: Automation or Using other applications Libraries

If you use the objects in other applications as part of your Visual Basic application, you may want to establish a reference to the object libraries of those applications. This Documentation provides a list, sources and examples of how to use libraries of different softwares, like Windows Shell, Internet Explorer, XML HttpRequest, and others.

Section 43.1: VBScript Regular Expressions

```
Set createVBScriptRegExObject = CreateObject("vbscript.RegExp")
```

Tools> References> Microsoft VBScript Regular Expressions #.#
Associated DLL: VBScript.dll
Source: Internet Explorer 1.0 and 5.5

- MSDN-Microsoft Beefs Up VBScript with Regular Expressions
- MSDN-Regular Expression Syntax (Scripting)
- experts-exchange - Using Regular Expressions in Visual Basic for Applications and Visual Basic 6
- How to use Regular Expressions (Regex) in Microsoft Excel both in-cell and loops on SO.
- regular-expressions.info/vbscript
- regular-expressions.info/vbscriptexample
- WIKI-Regular expression

Code

You can use this functions to get RegEx results, concatenate all matches (if more than 1) into 1 string, and display result in excel cell.

```
Public Function getRegExResult(ByVal SourceString As String, Optional ByVal RegExPattern As String
= "\d+", _
    Optional ByVal isGlobalSearch As Boolean = True, Optional ByVal isCaseSensitive As Boolean =
False, Optional ByVal Delimiter As String = ";") As String

    Static RegExObject As Object
    If RegExObject Is Nothing Then
        Set RegExObject = createVBScriptRegExObject
    End If

    getRegExResult = removeLeadingDelimiter(concatObjectItems(getRegExMatches(RegExObject,
SourceString, RegExPattern, isGlobalSearch, isCaseSensitive), Delimiter), Delimiter)

End Function

Private Function getRegExMatches(ByRef RegExObj As Object, _
    ByVal SourceString As String, ByVal RegExPattern As String, ByVal isGlobalSearch As Boolean,
ByVal isCaseSensitive As Boolean) As Object

    With RegExObj
        .Global = isGlobalSearch
        .IgnoreCase = Not (isCaseSensitive) 'it is more user friendly to use positive meaning of
argument, like isCaseSensitive, than to use negative IgnoreCase
        .Pattern = RegExPattern
        Set getRegExMatches = .Execute(SourceString)
    End With

End Function
```

```
Private Function concatObjectItems(ByRef Obj As Object, Optional ByVal DelimiterCustom As String =
";") As String
    Dim ObjElement As Variant
    For Each ObjElement In Obj
        concatObjectItems = concatObjectItems & DelimiterCustom & ObjElement.Value
    Next
End Function

Public Function removeLeadingDelimiter(ByVal SourceString As String, ByVal Delimiter As String) As
String
    If Left$(SourceString, Len(Delimiter)) = Delimiter Then
        removeLeadingDelimiter = Mid$(SourceString, Len(Delimiter) + 1)
    End If
End Function

Private Function createVBScriptRegExObject() As Object
    Set createVBScriptRegExObject = CreateObject("vbscript.RegExp") 'ex.:
createVBScriptRegExObject.Pattern
End Function
```

Section 43.2: Scripting File System Object

```
Set createScriptingFileSystemObject = CreateObject("Scripting.FileSystemObject")
```

Tools> References> Microsoft Scripting Runtime
Associated DLL: ScrRun.dll
Source: Windows OS

MSDN-Accessing Files with FileSystemObject

> The File System Object (FSO) model provides an object-based tool for working with folders and files. It allows you to use the familiar object.method syntax with a rich set of properties, methods, and events to process folders and files. You can also employ the traditional Visual Basic statements and commands.

> The FSO model gives your application the ability to create, alter, move, and delete folders, or to determine if and where particular folders exist. It also enables you to get information about folders, such as their names and the date they were created or last modified.

MSDN-FileSystemObject topics: "*...explain the concept of the FileSystemObject and how to use it.*" exceltrick-FileSystemObject in VBA – Explained
Scripting.FileSystemObject

Section 43.3: Scripting Dictionary object

```
Set dict = CreateObject("Scripting.Dictionary")
```

Tools> References> Microsoft Scripting Runtime
Associated DLL: ScrRun.dll
Source: Windows OS

Scripting.Dictionary object
MSDN-Dictionary Object

Section 43.4: Internet Explorer Object

```
Set createInternetExplorerObject = CreateObject("InternetExplorer.Application")
```

Tools> References> Microsoft Internet Controls
Associated DLL: ieframe.dll
Source: Internet Explorer Browser

MSDN-InternetExplorer object

> Controls an instance of Windows Internet Explorer through automation.

Internet Explorer Objec Basic Members

The code below should introduce how the IE object works and how to manipulate it through VBA. I recommend stepping through it, otherwise it might error out during multiple navigations.

```
Sub IEGetToKnow()
    Dim IE As InternetExplorer 'Reference to Microsoft Internet Controls
    Set IE = New InternetExplorer

    With IE
        .Visible = True 'Sets or gets a value that indicates whether the object is visible or
hidden.

        'Navigation
        .Navigate2 "http://www.example.com" 'Navigates the browser to a location that might not be
expressed as a URL, such as a PIDL for an entity in the Windows Shell namespace.
        Debug.Print .Busy 'Gets a value that indicates whether the object is engaged in a navigation
or downloading operation.
        Debug.Print .ReadyState 'Gets the ready state of the object.
        .Navigate2 "http://www.example.com/2"
        .GoBack 'Navigates backward one item in the history list
        .GoForward 'Navigates forward one item in the history list.
        .GoHome 'Navigates to the current home or start page.
        .Stop 'Cancels a pending navigation or download, and stops dynamic page elements, such as
background sounds and animations.
        .Refresh 'Reloads the file that is currently displayed in the object.

        Debug.Print .Silent 'Sets or gets a value that indicates whether the object can display
dialog boxes.
        Debug.Print .Type 'Gets the user type name of the contained document object.

        Debug.Print .Top 'Sets or gets the coordinate of the top edge of the object.
        Debug.Print .Left 'Sets or gets the coordinate of the left edge of the object.
        Debug.Print .Height 'Sets or gets the height of the object.
        Debug.Print .Width 'Sets or gets the width of the object.
    End With

    IE.Quit 'close the application window
End Sub
```

Web Scraping

The most common thing to do with IE is to scrape some information of a website, or to fill a website form and submit information. We will look at how to do it.

Let us consider *example.com* source code:

```
<!doctype html>
<html>
    <head>
        <title>Example Domain</title>
        <meta charset="utf-8" />
        <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1" />
        <style ... </style>
    </head>

    <body>
        <div>
            <h1>Example Domain</h1>
            <p>This domain is established to be used for illustrative examples in documents. You
may use this
            domain in examples without prior coordination or asking for permission.</p>
            <p><a href="http://www.iana.org/domains/example">More information...</a></p>
        </div>
    </body>
</html>
```

We can use code like below to get and set information:

```
Sub IEWebScrape1()
    Dim IE As InternetExplorer 'Reference to Microsoft Internet Controls
    Set IE = New InternetExplorer

    With IE
        .Visible = True
        .Navigate2 "http://www.example.com"

        'we add a loop to be sure the website is loaded and ready.
        'Does not work consistently. Cannot be relied upon.
        Do While .Busy = True Or .ReadyState <> READYSTATE_COMPLETE 'Equivalent = .ReadyState <> 4
            ' DoEvents - worth considering. Know implications before you use it.
            Application.Wait (Now + TimeValue("00:00:01")) 'Wait 1 second, then check again.
        Loop

        'Print info in immediate window
        With .Document 'the source code HTML "below" the displayed page.
            Stop 'VBE Stop. Continue line by line to see what happens.
            Debug.Print .GetElementsByTagName("title")(0).innerHtml 'prints "Example Domain"
            Debug.Print .GetElementsByTagName("h1")(0).innerHtml 'prints "Example Domain"
            Debug.Print .GetElementsByTagName("p")(0).innerHtml 'prints "This domain is
established..."
            Debug.Print .GetElementsByTagName("p")(1).innerHtml 'prints "<a
href="http://www.iana.org/domains/example">More information...</a>"
            Debug.Print .GetElementsByTagName("p")(1).innerText 'prints "More information..."
            Debug.Print .GetElementsByTagName("a")(0).innerText 'prints "More information..."

            'We can change the localy displayed website. Don't worry about breaking the site.
            .GetElementsByTagName("title")(0).innerHtml = "Psst, scraping..."
            .GetElementsByTagName("h1")(0).innerHtml = "Let me try something fishy." 'You have just
changed the local HTML of the site.
            .GetElementsByTagName("p")(0).innerHtml = "Lorem ipsum.......... The End"
            .GetElementsByTagName("a")(0).innerText = "iana.org"
        End With '.document

        .Quit 'close the application window
```

```
    End With 'ie

End Sub
```

What is going on? The key player here is the **.Document**, that is the HTML source code. We can apply some queries to get the Collections or Object we want.
For example the `IE.Document.GetElementsByTagName("title")(0).innerHtml`. `GetElementsByTagName` returns a **Collection** of HTML Elements, that have the "*title*" tag. There is only one such tag in the source code. The **Collection** is 0-based. So to get the first element we add `(0)`. Now, in our case, we want only the `innerHtml` (a String), not the Element Object itself. So we specify the property we want.

Click

To follow a link on a site, we can use multiple methods:

```
Sub IEGoToPlaces()
    Dim IE As InternetExplorer 'Reference to Microsoft Internet Controls
    Set IE = New InternetExplorer

    With IE
        .Visible = True
        .Navigate2 "http://www.example.com"
        Stop 'VBE Stop. Continue line by line to see what happens.

        'Click
        .Document.GetElementsByTagName("a")(0).Click
        Stop 'VBE Stop.

        'Return Back
        .GoBack
        Stop 'VBE Stop.

        'Navigate using the href attribute in the <a> tag, or "link"
        .Navigate2 .Document.GetElementsByTagName("a")(0).href
        Stop 'VBE Stop.

        .Quit 'close the application window
    End With
End Sub
```

Microsoft HTML Object Library or IE Best friend

To get the most out of the HTML that gets loaded into the IE, you can (or should) use another Library, i.e. *Microsoft HTML Object Library*. More about this in another example.

IE Main issues

The main issue with IE is verifying that the page is done loading and is ready to be interacted with. The `Do While...Loop` helps, but is not reliable.

Also, using IE just to scrape HTML content is OVERKILL. Why? Because the Browser is meant for browsing, i.e. displaying the web page with all the CSS, JavaScripts, Pictures, Popups, etc. If you only need the raw data, consider different approach. E.g. using XML HTTPRequest. More about this in another example.