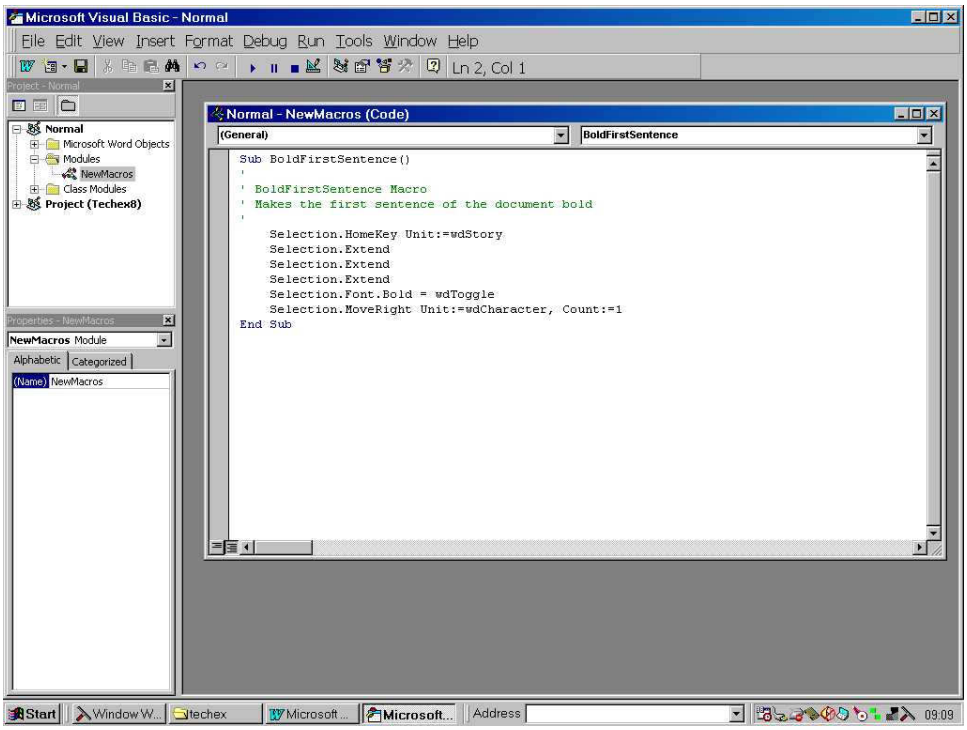
TECH EXCHANGE No. 8 – Microsoft Visual Basic for Applications (VBA) (Part 1)

This is the first of a two-part article showing how we can use Visual Basic for Applications (VBA) to program simple tasks using the main Microsoft Office components. Don't worry you do not need to be a programmer to use these techniques to enhance your use of Office. In this part we will cover the history of VBA, the software tools and explain the common terminology used. First the history, VBA first saw the light of day as a replacement for the macro language in Microsoft Excel in 1993 and was included in Microsoft Project in 1994. VBA replaced Access Basic in Microsoft Access in 1995 and in 1996 replaced Word Basic in Microsoft Word. With the release of Office 97, all the big four - Word, Excel, Access and PowerPoint - all have VBA 5 at their core. In 1998 VBA version 6 was launched. Other companies have licensed VBA from Microsoft to include in their own products, hence providing a consistent programming interface across a wide range of products. Ok so how does VBA differ from VBScript? VBScript is a subset of VBA, and is simply an interpreted language component (scrrun.dll) with no Integrated Development Environment (IDE) unless you count the ActiveX control pad of course. VBScript is used primarily in the following four scripted environments: Active Server Pages (ASP), Microsoft's server-side scripting technology for the Internet Information Server (IIS), Outlook forms, Windows Scripting Host (WSH), the scripting technology for automating the Windows desktop, and Microsoft Internet Explorer. All these will be subjects for other articles.

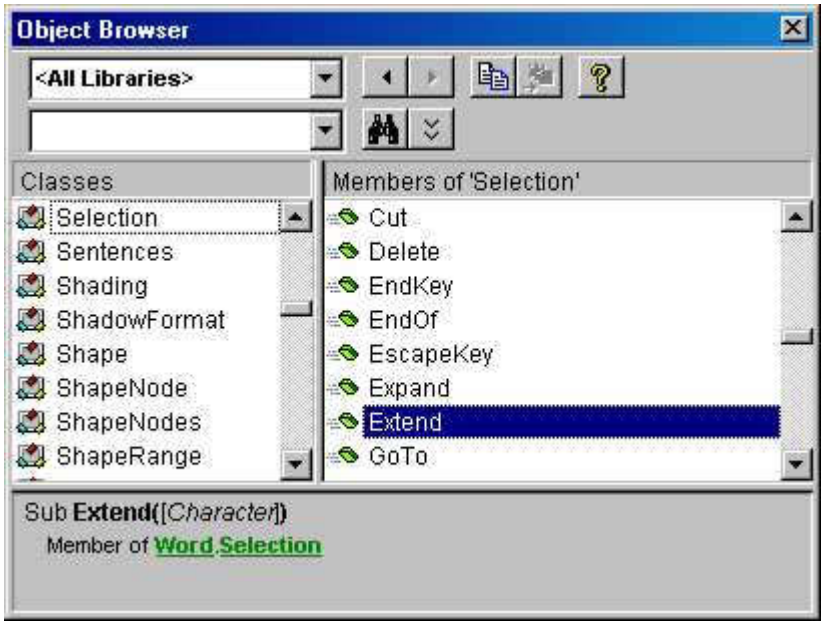


Let's put all this into context with an example. Start Microsoft Word and type in a few sentences of text you want. We are going to write a macro to make the first sentence of your document **bold**. Select Tools | Macro | Record New Macro menu option to display the Record Macro dialog box. Type 'BoldFirstSentence' as the macro name. Enter 'Makes the first sentence of the document bold' as the macro description. Click the OK button to return to the document with the Stop Recording Macro toolbar displayed. Type CTRL-HOME to position the cursor at the beginning of the document. Press F8 (extend text selection) three times to highlight the first sentence. Type CTRL-B to make the marked text bold and the right arrow to remove the selection highlight. Select Tools | Macro | Stop Recording menu option or click the Stop Macro button. To see the code for your macro select Tools | Macro | Macros menu option (or press ALT-F8) to display the Macro dialog box. In the macro name list highlight the macro we have just recorded, and then click the Edit button. This Displays the Visual Basic Editor. Recording a macro often gives us a good starting point for a VBA script, allowing us then to hand-craft it to achieve the desired result.

The Visual Basic Editor (VBE) is our view of the VBA universe. Within any of Office's components we can use the keystroke **ALT-F11** to display the VBE (see included screen shot). The VBE is divided into three areas: The Project Explorer, the Properties window and the work area. The Project Explorer shows a hierarchical view of the contents of the current VBA project. The Properties window shows the various properties for the object selected in the Project Explorer. The Properties window is divided into two columns, the left showing all the properties associated with the object, and the right shows you the current value of the property. The work area is where the modules (when you record a macro, the application creates a VBA project - a container into which is

put both the document and a special object called the module that contains the code) you work with are displayed. You will notice that the VBE has colour-coded your code - green for comments, blue for VBA keywords and all the other text is black. The VBE will also convert all VBA keywords to their proper case (e.g. if we entered end sub it would be converted to End Sub and displayed as blue text) and check for syntax errors.

So using the sample code lets analyse the anatomy of a VBA procedure. The first thing to notice is that all the code has been wrapped in the basic unit of VBA programming - the procedure. This procedure has the name we gave to the macro - BoldFirstSentence. The description we gave the macro has been translated to comments - lines that the computer ignores and just there for our information. Our VBA procedure manipulates the application's environment, specifically making a text selection bold. Items we can manipulate are called 'objects' in VBA. To VBA an 'object' is anything in an application you can see and manipulate in some way. For example an Excel range is something you can see, and you can manipulate it by entering data, changing colours, setting fonts, etc. A range is therefore an 'object'. Our VBA procedure uses the 'Selection' object. VBA objects have a number of properties - they control the appearance and position of the 'object'. There are a large number of objects and their associated properties, if we press F2 from within the VBE we will display the 'Object Browser' dialog box (see screenshot). As you can see from the code the notation used is 'Object.Property'. We can also have an optional 'Action'. So 'Selection.Characters.Count' would count (the action) the number of characters (the property) in the current text selection (the object).




```
Dim numCharacters
numCharacters = Selection.Characters.Count
MsgBox ("Number of characters is " & numCharacters)
```

As a starting point for your experiments we are going to add some code to the macro so it also displays the number of characters in the first sentence in a message box. Within the work area of the VBE add the lines of code in the box above the 'Selection.MoveRight...' line. Close the VBE and press **ALT-F8** to display the macros dialog box. Select your macro and run it. The 'Dim' keyword is used to allocate storage space for a variable - a memory location used to remember a value.

In the next part we will continue our look at VBA and its possibilities this time in Excel. We are all familiar with keyword searching with Google (http://www.google.com), but did you now you can also search news groups (http://groups.google.com), web logs (http://webblogs.google.com) and more. That's it for this week, please send any questions or comments to techexchange@endhousesoftware.com.

By Gavin Baker.