July, 2003

Advisor Answers

Check Automation Servers

VFP 8/7/6

Q: I have code that automates the Office applications. Sometimes users close one of the applications and then my code fails. Is there a way I can check whether the server I opened is still running from my Automation code?

–Name withheld by request

A: When you automate the Office servers, as long as you keep them invisible, you have nearly total control over the server. (A knowledgeable user can open the Task Manager and kill the server that way.) But if you need users to interact with the Automation servers, you run the risk that the user will shut a server down before you're done with it. In that case, it's a good idea for your code to make sure the server is available before using it.

It turns out you can't use the same code for all of the Office servers. Word is the easiest. If your reference to Word is stored in oWord, the following code is sufficient to ensure that you have a good reference:

```
IF VARTYPE(oWord) = "O" AND TYPE("oWord.Name") = "C"
   * Word is running
ENDIF
```

It's not enough to check just that oWord refers to an object, because even after Word shuts down, the variable still has type “object.” Checking the Name property ensures that there’s still a good reference.

For Excel and PowerPoint, this test fails, however. That's because their executables stay in memory after the user shuts them down. In fact, Excel stays in memory even if you shut it down with the Quit method. (You can force the executable from memory by releasing the variable holding the reference.)

If you shut the server down in your code, presumably you know it, so the solution here is designed for the situation where the user closes the server. The trick is to keep track of the server's expected visibility

manually. Then, you can compare that to the server's actual visibility to determine whether it's still available.

To do this, create a variable or property to track visibility. When you make the server visible, set the variable or property to .T., like this:

```
oExcel.Visible = .T.
lIsExcelVisible = .T.
```

Then, to check whether the server is available use code like this:

```
lAvailable = .T.
IF ISNULL(oExcel)
   lAvailable = .F.
ELSE
   IF oExcel.Visible <> lIsExcelVisible
      lAvailable = .F.
   ENDIF
ENDIF
```

Outlook presents yet another picture. Like Excel and PowerPoint, its executable stays in memory even after you shut it down. But Outlook is remarkably tenacious and even after the user shuts it down or you call the Quit method, your object reference is still good. The only way to lose the object reference is to release the variable. So, to check whether you have a good reference, it's sufficient to test its type:

```
IF VARTYPE(oOutlook)  = "O"
   * You got it
ENDIF
```

To determine whether you have something visible in Outlook is another story. Outlook displays two kinds of items, explorers and inspectors. An explorer is the main Outlook interface, the one you see when you open Outlook interactively. An inspector is the interface for editing an Outlook item, such as an email message. For Outlook to be visible, at least one or the other must be displayed.

So, you can check whether Outlook is currently displaying something with code like this:

```
IF (TYPE("oOutlook.ActiveExplorer") = "O" ;
   AND TYPE("oOutlook.ActiveExplorer.Caption") = "C") OR ;
   (TYPE("oOutlook.ActiveInspector") = "O" ;
   AND TYPE("oOutlook.ActiveInspector.Caption") = "C")
   * Something's showing
ENDIF
```

The best way to deal with the overall problem generally is write a function or method that performs the test you need and call it each

time rather than using all this code every time you need to check your server reference. I like to work with a wrapper class for the Automation servers and I include an IsServerOpen method that encapsulates this test. Then, each method that depends on the server being available calls the IsServerOpen method.

–Tamar