



DpuScan

Janich & Klass
Computertechnik GmbH



DpuScan 6.x

Referenzhandbuch

Copyrights

© 1997 bis 2021 Janich & Klass Computertechnik GmbH. Alle Rechte vorbehalten. Gedruckt in Deutschland. Die in dieser Dokumentation enthaltenen Informationen sind Eigentum der Janich & Klass Computertechnik GmbH. Ohne schriftliche Genehmigung der Janich & Klass Computertechnik GmbH begründen weder der Empfang noch der Besitz dieser Informationen irgendein Recht auf Reproduktion oder Veröffentlichung irgendwelcher Teile davon.

Warenzeichen

Alle Produktnamen und Logos sind Warenzeichen oder eingetragene Warenzeichen der jeweiligen Eigentümer.

Haftungsausschluss

Die Anweisungen und Beschreibungen in diesem Handbuch waren zum Druckzeitpunkt zutreffend. Wir behalten uns jedoch das Recht vor, sowohl Beschreibung als auch Produkt jederzeit ohne Benachrichtigung zu ändern. Nach dem derzeitigen Stand der Softwaretechnik ist es nicht möglich, Programme zu entwickeln, die unter allen Bedingungen in jeder Konfiguration fehlerfrei arbeiten. Die Janich & Klass Computertechnik GmbH übernimmt keinerlei Haftung für Defekte, die direkt oder indirekt durch Fehler dieses Handbuches, Weglassen von Informationen oder durch Unstimmigkeiten zwischen diesem Referenzhandbuch und dem Produkt entstanden sind.

Aktualität

Es ist möglich, dass im Internet eine neuere Version dieses Handbuches verfügbar ist. Wir empfehlen deshalb, die Version anhand des auf dieser Seite abgedruckten Datums mit der Version auf dem Internet zu vergleichen. Falls die Version im Internet neueren Datums ist, sollten Sie diese herunterladen und ggf. selbst ausdrucken.

Die aktuelle Version des DpuScan Referenzhandbuch finden Sie im Web unter:

<http://www.jkimaging.com/pdf/DpuScan-Referenzhandbuch.pdf>

Inhaltsverzeichnis

1 VBScript	4
2 Requirements for Using VBScript	5
3 Use case for VBScript	7
3.1 VBScript as a taskstep in the task (Process)	7
3.2 VBScript on a button or within a macro (Process)	7
3.3 VBScript as an Event handler (Event)	7
3.4 VBScript image selection change (Selection)	7
4 Designing the VBScript program	8
4.1 Internal Functions to VBScript	8
4.2 Exported Functions	8
4.2.1 Call of functions and order of call	9
4.2.2 Function parameters	9
4.2.3 Function's return value	10
4.2.4 Events that may occur (Broker like events)	11
4.3 The DpuScan-Object	12
4.4 Editor Shortcuts	12
5 Configuring the Plugin	14
5.1 Selecting the Plugin and its Subprofil	14
5.2 Entering VBScript program text	14
5.3 Special procedures for VBScript programming	15
6 Catching Runtime Errors, Debugging	17

1 VBScript

The functionality of DpuScan can be extended at several interface points to handle exceptional requirements. Interface points of DpuScan are available in the Event Rules as 'function call', in the output storage as 'broker' and in a more universal fashion as a 'Plugin'. A Plugin can handle special events (broker events, image selection), can be called on purpose (toolbar button, task list) and can do image processing work.

For creating a Plugin special programmer's knowledge is required. It also requires a development system with compiler and debugger.

The Script-Plugin is a tool giving the DpuScan administrator the opportunity to create a Plugin without the need for more than just DpuScan itself. From simple to more complex data processing: all this is possible with Script-Plugin.

Script-Plugin runs program scripts given in the programming language of VBScript.

VBScript function can be called

- from a task step
- from a button in the toolbar
- from a macro
- on an event, as for instance: changed image selection, broker event

Limitations

- Script-Plugin cannot perform image processing

Example of usage


- filter some patterns from an OCR result
- complex pattern search by Regular Expression
- data base access, reading and writing
- obtain a unique batch number from some number source
- complex mathematical operations

2 Requirements for Using VBScript

For being able to use VBScript the Script-Plugin must be selected into the Profile

Script Plugin JKScrp JKScriptPlg.dll

After that, exported functions from the VBScript can be called at any point where a Plugin call is possible. Events will be routed to the VBScript that is selected to the Script-Plugin in the Baseprofile.

To Load the Script –Plugin into the Baseprofile go into the **Baseprofile configuration**, choose tab **Process** and hit the  button for **Plugins**.

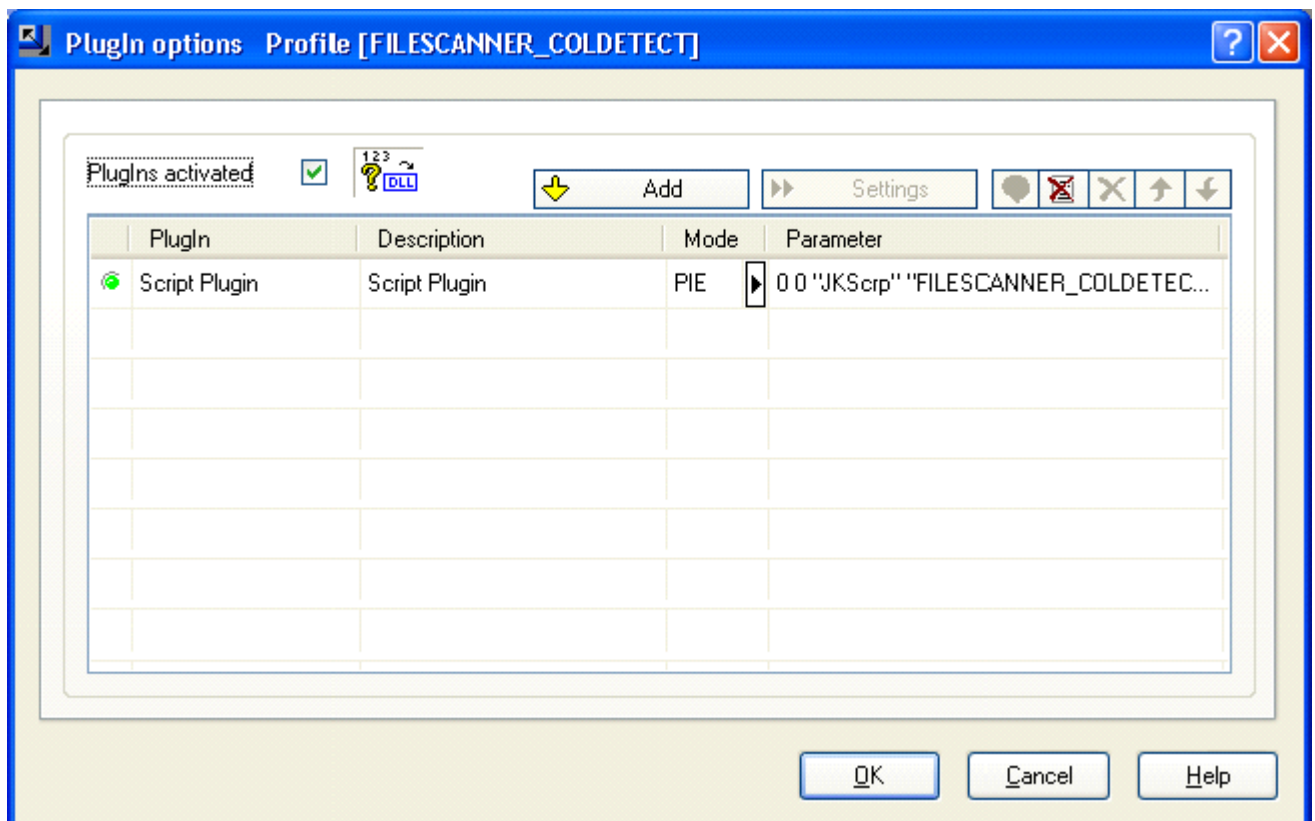


Image 1 – Adding the Script-Plugin

Add will open the list of available Plugins to choose from.

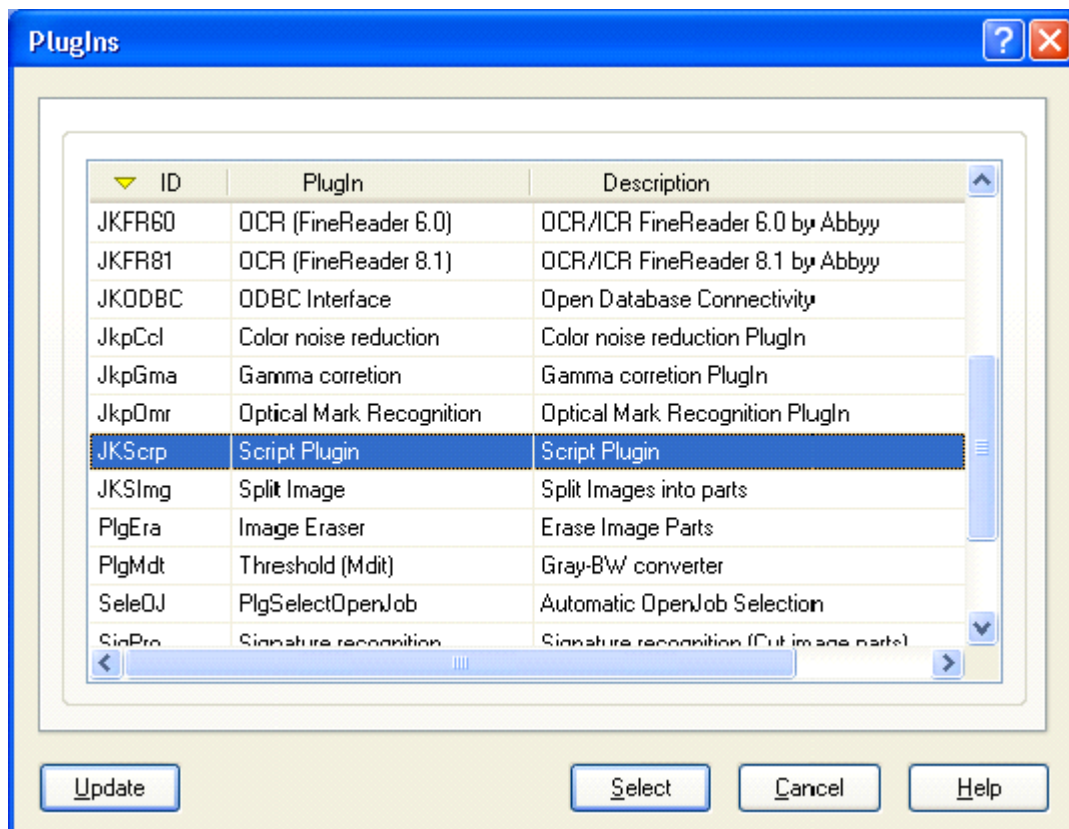


Image 2 – Plugin selection

Select the line of **Script Plugin** and hit the **Select** button.

Within **Plugin options** the button **Settings** will lead to the configuration of the selected VBScript. Select a predefined subprofile, or make a copy of it, or create a blank subprofile. Within the subprofile you can directly type in the VBScript program text, or you can load it from a file. The subprofile selected at this point will receive event handler calls from DpuScan.

3 Use case for VBScript

3.1 VBScript as a taskstep in the task (Process)

A call to VBScript can be placed at any position within the task list, but should not be placed before 'Load base profile' and not before 'Load user profile'. Calling VBScript is done by a **Call Plugin** command or **Call Plugin for every image**.

There is no limit on the number of tasksteps of this type. Every call can have its own VBScript assigned to it. Each distinct VBScript is stored as a subprofile for Script Plugin.

3.2 VBScript on a button or within a macro (Process)

VBScript can be called by command PluginParam01 .. PluginParam16, either as a toolbar button function, or from within a macro. Configuration rules are similar to that of a Plugin call with a taskstep.

3.3 VBScript as an Event handler (Event)

VBScript functions handling events must be defined in the one and only subprofile that is assigned to the Script Plugin within the base profile. Within the VBScript program script event handler functions are specially marked in their function header. Details are given in chapter 4.2

Events invoking VBScript event handler functions are e.g.

Task Start

Task End

Creating an imagefile

A detailed view on all events is given in chapter 4.2 too.

Reminder: Eventhandler only within **Baseprofile | Process | Plugin | Script Plugin**

3.4 VBScript image selection change (Selection)

Selection change is a special type of event and so is described in the chapter separately. The same rules as for event handlers apply. Event handler and selection handler must be located within the same VBScript program text.

The Selection catches the time where the new image has just been selected:

- all the I-variables for the new image are available
- returned results from functions are assigned to the I-variables of the newly selected image
- other variables not bound to an image can be modified in the same way

There is no event for Preselection yet.

Reminder: Event/Selection-Handler only within **Baseprofile | Process | Plugin | Script Plugin**

4 Designing the VBScript program

Within the VBScript program nearly every function and object can be used that is supplied either by the system (Windows) or by VBScript runtime. There is no object giving backward reference to DpuScan though. The interface from DpuScan is made by **exported Functions** within the VBScript and their return values. Exported functions have a special function header, that is unique to its DpuScan interface. The special function header maps DpuScan variables (%-Codes) to the parameters and the return value of the VBScript function.

4.1 Internal Functions to VBScript

Internal functions to VBScript are those functions that are called from within the VBScript only. There is no direct way from DpuScan to call these functions.

Internal functions can be defined as usual in VBScript

```
Function IncrCounter (ICounter)
    IncrCounter= ICounter+1
End Function
```

or

```
Private Function IncrCounter (ICounter)
    IncrCounter= ICounter+1
End Function

Sub MyMsgBox (dispText,dispHeader)
    MsgBox dispText,,dispHeader
End Sub
```

4.2 Exported Functions

Exported functions are those functions that are directly called by DpuScan.

Exported functions are defined with the extension of a special function header. Exported function should be created with the function generator tool of the Plugin configurator. The function generator automatically generates the special function header and this way assigns VBScript function parameters to DpuScan variables.

Within the function header a special parameter defines when to call this function. There are three possible cases:

\$\$ MODE Process	called by task, button or macro
\$\$ MODE Event=14	called on particular DpuScan event given
\$\$ MODE Selection	called on image selection change

MODE is set by the function generator. The proper number for the selected event is assigned to it.

Event handler functions for Event and Selection must be defined within the same VBScript code that is assigned to the base profile at **Base profile | Process | Plugin | Script Plugin**.

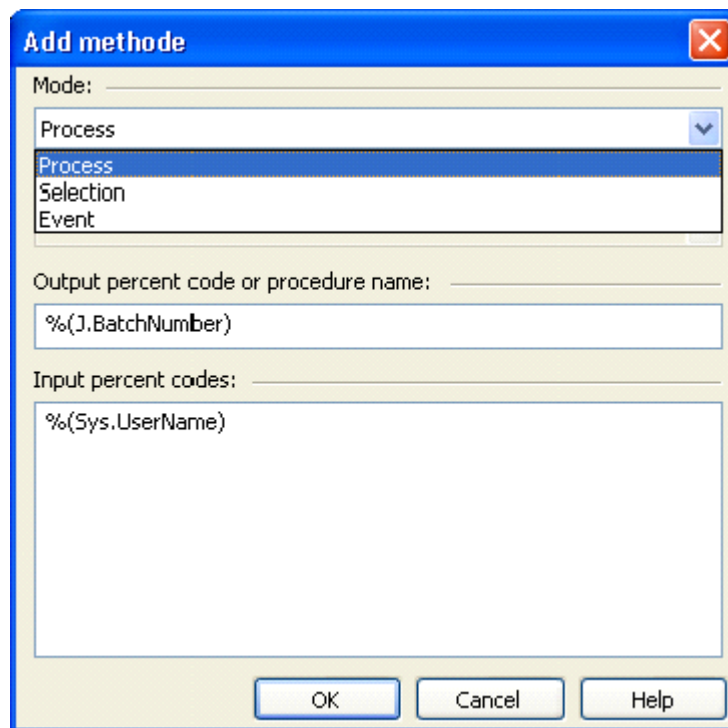


Image 3 – function generator

So within a DpuScan profile there may be
multiple Process VBScripts
one single Event/Selection VBScript

4.2.1 Call of functions and order of call

DpuScan invokes **all** exported Process functions with the same call of the Plugin.
DpuScan invokes exported function in the order given by their appearance within the
VBScript's program text.

Event functions are called with the occurrence of the event.

4.2.2 Function parameters

The parameters passed to the functions are of type string.
VBScript does some implicit type conversion. However, sometimes it is more safe to do
explicit type conversion:

```
Function Calculate(a)
'$$ DPUSCAN_DECLARATION_BEGIN
'$$ MODE Process
'$$ VAR a = %(I.A)
'$$ VAR Calculate = %(I.Res0)
'$$ DPUSCAN_DECLARATION_END

a = a + 1 ' converts a from string to numeric and increments the number
Calculate= a ' numeric will be converted to string by DpuScan
End Function
```

The parameters of those functions called simultaneously are fixed before the call to the first function. DpuScan variables do not change from calling the first function to calling the second and further functions. So the first function cannot pass a modified DpuScan variable to a subsequent function. All parameter passing between VBScript functions that are called simultaneously must be done within VBScript.

4.2.3 Function's return value

An exported function returns a single value. There are no var-type parameters. The return value is assigned to a DpuScan variable. To return multiple values, multiple function can be written, where the first functions does all the work, then stores partial results in global VBScript variables. Other function simply return the values of those global functions.

```

Dim Res1, Res2, Res3

Function QueryDB(a)
'$$ DPUSCAN_DECLARATION_BEGIN
'$$ MODE Process
'$$ VAR a = %(I.A)
'$$ VAR QueryDB= %(I.Res0)
'$$ DPUSCAN_DECLARATION_END
Dim lRes0, lRes1, lRes2

    InvokeSqlQuery(a,lRes0,lRes1,lRes2)
    Res1= lRes1 ' buffer to a global variable
    Res2= lRes2 ' buffer to a global variable
    QueryDB= lRes0
End Function

Function QDb_Res1
'$$ DPUSCAN_DECLARATION_BEGIN
'$$ MODE Process
'$$ VAR QDb_Res1= %(I.Res1)
'$$ DPUSCAN_DECLARATION_END

    QDb_Res1= Res1 ' read from global buffer variable
End Function

Function QDb_Res2
'$$ DPUSCAN_DECLARATION_BEGIN
'$$ MODE Process
'$$ VAR QDb_Res2= %(I.Res2)
'$$ DPUSCAN_DECLARATION_END

    QDb_Res2= Res2 ' read from global buffer variable
End Function

```

An alternative way to return multiple values by a single function call is to return an array. There are no array variables within DpuScan, so Script-Plugin spreads the array element to those DpuScan variables defined within the function header:

```

Function QDb_ResAll (par1,par2)
'$$ DPUSCAN_DECLARATION_BEGIN
'$$ MODE Process
'$$ VAR par1 = %(I.Par1)
'$$ VAR par2 = %(I.Par2)
'$$ VAR QDb_ResAll=%(I.Res0),%(I.Res1)

```

```
'$$ DPUSCAN_DECLARATION_END

Dim imResult(2)

    imResult(0)= par1+par2
    imResult(1)= par1*par2

    QDb_ResAll= imResult
End Function
```

The functions prototype can be created automatically if the DpuScan variable names are typed in comma separated.

Output percent code(s) or procedure name:—
 |(I.VAR1),(I.VAR2)|

```
Function IVar1()
'$$ DPUSCAN_DECLARATION_BEGIN
'$$ MODE Process
'$$ VAR IVar1 = |(I.VAR1),(I.VAR2)|
'$$ DPUSCAN_DECLARATION_END
Dim res(2)

    IVar1 = res
End Function
```

Types of return values

There are string types only defined in DpuScan. Return values given by the VBScript are converted to string where possible.

There is one special VBScript return type, that is NULL. If returned, the mapped DpuScan variable will not be changed.

4.2.4 Events that may occur (Broker like events)

DpuScan will call a function designated as an event handler for:

```
On Task Start (Task Start)
On Task End   (Task End)

On creation of a new batch (Batch Creation)

Prior to exporting a batch (Before Batch Export)
After exporting a batch    (After Batch Export)
Prior to finalizing a batch (Before Batch Finalization)
After finalizing a batch   (After Batch Finalization)

As Event rule              (As Event Rule)
```

These events occur at different times, depending on DpuScan's mode of operation:

- in Direct Mode – directly with the event
- in OpenJob Mode – on finalizing the batch

On processing an image group (Every Image Group)
 On first image of a batch (First Image)
 On last image of a batch (Last Image)

On creation of a new image file (File Creation)
 On closing of an image file (File Closing)

On creation of a new folder for image files (Path/Folder Creation)
 On closing a folder of image files (Path/Folder Closing)

4.3 The DpuScan-Object

To access certain properties and methods of the calling program, the DpuScan object is available in the script:

This object has these attributes and methods:

Attributes

DpuScan.Version provides the program version as a number

DpuScan.VersionString provides the program version as text
g

Methods

DpuScan.Sleep Milliseconds
 Waits the specified number of **milliseconds**

DpuScan.MsgBox Message, Buttons, Title
 Displays a message dialog corresponding to the appearance of the program

Message appears as text in the box.
Buttons shows buttons similar to MsgBox.
Title occupies the title line of the box.

The function returns a value corresponding to the pressed button
 See VBS-MsgBox.

DpuScan.Warning Text
 Gives a **text** in the DpuScan warning channel.
 This text is displayed in the DpuScan warning box.
 The warning does not appear in the configuration phase of the plug-in.
 The warning is not written to the log file.

4.4 Editor Shortcuts

strg-s Store in profile
Alt-s Store as file
strg-o Open from file

Alt-C	Perform syntax check
Alt-A	Add function/procedure
Strg-F	Find
Strg-G	Goto line
F3	Find next
F7	Goto next error line
F8	Goto previous error line

5 Configuring the Plugin

5.1 Selecting the Plugin and its Subprofil

To explicitly invoke VBScript, Script Plugin has to be selected at its point of invocation with the subprofile that holds the VBScript program text. An explicit invocation can be done within the tasklist or from a toolbar button or macro.

For invoking VBScript with the tasklist there are two commands

- Call Plugin
- Call Plugin for every image

Selected into the task step must be

- Script Plugin
- the subprofile that holds the VBScript program text

For invoking VBScript from a toolbar button there are commands

- PluginParam01
- PluginParam02
- ...
- PluginParam32

for a maximum of 32 buttons.

From within a macro an almost unlimited number of different VBScripts can be invoked. The command to insert is the same as for a toolbar button,


- PluginParam01..32

however, for each instance of a plugin call profile information is directly stored into the container of the macro. The index of the PluginParam is irrelevant. There is no conflict with a PluginParam placed into a toolbar having the same index..

These parameters are assigned to PluginParamXx

- Script Plugin
- subprofile holding the VBScript program text

5.2 Entering VBScript program text

At all places where a subprofile is assigned to the Plugin call, there is always the button to configure the selected subprofile.  It gives the opportunity to enter or change the VBScript program text.

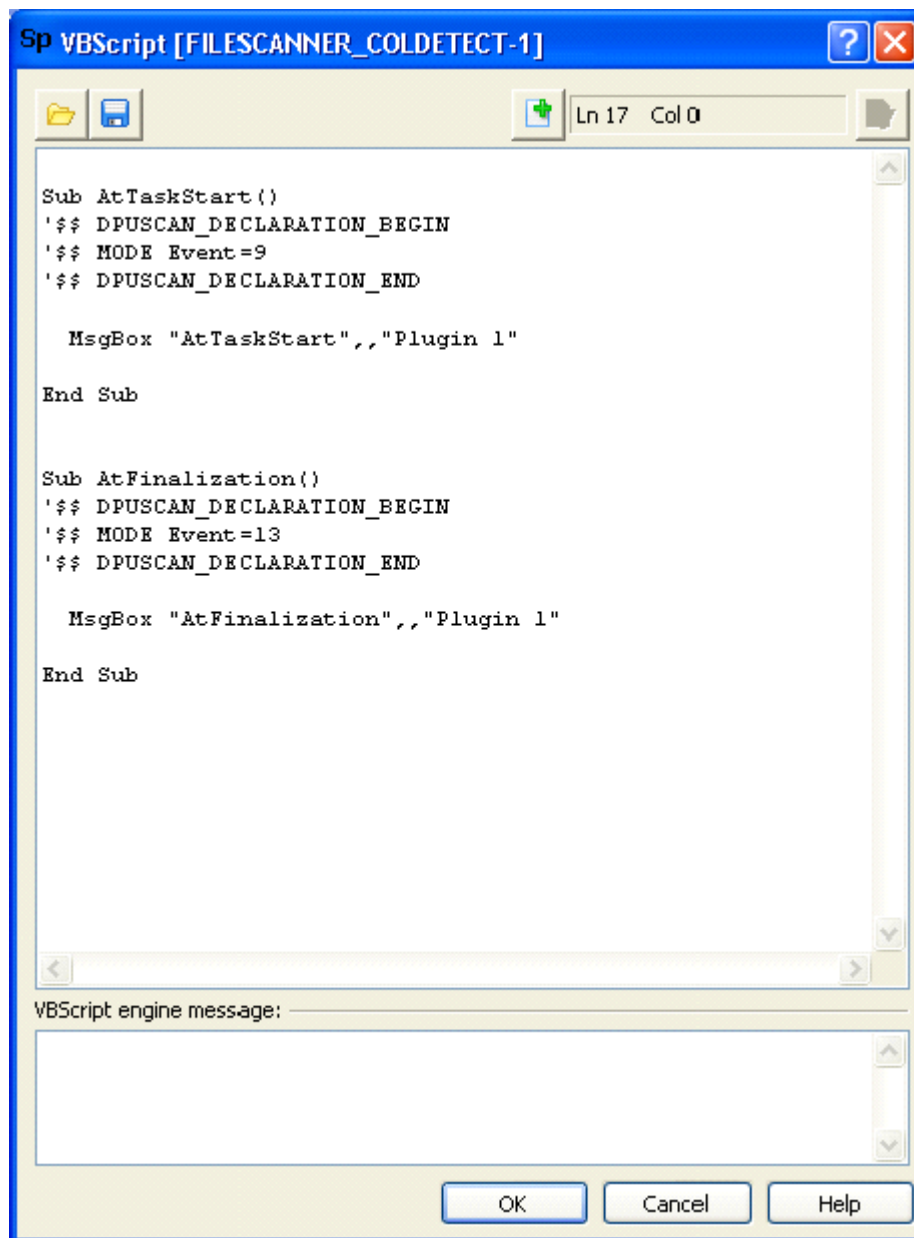




Image 4 – Entering VBScript program text

Beside directly entering program text there is a way to load from file or to store to file. The file is not part of the DpuScan configuration. It will not be used by DpuScan. Only the text stored in DpuScan subprofile will be used.

Button  invokes the function generator for exported VBScript functions. The function generator automatically generates functions of type Process, Event, Selection.

Button  performs a syntax check on the given VBScript program text.

5.3 Special procedures for VBScript programming

For doing file input and output, VBScript offers the FileSystem Object

```
Dim Fso
Set Fso = CreateObject("Scripting.FileSystemObject")
```

For accessing databases a useful object to work with is

```
Dim Conn
Set Conn = CreateObject("ADODB.Connection")
```

Regular Expression is a built-in object to VBScript

```
Dim Reg
Set Reg = New RegExp           ' Create a regular expression.
```

Where it is necessary to transfer information by HTTP on the system there might be available and thus accessible (or other versions of XMLHTTP)

```
Dim Http
Set Http = CreateObject("Msxml2.XMLHTTP.6.0")
```

XML processing can be done by

```
Dim XmlDoc
Set XmlDoc= CreateObject("Msxml2.DOMDocument.6.0")
```

A documentation to VBScript is available on Microsoft's MSDN

<http://msdn.microsoft.com>

Search for 'VBScript'.

Users of ADODB, XMLHTTP, DOMDocument will also find a documentation on MSDN.

6 Catching Runtime Errors, Debugging

Script Plugin passes any runtime error to the warning window of DpuScan, giving information on what error occurred at what line:



The configuration dialog of Script Plugin can directly store the VBScript program text into a file of type VBS. Windows can directly execute this file as a program. Simulating a DpuScan call to any exported function can be performed by placing a call to the function into the VBScript program text anywhere outside any function. Be sure to pass string type parameters to the function. Use MsgBox to display debug messages.

Index

- A -

Attributes 12

- D -

DpuScan object 12

- M -

Message dialog 12

Milliseconds 12

- S -

Script 12

Sleep 12

- V -

version 12

VersionString 12

- W -

waiting 12

Warning dialog 12