

Re: Determine how Word was launched

Source:

<http://www.tech-archive.net/Archive/Office/microsoft.public.office.developer.vba/2006-04/msg00103.html>

- *From:* Mike Clayton <mike-c@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Fri, 28 Apr 2006 06:29:01 -0700
-

Hi Helmut,

Thanks for that. I think I've found a way to do it using the Windows API to inspect the command line used to launch Word. If the user launches Word from a shortcut the command line will just contain winword.exe (unless they've customised the shortcut). If they double-clicked a file the commandline will contain the name of the file.

My code is below in case anyone else finds it useful. Put it in a module in your addin and then add the following sub:

```
Public Sub AutoExec()  
If UBound(GetCommandLineArgs(GetCommandLine)) = 0 Then  
' started with no commandline args  
Else  
' started with commandline args – process them to decide what to do  
End If  
End Sub
```

Regards,

Mike

```
' ---- insert as a new module  
Option Explicit
```

```
' gets the commandline used to launch the current process  
Private Declare Function GetCommandLineA Lib "kernel32" () As Long
```

```
' processes a unicode commandline and turns it into a array of unicode  
' null-terminated strings. ensures that arguments with spaces in are  
' processed properly and not split into separate arguments. for example,  
' /a:"x y z" becomes "/a:x y z" as opposed to "/a:x", "y" and "z"  
Private Declare Function CommandLineToArgvW Lib "shell32" (ByVal lpCmdLine  
As Long, pNumArgs As Long) As Long
```

Re: Determine how Word was launched

```
' native Windows string and memory management functions
Private Declare Function lstrcpyA Lib "kernel32" (ByVal lpString1 As String,
ByVal lpString2 As Long) As Long
Private Declare Function lstrcpyW Lib "kernel32" (ByVal lpString1 As String,
ByVal lpString2 As Long) As Long
Private Declare Function lstrlenA Lib "kernel32" (ByVal lpString As Long) As
Long
Private Declare Function lstrlenW Lib "kernel32" (ByVal lpString As Long) As
Long
Private Declare Sub RtlMoveMemory Lib "kernel32" (pTo As Any, uFrom As Any,
ByVal lSize As Long)
Private Declare Function LocalFree Lib "kernel32" (ByVal hMem As Long) As Long

' get the command line used to launch the current process
Public Function GetCommandLine() As String
Dim lngApiResult As Long
Dim lngCmdLinePtr As Long
Dim strCmdLine As String
' get a pointer to the command line string
lngCmdLinePtr = GetCommandLineA()
' read the contents into a vb string
strCmdLine = PtrToStringA(lngCmdLinePtr)
' return the result
GetCommandLine = strCmdLine
End Function

' get a fully qualified path to the running application
Public Function GetExecutablePath() As String
Dim lngDataPtr As Long
Dim strExePath As String
Dim lngArgCount As Long
Dim lngArgIndex As Long
Dim lngArgPtr As Long
' calling CommandLineToArgvW with an empty string pointer retrieves the
name
' of the current executable instead of the args from a command line
strExePath = ""
lngDataPtr = CommandLineToArgvW(StrPtr(strExePath), lngArgCount)
If (lngDataPtr = 0) Then Err.Raise Err.LastDllError
' the full filename is split at each space character – e.g. "C:\Program
Files"
' becomes "C:\Program" and "Files" so we need to reconstitute them back
into
' a single path
For lngArgIndex = 0 To (lngArgCount - 1)
lngArgPtr = PtrToDWord(lngDataPtr + (lngArgIndex * 4))
strExePath = strExePath & PtrToStringW(lngArgPtr) & " "
Next lngArgIndex
' remove the trailing space
strExePath = Left(strExePath, Len(strExePath) - 1)
```

Re: Determine how Word was launched

Re: Determine how Word was launched

```
' return the result
GetExecutablePath = strExePath
End Function

' splits a command line into a zero-based array of individual arguments, the
first
' of which is the name of the exe the command line launched.
Public Function GetCommandLineArgs(cmdLine As String) As String()
Dim lngApiResult As Long
Dim lngDataPtr As Long
Dim lngArgCount As Long
Dim lngArgIndex As Long
Dim lngArgPtr As Long
Dim objArguments() As String
' process the command line into an argument array
lngDataPtr = CommandLineToArgvW(StrPtr(cmdLine), lngArgCount)
If (lngDataPtr = 0) Then Err.Raise Err.LastDllError
' extract individual arguments from array. note that the first argument
' contains the app name, which may or may not be a fully qualified path
ReDim strArguments(0 To lngArgCount - 1) As String
For lngArgIndex = 0 To (lngArgCount - 1)
lngArgPtr = PtrToDWord(lngDataPtr + (lngArgIndex * 4))
strArguments(lngArgIndex) = PtrToStringW(lngArgPtr)
Next lngArgIndex
' free the memory allocated by CommandLineToArgvW
lngApiResult = LocalFree(lngDataPtr)
If (lngDataPtr = 0) Then Err.Raise Err.LastDllError
' return the result
GetCommandLineArgs = strArguments
End Function

' read a 4-byte long from a pointer
Private Function PtrToDWord(ByVal ptr As Long) As Long
Dim lngValue As Long
If ptr Then
RtlMoveMemory lngValue, ByVal ptr, 4
PtrToDWord = lngValue
End If
End Function

' reads a pointer into a null-terminated ansi string into a vb string
Private Function PtrToStringA(ptr As Long) As String
Dim lngApiResult As Long
Dim lngStringLen As Long
Dim strResult As String
' get the length of the string in the buffer
lngStringLen = lstrlenA(ptr)
' initialise the result to receive the buffer (including an extra null
character)
strResult = String(lngStringLen, 0)
' copy the string we have a pointer to into our buffer
```

Re: Determine how Word was launched

```
lngApiResult = lstrcpyA(strResult, ptr)
If (lngApiResult = 0) Then Err.Raise Err.LastDllError
' strip off the null character at the end
strResult = Left(strResult, lngStringLen - 1)
' return the result
PtrToStringA = strResult
End Function

' reads a pointer into a null-terminated unicode string into a vb string
Private Function PtrToStringW(ptr As Long) As String
Dim bytBuffer() As Byte
Dim lngBufferLen As Long
If (ptr > 0) Then
' get the amount of memory needed to store the string
' (remember unicode chars are 2 bytes)
lngBufferLen = lstrlenW(ptr) * 2
If (lngBufferLen > 0) Then
' copy the pointer's data into a byte array
ReDim Buffer(0 To (lngBufferLen - 1)) As Byte
RtlMoveMemory Buffer(0), ByVal ptr, lngBufferLen
' convert the vuffer into a string and return the result
PtrToStringW = Buffer
End If
End If
End Function
```