Script language plugin for Pixelman

This script was created in order to help users create more complicated measurements. Programming skills are not required. Plugin processes the text file which includes commands. User can choose file by pressing “Select script” button.

By pressing the ”Run scipt” button, execution of script processing is started. The script is parsed and commands processed.

Executed line is written in panel.

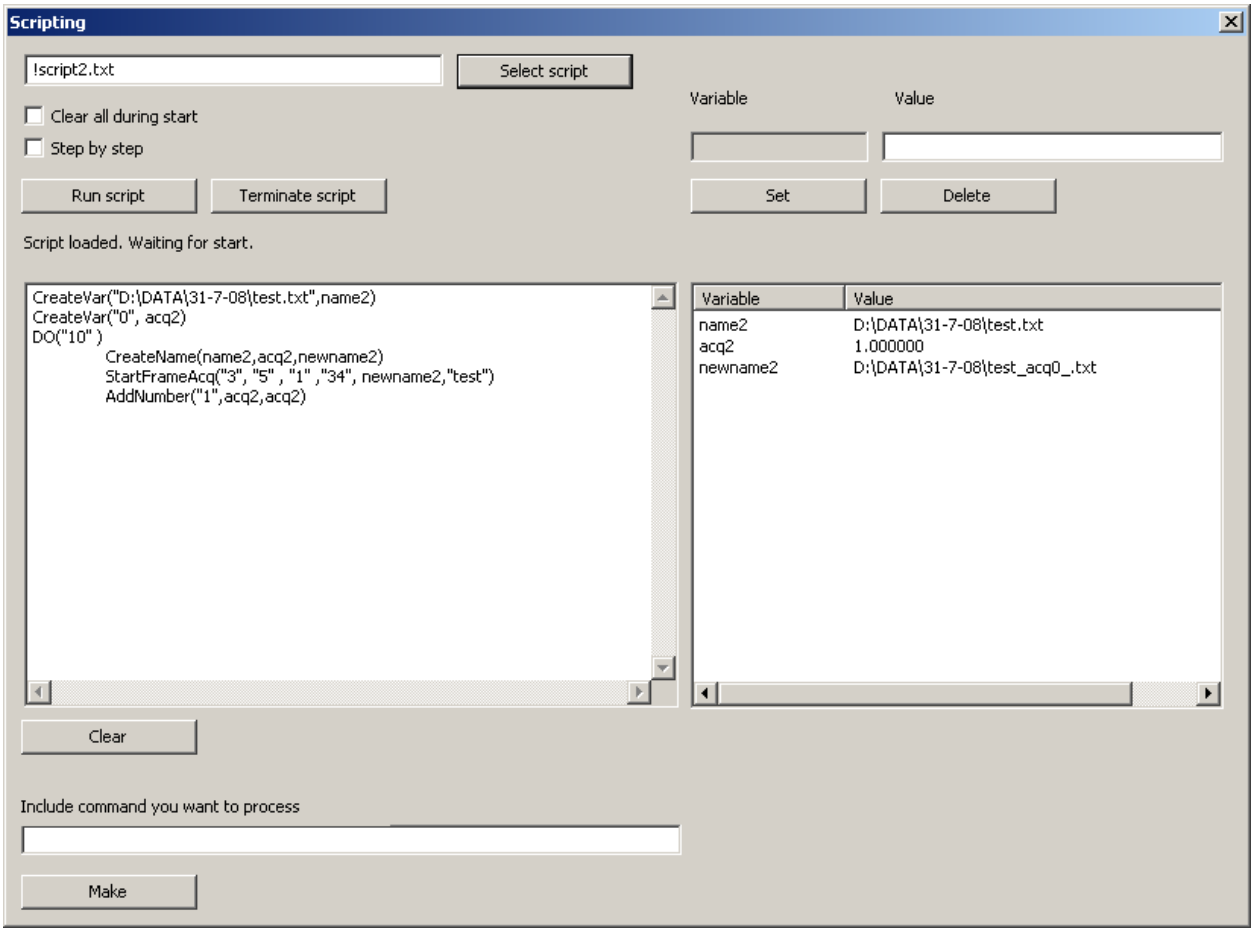

Fig. 1 Panel of the plugin

User can terminate script processing by pressing “Terminate script” button.

When the error occurs, the script processing is terminated and error message appears.

User can choose variable in list of variables. Left click on the variable enables variable to be changed or to be deleted. (Fig. 2)

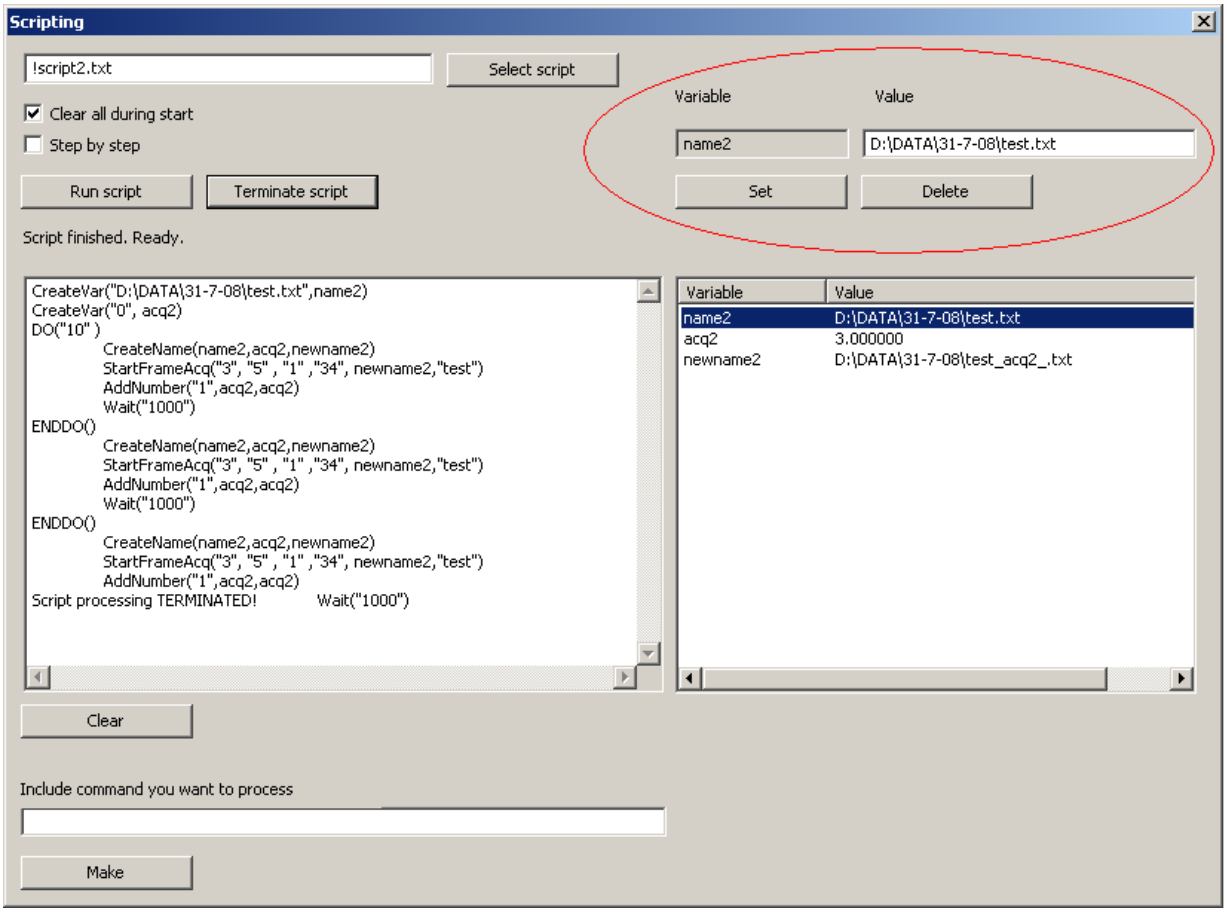

Fig. 2 Variables could be changed or deleted

When is the button “Clear all during start” checked, all variables and lists are deleted in the beginning of the next script processing.

Button “Step by step” can be used to execution of the script in steps. When it is checked the “ Run script” button changes to “Next line”. By pressing it the next line of script is executed.

For debug purposes one can use edit box “Include command you want to process”. Command which is written to this line is executed after pressing “Make” button. This is very handy for plugin developers. (Fig. 3)

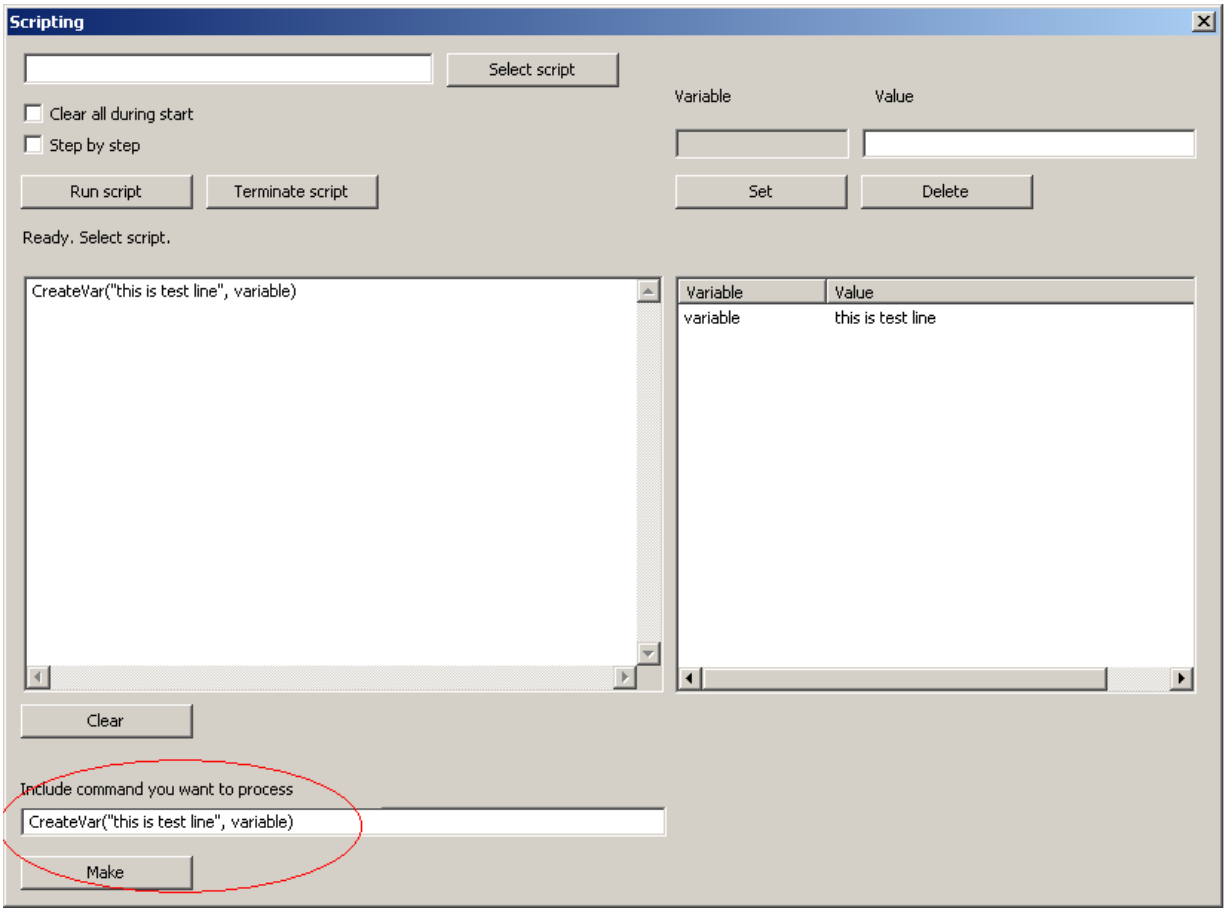

Fig.3 Execution of the single command

The syntax of the script is:

THE_NAME_OF_FUNCTION_OR_COMMAND(input parameters, output parameters)

The name of function is case sensitive. After this the function parameters are expected. The parameters are separated by commas. First are expected input parameters, then output parameters. One can see the expected parameters in ShowManagedItems panel in Pixelman. The output parameters are NOT mandatory.

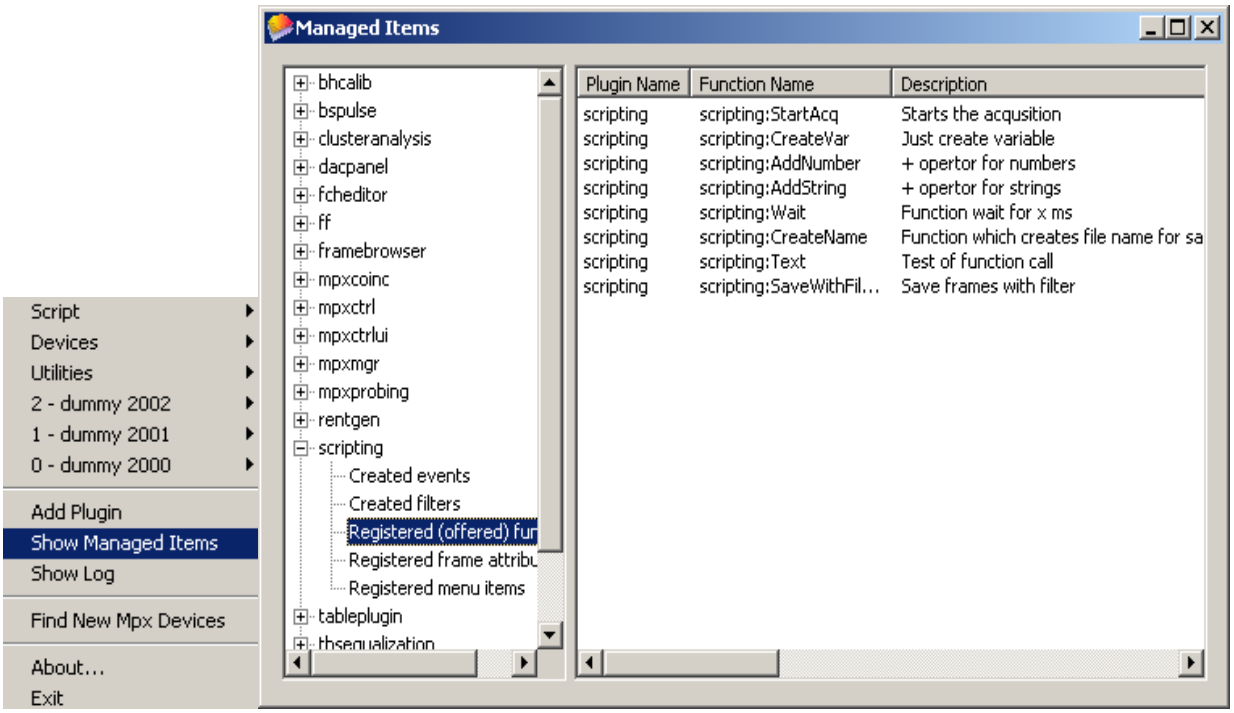

Fig. 4 Show Managed Items

When is the parameter closed between two “”, it is considered to be value (if the value is pure number, the quotations are not mandatory). Without quotation marks is parameter considered to be a variable

The retype of variables and values is done automatically.

The user can create variables, which can be used as output/input parameters for function.

The variable can be used as input parameter only after initialization, which means that variable has to be used first as output parameter.

As the output parameter user can use new variable which will be created and the value from function is included. Be careful and include right number of input parameters.

Besides the function user can use some commands such as:

IF command: makes if statement, with = mark and compare two values, it ends with ENDIF()

DO command: makes cycle of lines between DO () and ENDDO() , the value inside parentheses means how many time the cycle will be processed (e.g. DO(3) means that the cycle will be done 3 times)

Every script has to end with line ENDSCRIPT().

Here is the list of some precreated functions:

CreateVar(“value”, name) – just dummy function which creates the variable with the name and fill it with value

AddNumber(“value”,variable,newvariable) – adds to numbers and put them to new variable (it is not necessary to be the new variable : AddNUmber(number, “10”, number)

AddString(“value”, var, newvar) – same as before but for strings

Wait(“value”) – the wait function for value ms

CreateName(“origName”, “counter”, newname) – takes the origName of the file which will be saved and put the number from counter to the end ,can be used in acq. In DO cycle

StartFrameAcq(devID, numberOfFrames, timeOfAcq, flags, filename, filterChainName) – starts the frame acq. with set parameters

- devID – ID of the device in Pixelman
- numberOfFrames – how many frames, should be collected
- timeOfAcq - length of exposition for each frame
- flags: FSAVE_BINARY 0x0001 // save in binary format
  FSAVE_ASCII 0x0002 // save in ASCII format
  FSAVE_APPEND 0x0004 // append data file to existing file if exists
  FSAVE_I16 0x0010 // save as 16bit integer
  FSAVE_U32 0x0020 // save as unsigned 32bit integer
  FSAVE_DOUBLE 0x0040 // save as double
  FSAVE_NODESCFILE 0x0100 // do not save description file
  FSAVE_SPARSEXY 0x1000 // save only nonzero position in [x y count] format
  FSAVE_SPARSEX 0x2000 // save only nonzero position in [x count] format
  FSAVE_NOFILE 0x8000 // frame will not be saved :)

flags FSAVE_BINARY/FSAVE_ASCII and flags FSAVE_I16/FSAVE_U32/FSAVE_DOUBLE are mutually exclusive (user can set the flag in decimal or hexadecimal form)
if FSAVE_SPARSEX and FSAVE_SPARSEXY is not specified full matrix is saved

- filename – name of the file to which data will be saved
- filterChainName – name of the filter chain which will be used, if this filter chain is not found, raw data are saved

StartInegralAcq(devID, numberOfFrames, timeOfAcq, flags, filename, filterChainName) – starts the integral acq. with set parameters

Example of data collection with moving detector after each acqusition:

CreateVar(“D:\data\example.txt”, origname) //creates variable with name origname

CreateVar(0,numberofacq) //creates variable with name neumberofacq

DO(10) // makes 10 times DO cycle //beginning of DO cycle

```
CreateName(origname, numberofacq, newname)      //creates name for acq., in order to not rewrite same file

StartAcq(3, 10, 0.5, 34, newname, flatfield)    //starts the acq, device 3, 10 frames, each 0.5 s, flag means ASCII ,U32 frame, filename, name of filter chain

AddNumber(1,numberofacq,numberofacq)            //counter of acq

Move Motor(10, 1, ”Rotation”)                   //external function to move motors

ENDDO()                                         //end of DO cycles

ENDSCRIPT()                                     // end of script
```