

ProcBodyLine

The *ProcBodyLine* property returns the number of the line at which the body of a specified procedure begins in a standard or a class **module**.

tags:
Properties

Applies to ...

Object	Description
Module	The representation of a Basic script.

Syntax

module.*ProcBodyLine* (*ProcName*, *ProcKind*)

Returned values / Arguments

Argument	Type	Optional	Description	Returned type
module	Object		The concerned module.	Long
ProcName	String	N	The name of a procedure in the module.	
ProcKind	Integer	N	The type of procedure. It can be one of the following constants: vbext_pk_Get vbext_pk_Let vbext_pk_Proc vbext_pk_Set	

Remarks

- Lines in a module are numbered beginning with 1.
- The body of a procedure begins with the procedure definition, denoted by one of the following:
 - A **Sub** statement.
 - A **Function** statement.
 - A **Property Get** statement.
 - A **Property Let** statement.
 - A **Property Set** statement.
- The *ProcBodyLine* property returns a number that identifies the line on which the procedure definition begins. In contrast, the **ProcStartLine** property returns a number that identifies the line at which a procedure is separated from the preceding procedure in a module. Any comments that precede the procedure definition (the body of a procedure) are considered part of the procedure, but the *ProcBodyLine* property ignores them.
- The *ProcStartLine* and *ProcBodyLine* properties can have the same value, if the procedure definition is the first line of the procedure. If the procedure definition isn't the first line of the procedure, the *ProcBodyLine* property will have a greater value than the *ProcStartLine* property.
- If useful, insert the following constants in your own code:

```
Const vbext_pk_Get = 1           ' A Property Get procedure
Const vbext_pk_Let = 2           ' A Property Let procedure
Const vbext_pk_Proc = 0          ' A Sub or Function procedure
Const vbext_pk_Set = 3           ' A Property Set procedure
```

Error messages

Property 'ProcBodyLine' not applicable in this context
Argument nr. ... [Value = '...'] is invalid

See also

ProcCountLines
ProcOfLine
ProcStartLine

Example

Query the properties of a Basic module

```
Const cstModule = "myModule"
Const cstProc = "mySub"
Const vbext_pk_Proc = 0 ' A Sub or Function procedure
Const cstStringToFind = "some string"

Dim oModule As Object, sProc As String, iProcType As Integer
Dim vStartLine As Variant, vStartColumn As Variant, vEndLine As Variant, vEndColumn As Variant

Set oModule = Application.AllModules(cstModule)
With oModule
    DebugPrint "Name = " & .Name
    DebugPrint "# of lines = " & .CountOfLines
    DebugPrint "# of declaration lines = " & .CountOfdeclarationLines
    DebugPrint "Lines 26 to 31 = " & .Lines(26, 6)
    DebugPrint "# of lines in proc " & cstProc & " = " & .ProcCountLines(cstProc)
    DebugPrint "Start line in proc " & cstProc & " = " & .ProcStartLine(cstProc)
    DebugPrint "Start body line in proc " & cstProc & " = " & .ProcBodyLine(cstProc)
    ' Line 35 is located within procedure sProc (of type iProcType)
    sProc = .ProcOfLine(35, iProcType)
    ' Arguments are left uninitialized to consider the whole module
    If .Find(cstStringToFind, vStartLine, vStartColumn, vEndLine, vEndColumn) Then
        ' Found
    End If
End With
TraceConsole()
```

Bookmark this page » » [ProcBodyLine](#)