

Using Flow Control in ArtemiS

Introduction

Since version 9.0, the ArtemiS analysis software features automatic flow control. The flow control function allows complex program sequences to be defined and to be performed automatically. Fully automated sequences are possible as well as user interaction via dialogs and buttons. Program sequences are configured in an editor. In the online help of the flow control function, you will find some basic examples illustrating the use of flow control. This Application Note provides a more comprehensive example to demonstrate the many possibilities of the flow control function and thus goes beyond a simple introduction on the subject. The descriptions refer to ArtemiS version 10.00.100. The general proceeding also applies to older (as of ArtemiS 9.00.100) and newer versions. However, minor changes to the user interface are possible.

The example below not only demonstrates how ArtemiS is controlled via flow control, but also how it can connect to other programs. As examples, we will connect to the HEAD Recorder (version 1.00.800) and Microsoft® Excel®. The HEAD Recorder has its own flow control function, which can be synchronized with the ArtemiS flow control. Microsoft® Excel® is remotely controlled via the COM interface. Of course, connecting to other programs is possible as well, as long as they feature a COM interface like Excel®.

The following example represents an “end of line” test: A recording is made, which is then checked against a predefined tolerance scheme. The result of the check is presented to the user in a message window and also saved to an Excel® file. Even if this application example may not exactly match your needs, it still demonstrates the extensive application possibilities of the flow control function as well as its practical use.

In the following, first the necessary preparations for performing the “end of line” test and the configuration of the HEAD Recorder flow control are described. Afterwards, the programming of the ArtemiS flow control and the script elements for connecting to Excel® are shown.

Using Flow Control in ArtemiS	1
Introduction	1
Application Example: “End of Line” Test	2
Preparations	2
Configuring the HEAD Recorder Flow Control	2
Programming the Flow Control in ArtemiS	3
Script Elements for the Connection to Microsoft® Excel®	10
Script Block: Initialize Excel®	10
Script Block: Write to Excel®	12
Additional Notes	13

Application Example: "End of Line" Test

Preparations

After starting ArtemiS, first a new project must be opened, into which an analysis function of your choice, such as FFT (Average), must be inserted. Furthermore, for performing the "end of line" test, a tolerance scheme is required to check the recorded signals against. For creating such a scheme, use the Tolerance Editor and save the desired tolerance curve. The tolerance scheme must be compatible with the chosen analysis function. That means, for example, if the analysis is an averaged FFT, the tolerance scheme must be created with the appropriate units for level vs. frequency.

Configuring the HEAD Recorder Flow Control

Another precondition for the ArtemiS flow control to run correctly is a suitably programmed flow control procedure on the HEAD Recorder side. The signals to be subjected to the tolerance check must first be recorded with the HEAD Recorder. This requires a suitable flow control procedure within the HEAD Recorder. It contains several start and stop points, which make sure that HEAD Recorder flow control is synchronized to that of ArtemiS. A start point in the HEAD Recorder flow control always requires a corresponding stop point with the same name in the active flow control of ArtemiS and vice versa. At these points, the two programs mesh with each other and are synchronized. The complete flow control program of the HEAD Recorder is shown in figure 1.

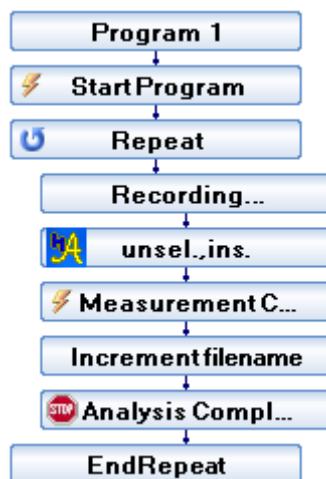



Figure 1: Complete HEAD Recorder flow control program for performing the "end of line" test

In this example, three synchronization points are required for the HEAD Recorder flow control: Two start points (marked with a lightning bolt icon) and a stop point (marked with a stop sign icon).


To create such a flow control program, open the default HEAD Recorder flow control program by opening the "Tools" menu and first selecting the command "Reset flow control" and afterwards the command "Flow control editor" to open the editor. The default flow control program must then be stopped for editing by clicking on the Stop button .

Now set the first start point via drag & drop directly after the block containing the program name. It will later make sure that the programs in the HEAD Recorder and in ArtemiS will start

synchronously. Therefore this start point is labeled "Start Program". The point can be given any other name, too, as long as the corresponding stop point in the ArtemiS flow control has the same name as the start point in the HEAD Recorder flow control. The name of the start or stop block can be changed in the properties dialog, which is shown as soon as a block has been selected with the left mouse button.

The next start point to be included in the HEAD Recorder program is called "Measurement Complete". Together with the corresponding stop point in the ArtemiS flow control, it makes sure that the analysis in ArtemiS does not start before the HEAD Recorder has completed a new measurement. This start point must be inserted into the HEAD Recorder flow control program after the ArtemiS block "unselect,insert".

The last pair of synchronization points is required for stopping the HEAD Recorder until the calculations in ArtemiS are completed. For this purpose, insert a stop point labeled "Analysis Complete" in the HEAD Recorder flow control program after the "Increment filename" block. The corresponding start point in the ArtemiS flow control must be placed after the blocks that calculate the analysis. The HEAD Recorder flow control stops at the stop point until the corresponding start point in the ArtemiS flow control allows the program to be continued.

After these modifications, the HEAD Recorder flow control program is complete and can be saved (e.g. by clicking on the Save button .

Programming the Flow Control in ArtemiS

Once the above preparations have been completed, we can now program the ArtemiS flow control. Figure 2 shows the complete program in the ArtemiS flow control required for the "end of line" test. This program synchronizes the HEAD Recorder flow control, performs a recording, conducts a tolerance check, displays its result and saves it to an Excel[®] spreadsheet.

In the following, the programming steps are described in detail. In the first section, the blocks for the program are put together and configured. The second section describes the Visual Basic scripts required for the connection to Excel[®].

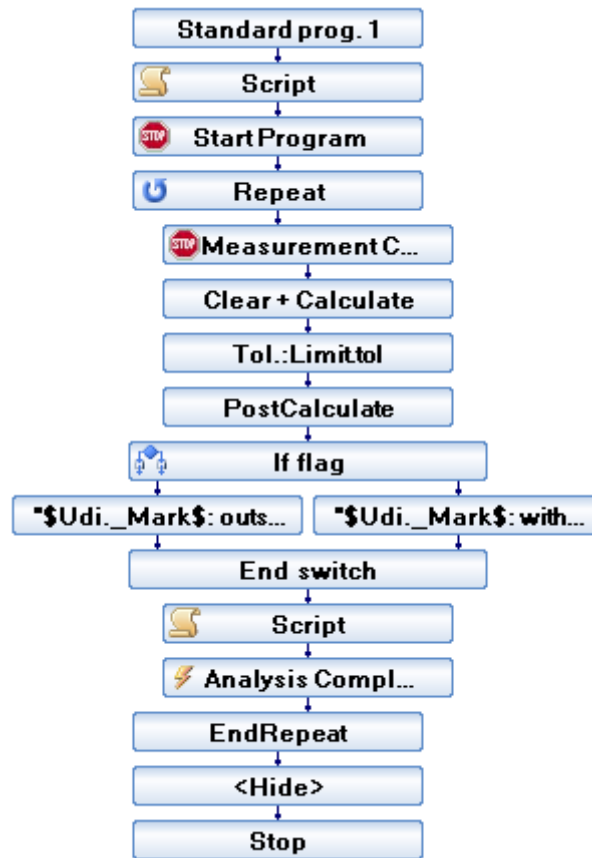


Figure 2: Complete ArtemiS flow control program for performing the “end of line” test

The editor for creating a flow control program in ArtemiS can be opened via the “Options” menu → “Flow control” → “Flow control editor”. When the editor is started for the first time, the window shown in figure 3 appears.

Just like in the HEAD Recorder flow control, the standard blocks can be dragged from the left side into the program on the right side and configured via the properties dialog shown below the program.

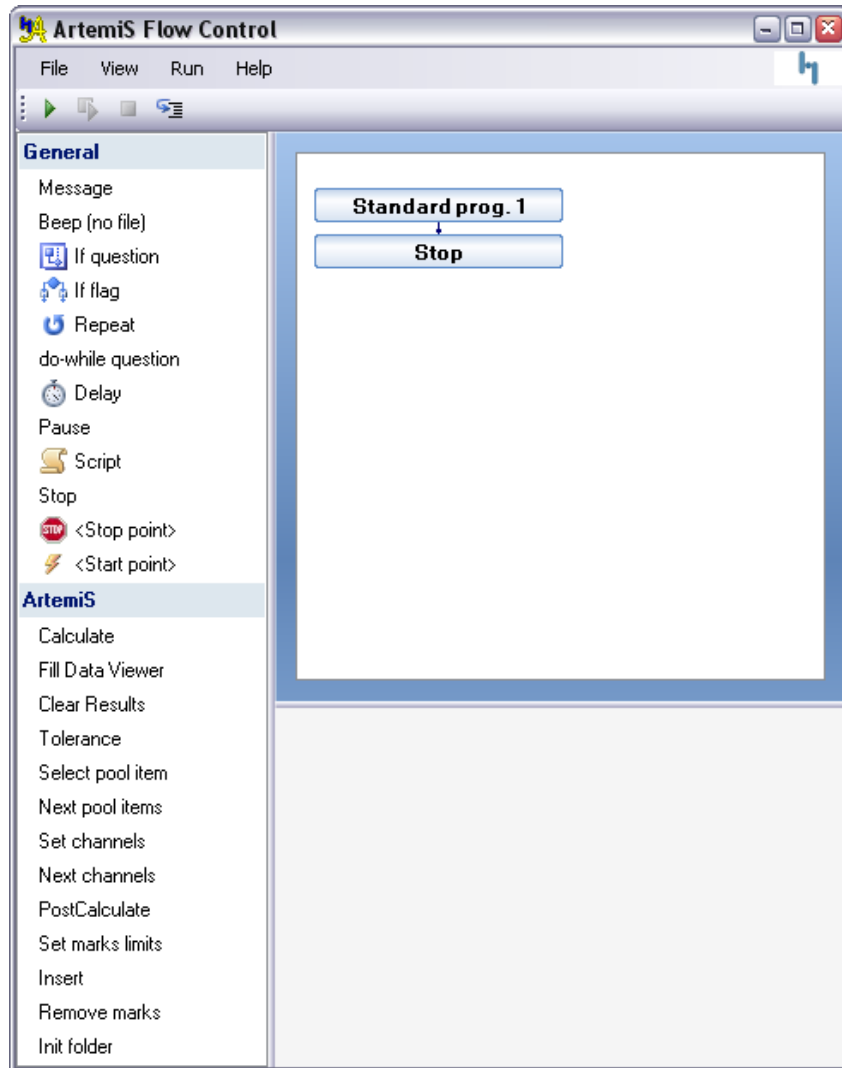


Figure 3: Flow control editor

To complete the flow control program in ArtemiS, proceed as follows: First, insert a stop point labeled “Start Program” directly after the first block. This is needed for the synchronized startup of both flow control programs (ArtemiS and HEAD Recorder).

After that, we can put together the program section for performing the calculation. The calculation must be enclosed by a “Repeat” loop in order to repeat it as often as required. For this purpose, insert a “Repeat” block after the start point. ArtemiS will automatically add the corresponding “EndRepeat” block as well. Between the blocks “Repeat” and “EndRepeat”, we can now insert the steps for the actual analysis. As shown in figure 4, the “Infinitely” option must be enabled in the properties of the “Repeat” block to make sure that the execution of the calculation is not aborted after a predefined number of repetitions.

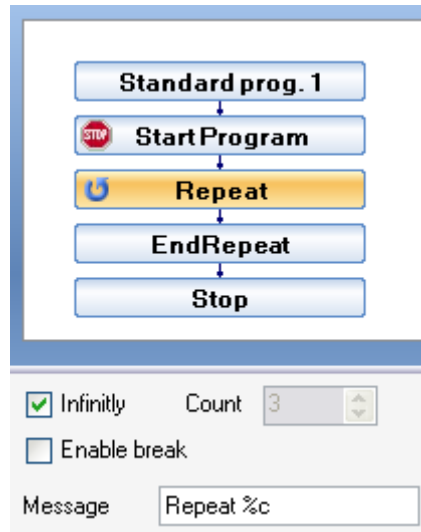


Figure 4: Properties of the "Repeat" block

Within the "Repeat" loop, we will now add the blocks for the analysis calculation and the tolerance check. For the calculation of the analysis, insert a "Calculate" block after the "Repeat" block. In the properties dialog of this block, you need to select the appropriate ArtemiS project (the one you want the calculation to be made for) from the list of currently open projects. In the project you choose, an analysis suitable for the tolerance check should be active. Activating the function "Clear old results" will delete any old analysis results in memory. It makes sense to apply this function, because the program is to be run several times in succession, and this function prevents new results from being mixed up with old ones.

After the calculation block, now insert the block for the tolerance check. In the properties of this "Tolerance" block, you can select the tolerance curve. Furthermore, you can specify the names for the maximum tolerance violation and for its position.

Figure 5 shows the additions described above.

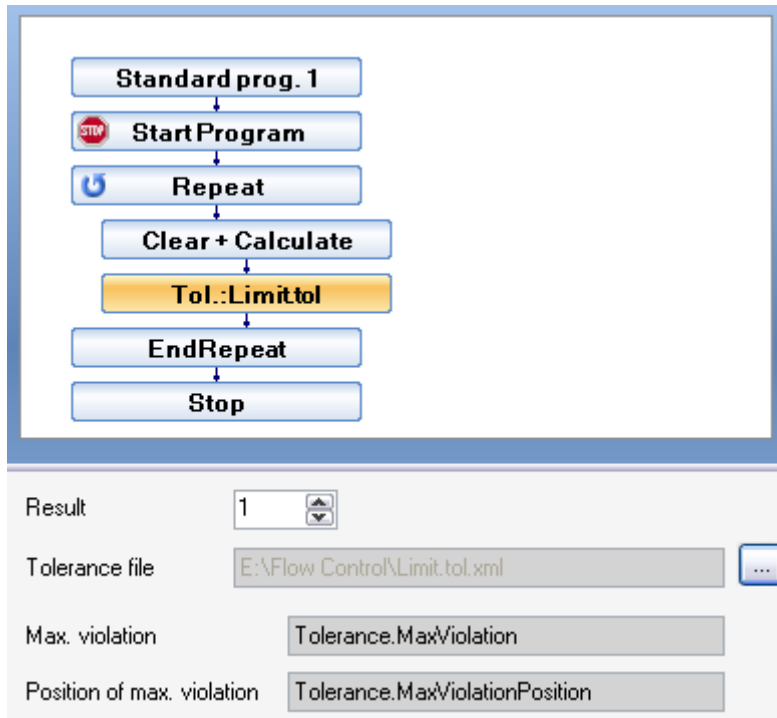


Figure 5: Flow control program showing the properties of the “Tolerance” block

In the tolerance check, a flag is set if a tolerance violation is detected. Further on in the program, this flag can be polled with the “If flag” block, which allows a differentiation between “tolerance violated” (flag set) and “tolerance not violated” (flag not set).

In order for a tolerance violation to be indicated and to be logged in Excel® later, the name of the file for which the tolerance check was performed is required. This name can be provided via the “Post-Calculation” block. Insert this block after the block for the tolerance check. In the properties dialog you can then select the desired information that should be available in the further course of the program, e.g. to be displayed. In this example, we need the user information, which includes the file name and the analysis name (see figure 6).

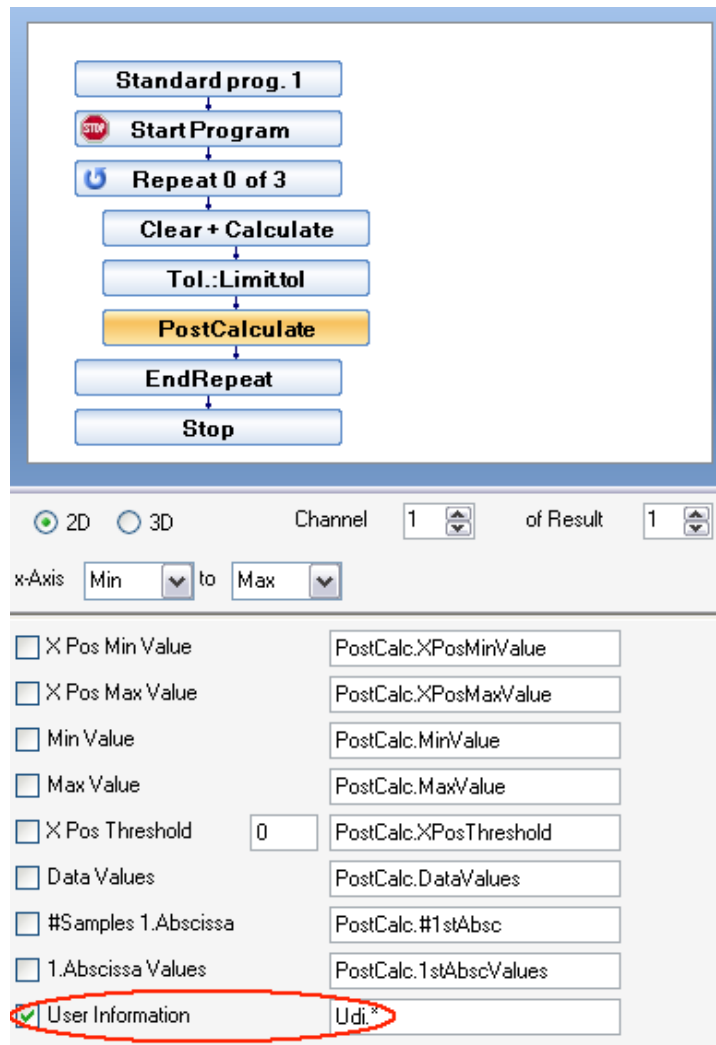


Figure 6: Properties of the "PostCalculate" block

Once the information about the tolerance check and the file name are available, they can be presented to the user in a window. For this purpose, we will add two "Message" blocks. One message contains the file name and the text "outside tolerance", and the other shows the text "within tolerance" after the file name. The text can be specified in the properties dialog of the "Message" block. In order to differentiate whether a file has passed the tolerance check, we first need to insert an "If flag" block after the "Post-Calculation" block. It splits the subsequent actions into two separate branches. The left branch is processed if the tolerance violation flag was set, whereas the right branch is performed if there was no tolerance violation.

Figure 7 shows the extended flow control program. You can also see the properties dialog of the left "Message" block. In the "Message" field, you can enter the text to be displayed. The variable "\$Udi._Mark\$" polls the file name information and displays it. After that, enter the text "outside tolerance" into the field. In the properties dialog of the right "Message" block, enter the same variable "\$Udi._Mark\$", this time followed by the text "within tolerance".

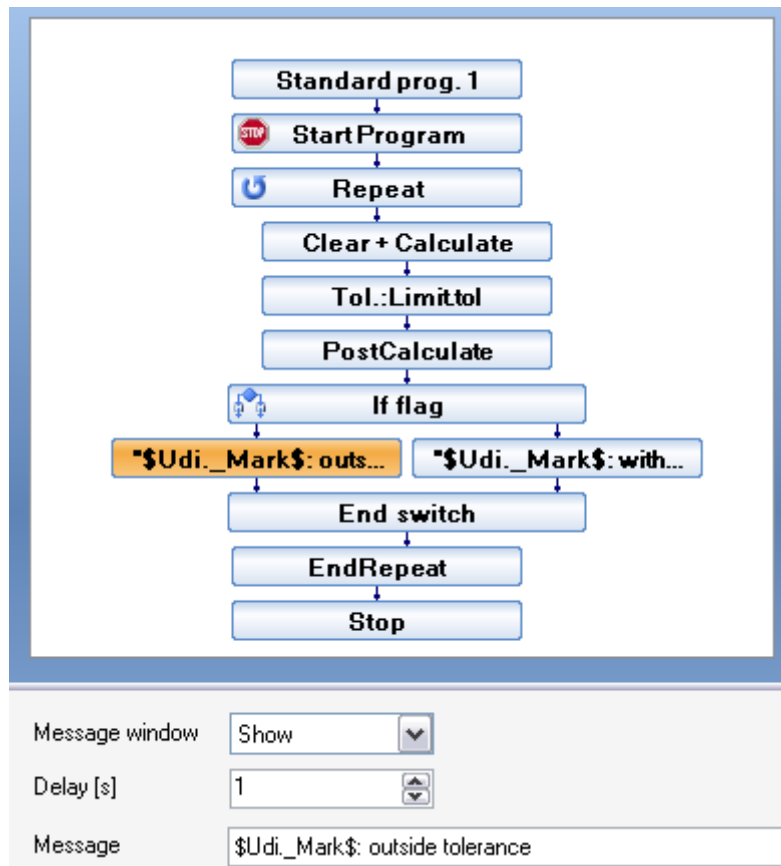


Figure 7: Flow control program with properties of the “Message” block

To close the message window, insert another “Message” block after the “EndRepeat” block. In the properties of this block, activate the “Hide” function (see figure 8). This block causes the message with the tolerance check result to disappear.

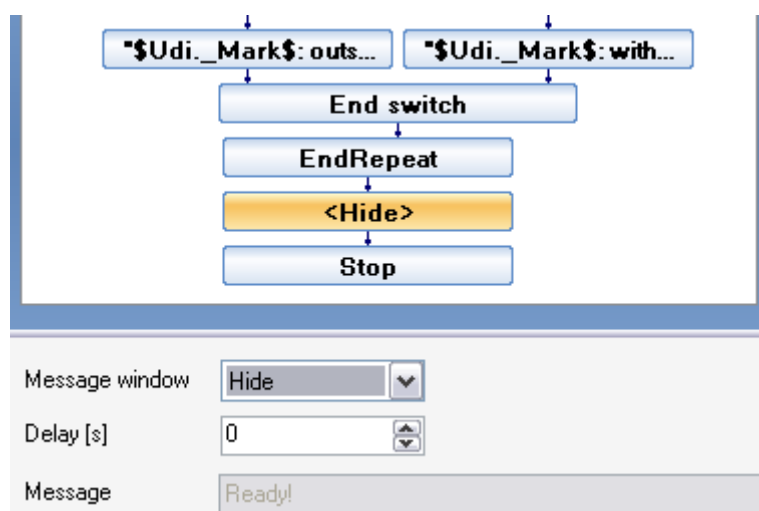


Figure 8: Activating the “Hide” function of the “Message” block

For the next step, we will insert the remaining start and stop points required for the synchronization with the HEAD Recorder. Directly after the “Repeat” block, add a stop point labeled “Measurement Complete”, which puts the ArtemiS flow control on hold until a measurement was rec-

ordered by the HEAD Recorder. The start point “Analysis Complete” must be inserted after the “End switch” block. It informs the HEAD Recorder that the analysis calculation has been completed and a new measurement can be made. Figure 9 shows the flow control program with all necessary start and stop points.

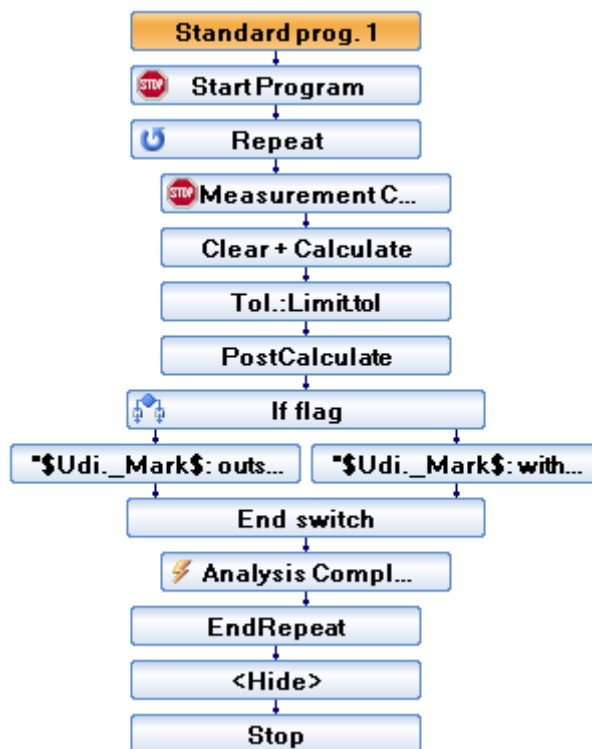


Figure 9: Flow control program with all required start and stop points

Script Elements for the Connection to Microsoft® Excel®

In order to connect to Excel®, two script blocks are required that specify a Visual Basic script in their properties dialog. The scripts are performed each time the respective script block is called in the flow control program. The first element is placed at the beginning of the program and serves for the initialization of a running Excel® application (block name: “Initialize Excel®”). The second script block is inserted after the “End switch” block and serves for the logging of the analysis results to an Excel® spreadsheet (block name: “Write to Excel®”). When creating the two scripts, you can make use of predefined “code snippets” that can be selected via the context menu of the properties dialog. They contain small code sections that can be customized and put together into larger code sections. They can be used as templates when creating your own programs.

Script Block: Initialize Excel®

The initialization of an already running Excel® application is done with the code snippet “Get running Excel” (see figure 10).

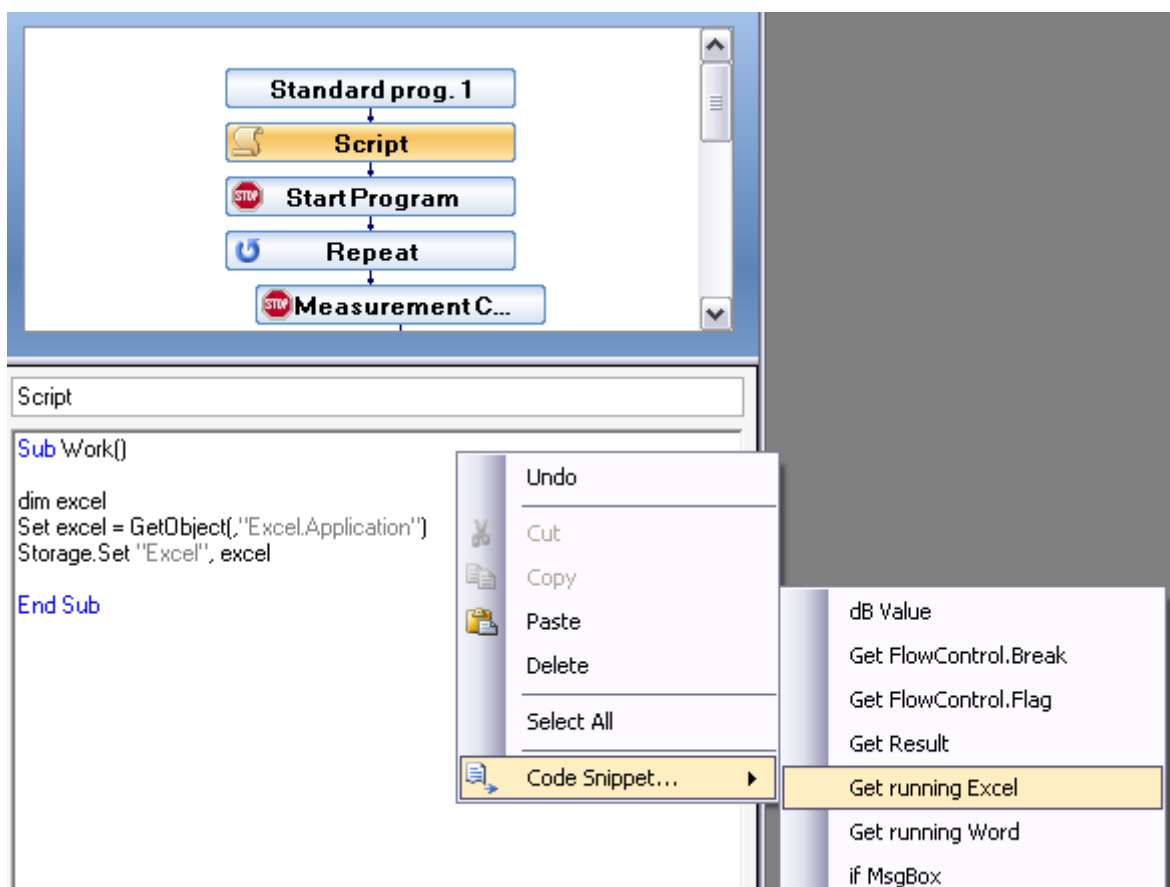


Figure 10: Inserting the code snippet “Get running Excel”

After that, two variables must be defined, which will be used to access the target cells in the Excel® spreadsheet. There is a suitable code snippet for this purpose, too. It is labeled “Storage.Set”. After inserting it, you will see the definition of the variable “MyVar” in the script window. This code section is required twice. In one section, change the variable name to “cellrow”, in the second one to “cellcol”. Then set both variables to 1 (see figure 10). In the further course of the program, “cellcol” specifies the column in the Excel® sheet and “cellrow” specifies the row. The complete code should now read as follows:

```

Sub Work()
dim excel
Set excel = GetObject("Excel.Application")
Storage.Set "Excel", excel
} Initialization of the running Excel® application

dim cellrow
cellrow = 1
Storage.Set "cellrow", cellrow
} Definition of the variable "cellrow"

dim cellcol
cellcol = 1
Storage.Set "cellcol", cellcol
} Definition of the variable "cellcol"
End Sub

```

Script Block: Write to Excel®

In addition to the script created above, which is run only once, a second script is required, which is run after each measurement and analysis. This second script writes the results of the tolerance check to the active spreadsheet in the running Excel® application.

The second script block must be inserted after the “End switch” block and should contain the following code:

```

Sub Work()
dim excel
set excel = Storage.Get ("Excel")
dim workbook
set workbook = excel.Workbooks(1)
dim worksheet
set worksheet = workbook.worksheets(1)
dim cellrow
cellrow = Storage.Get ("cellrow")
dim cellcol
cellcol = Storage.Get ("cellcol")
dim flag
flag = Storage.Get("FlowControl.Flag")
worksheet.cells(cellrow,cellcol) = storage.get("Udi._Mark")
cellcol = cellcol + 1
Storage.Set "cellcol", cellcol
worksheet.cells(cellrow,cellcol) = storage.get("Udi._Analysis")
cellcol = cellcol + 1
Storage.Set "cellcol", cellcol
if flag = TRUE then
worksheet.cells(cellrow,cellcol) = "outside tolerance"
cellcol = cellcol + 1
Storage.Set "cellcol", cellcol
worksheet.cells(cellrow,cellcol) = storage.get("Tolerance.MaxViolation")
cellcol = cellcol + 1
Storage.Set "cellcol", cellcol
worksheet.cells(cellrow,cellcol) = storage.get("Tolerance.MaxViolationPosition")
else
worksheet.cells(cellrow,cellcol) = "within tolerance"
end if
cellcol = 1
Storage.Set "cellcol", cellcol
cellrow = cellrow + 1
Storage.Set "cellrow", cellrow
End Sub

```

Call the running Excel® application, the Excel® file and the Excel® spreadsheet

Write the file name to the Excel® sheet
Move cell focus to the right

Write the analysis name to the Excel® sheet
Move cell focus to the right

If flag is set, enter “outside tolerance”
Move cell focus to the right

Write maximum tolerance violation
Move cell focus to the right

Write position of maximum tolerance violation

If flag is not set, enter “within tolerance”

Set cell focus back to the first column

Move cell focus down one row

The code section for calling the running Excel® application can be created using the code snippet “Use Excel”. This snippet contains four sections, of which only the first three are needed in this case. They specify a certain spreadsheet within the running Excel® application. In the default

code, the respective first spreadsheet etc. is called. The definition of the cells is not needed at this time and can be deleted. The target cell is selected via the variables "cellcol" and "cellrow". In order to access the variables already defined in the first script element, we will use the code snippet "Storage.Get". To adapt the default code to our needs, change the variable name "MyVar" to "cellrow" or "cellcol", respectively. For the definition of the final required variable, another code snippet is available. It is called "GetFlowControl.Flag" and polls the flag that is set whenever a tolerance violation is detected.

Once all the variables have been defined correctly, we can begin with the actual programming for the logging of the analysis results. In our example, the following information should be put together in one row of the spreadsheet: file name, name of the analysis, result of the tolerance check ("outside tolerance" or "within tolerance"), the maximum tolerance violation and its position. To prevent the information from being overwritten, after each entry the target area must be moved by one cell. This is done by incrementing the variable "cellcol". Once all the information has been written, the variable "cellrow" is incremented, too, so the entries for the next recording are written to the next empty row.

With the code section

```
worksheet.cells(cellrow,cellcol) = storage.get("Udi._Mark")
```

the file name is written to the first cell of the Excel® sheet. Afterwards, the code

```
cellcol = cellcol + 1  
Storage.Set "cellcol", cellcol
```

moves the target area to the right by one cell.

All other desired information is treated in the same way and written to the Excel® sheet.

For evaluating the tolerance check, the flag is used to create an "If" fork, which allows a differentiation between the two cases "tolerance violated" and "tolerance not violated". To create this fork, you can use a code snippet labeled "if then". At the beginning of the "If" fork, first the condition whether the tolerance violation flag has been set is checked ("if flag = TRUE"). If this is the case, first the tolerance violation message ("outside tolerance"), the corresponding value ("Tolerance.Max Violation") and its position ("Tolerance.MaxViolationPosition") are written to the Excel® sheet.

If the flag is not set, the code in the "Else" branch is executed instead of the code in the "If" branch. In this section, only the fact that the tolerance was kept is logged ("within tolerance").

After the "If" fork, the variable "cellcol" must be reset to "1", and the variable "cellrow" must be incremented. That way, the information for the next recording is written to the next row of the Excel® sheet.

Additional Notes

Please take care of the following important notes when using the flow control function in ArtemiS:

- The possibilities of using script elements are very extensive and are only briefly introduced in the above example. Of course, it is also possible to write your own code from scratch instead of using code snippets. A good introduction to the Visual Basic programming language can be found at the website <http://msdn2.microsoft.com/en-us/library/sx7b3k7y.aspx>, which includes a VB Script user manual. The site

<http://msdn2.microsoft.com/en-us/library/d1wf56tt.aspx> provides lots of additional code examples, which can be useful for writing your own programs.

- The above code for connecting to Excel[®] requires that the Excel[®] application is already running. You could also use a different code to launch the Excel[®] application from within your script. That code would look about like the following:

```
Sub Work()  
  
dim Excel  
  
on Error Resume Next  
Set Excel = GetObject("Excel.Application")  
  
if isEmpty(Excel) then  
set Excel = createObject("Excel.Application")  
set w=Excel.workbooks.open("c:\program_path\name.xls")  
  
end if  
End Sub
```

- Great care is required when defining variables. A typing error can cause the entire script to run incorrectly. A useful tool in this context is the "Option Explicit" function. It must be inserted at the very beginning of the script and makes sure that only variables can be used that have been expressly defined before. Otherwise an error message is displayed. This means that if a variable is called by a wrong name due to a typing error (e.g. "cellcoll" instead of "cell-col"), an error message appears. Without the "Option Explicit" function, the script would be continued without warning, which would lead to errors in other places that would be harder to trace.
- The HEAD Recorder can be configured so that the measurement starts immediately as soon as a certain trigger condition is met. In the properties dialog of the "record" block, the option "StartImmediately" can be set to "True" for this purpose. The desired trigger condition can then be configured in the Trigger window.
- The block "Increment filename" in the HEAD Recorder flow control can be configured in its properties dialog so that before each recording, a select box for specifying a file name is opened. That way, file names can be created with information about the current configuration during the recording. If this option is used, the command order in the Recorder flow control must be changed so that the "Select filename" block is placed before the "Record" block. It is also possible to save other user-defined information in addition to the file name and to read them for the output.

Other possibilities to use the flow control function as well as detailed descriptions of the individual blocks can be found in the online help of the ArtemiS flow control.

Regarding the HEAD Recorder flow control, we have written another Application Note for you. You can find it, for example, in the download area of our website. Of course, we can also send it to you by mail.

Do you have questions for the author? Please contact us at imke.hauswirth@head-acoustics.de. We look forward to your feedback!