July, 2000

Advisor Answers

Accessing Word Bookmarks through Automation

Visual FoxPro 6.0/5.0/3.0 and Word 2000/97

Q: I've created a Word Template containing certain bookmarks. I'm able to write code in Visual FoxPro that creates a Word document based on this template. How can I substitute text or the contents of a variable for the bookmarks?

I know that in VBA, I'd use:

```
*--------------------------------------
Dim objBookmark As Bookmark
Dim objRange As Range
**In this place we create a word document (objNewDoc).
```

Then:

```
Set objBookmark = objNewDoc.Bookmarks("Subject")
**Keep in mind the bookmark "Subject" exists on the NewDoc.
Set objRange = objBookmark.Range
objRange.Text = "This is my subject!"
*--------------------------------------
```

What would be the VFP equivalent of that code?

–Ali Zahedi (via Advisor.COM)

A: In fact, the VFP version of the code looks quite similar to the VBA code. The differences are only in syntax. Before I get into that, let me go over some basics that you already know, but other readers may not.

Visual FoxPro is capable of serving as an Automation client. That means it can tell other applications what to do. Microsoft Word, like the other major Office applications, is an Automation server – it can be ordered around by Automation clients. (In fact, both Word and VFP can be either Automation client or server.)

To create a Word automation session, use the CreateObject() function, like this:

```
oWord = CreateObject("Word.Application")
```

The Application object is at the top of Word's object model. To manipulate Word programmatically, you use its properties and methods or the properties and methods of the various other objects in the object model, such as Document, Range, Paragraph, and so forth.

If you want to be able to see what you're doing as you go along, issue this line to make the Word session visible:

```
oWord.Visible = .T.
```

You indicated that you're taking advantage of one of Word's most useful capabilities – templates. A template is a document stored with a DOT extension. It can include styles, macros, boilerplate text and, as you've done, bookmarks in the text where you want to substitute. Once you store a document as a template, you can create new documents from it without affecting the template itself. Templates that are stored in certain locations appear in Word's File New dialog so that you can create new documents based on them.

To create a new document, call the Documents collection's Add method, passing the name of the template on which to base the document, including the full path. There are two default locations for templates, one for templates for the individual user and one for templates for the user's workgroup. You can retrieve those using the DefaultFilePath method of Word's Options object.

This code gets the path to the user's templates:

```
#DEFINE wdUserTemplatesPath 2
cTemplatePath = oWord.Options.DefaultFilePath( ;
                    wdUserTemplatesPath )
```

The line below creates a new document based on the Normal template. A reference to the new document is returned and is stored in the oDocument variable.

```
oDocument = oWord.Documents.Add( ;
               ADDBS(cTemplatePath) + "Normal.DOT")
```

In fact, using the Normal template is the default if you omit the parameter to the Add method. But I wanted to show an example that would work, no matter what templates were installed. (Note also that the templates that come with Word are not stored in the user templates path, but in a subdirectory of the Word installation directory.)

Now back to your problem. Once you've created a document based on your custom template, replacing your bookmarks with text is uncannily similar to the way it's done in VBA. Let's look at the code first, then talk about it.

```
oBookMark = oDocument.Bookmarks("Subject")
oRange = oBookMark.Range()
oRange.Text = "This is my subject!"
```

Bookmarks in Word let you name a section of the document and have Word keep track of it. A bookmark can indicate a portion with content or just a point in the document (that is, a place where text can be inserted). Bookmarks is a collection in Word of all the individual bookmarks in a document. (Word confuses matters a little because Document has a property named Bookmarks that is a reference to a collection also called Bookmarks. This type of overloading of terms is quite common in Word.) The first line of code finds the bookmark named "Subject" and stores an object reference to it in the variable oBookmark.

A range is a key concept in automating Word. It's an unnamed section of a document that can be as small as a point or as large as the entire document. Most of the work you do in manipulating documents through Automation is done through Range objects. The Range method of many Word objects lets you convert between other object types and Range objects. The second line of code creates a range from the Subject bookmark and stores a reference to it in the variable oRange.

The Text property of Range contains the textual contents of the Range. By changing that property, you change the document. So, the third line changes the Subject bookmark to contain the text "This is my subject!"

It's worth noting that the three lines of code can be reduced to a single (somewhat less readable) line, as follows:

```
oDocument.Bookmarks("Subject").Range.Text = ;
   "This is my subject!"
```

Word's object model is extremely rich. For the most part, anything that you can do interactively in Word can be done through Automation as well. The best resource for figuring out what objects, methods and properties to work with is the Word Visual Basic Help file. For Word 2000, it's called VBAWd9.CHM; for Word 97, it's VBAWd8.Hlp. In both cases, it's possible to install Word without installing this file, so if you

can't find it on your machine, run the Office or Word installation program again to add Visual Basic Help. For additional ideas on learning about the Office object models, see Dan Freeman's article in the July '99 issue of FoxPro Advisor.

–Tamar