## Command Reference: RunProgram()

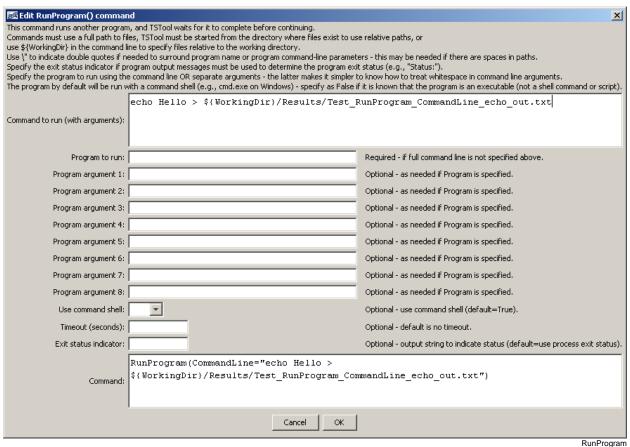
Run an external program

General Command
Version 3.08.02. 2010-01-07

This command is under development. Preliminary development has occurred in the TSTool software. The RunProgram() command runs an external program, given the full command line or individual command line parts, and waits until the program is finished before processing additional commands. The command will indicate a failure if the exit status from the program being run is non-zero. It is therefore possible to call an external program that reads and/or writes recognized time series formats to perform processing that StateDMI cannot. It is also useful to use StateDMI's testing features to

implement quality control checks for other software tools.

StateDMI internally maintains a working directory that is used to convert relative paths to absolute paths to locate files. The working directory is by default the location of the last command file that was opened. The external program may assume that the working directory is the location from which StateDMI software was started (or the installation location if started from a menu). Therefore, it may be necessary to run StateDMI in batch mode from the directory where the external software's data files exist, use absolute paths to files, or use the \${WorkingDir}} property in the command line. Use \" in the command line or arguments to surround whitespace. Some operating systems may have limitations on command line length. The following dialog is used to edit the command and illustrates command syntax.



RunProgram() Command Editor when Specifying Command Line

RunProgram

The command syntax is as follows:

RunProgram(Parameter=Value...)

## **Command Parameters**

Parameter	Description	Default
CommandLine	The full program command line, with arguments.	Must be specified if the
	If the program executable is found in the PATH	Program parameter is
	environment variable, then only the program name	not specified.
	needs to be specified. Otherwise, specify an	
	absolute path to the program or run StateDMI	The Program
	from a command shell the same directory.	parameter will be used
		if both are specified.
	The \${WorkingDir} property can be used in	
	the command line to indicate the working	
	directory (command file location) when	
	specifying file names.	
	For Windows, it may be necessary to place a \"	
	at the start and end of the command line, if a full	
	command line is specified.	
Program	The name of the program to run. Program	Must be specified if the
	arguments are specified using the ProgramArg#	CommandLine
	parameter(s). See the CommandLine parameter	parameter is not
	for more information about parameter formatting	specified.
	and locating the executable.	
ProgramArg1,	Command like arguments used with Program. If	No arguments will be
ProgramArg2,	necessary, use \${WorkingDir} to specify the	used with Program.
etc.	working directory to locate files.	
UseCommandShell	If specified as False, the program will be run	True, using cmd.exe
	without using a command shell. A command shell	/C on Windows and
	is needed if the program is a script (batch file), a	/bin/sh -con
	shell command, or uses >,  , etc.	UNIX/Linux.
Timeout	The timeout in seconds – if the program has not	No timeout.
	yet returned, the process will be ended. Zero	
	indicates no timeout. This behavior varies and	
	is being enhanced.	
ExitStatus	By default, the program exit status is determined	Determine the exit
Indicator	from the process that is run. Normally 0 means	status from the process
	success and non-zero indicates an error.	exit value.
	However, the program may not exit with a non-	
	zero exit status when an error occurs. If the	
	program instead uses an output string like STOP	
	3 to indicate the status, use this parameter to	
	indicate the leading string, which is followed by	
	the exit status (e.g., STOP).	

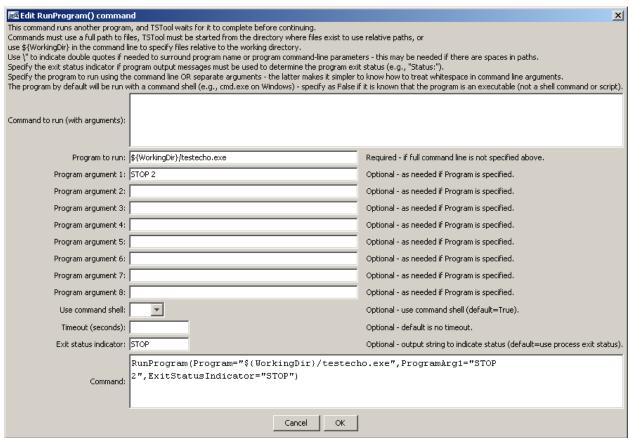
The following figure illustrates how a command would be entered using the program name and parts, and use the command shell to run. Note that the output redirection character ">" is entered as a program argument. The *echo* program on Windows is actually internal to the *cmd.exe* command shell and therefore must be run using the command shell (the default behavior).

This command runs another program, and TSTool waits for it to complete before continuing.  Commands must use a full path to files, TSTool must be started from the directory where files exist to use relative paths, or use \$\frac{4}{3}\text{WorkingDir}\$ in the command line to specify files relative to the working directory.  Use \" to indicate double quotes if needed to surround program name or program command-line parameters - this may be needed if there are spaces in paths.  Specify the exit status indicator if program output messages must be used to determine the program exit status (e.g., "Status:").  Specify the program to run using the command line OR separate arguments - the latter makes it simpler to know how to treat whitespace in command line arguments.  The program by default will be run with a command shell (e.g., cmd.exe on Windows) - specify as False if it is known that the program is an executable (not a shell command or script).			
Command to run (with arguments):			
Program to run:	echo	Required - if full command line is not specified above.	
Program argument 1:	Hello	Optional - as needed if Program is specified.	
Program argument 2:	>	Optional - as needed if Program is specified.	
Program argument 3:	\${WorkingDir}/Results/Test_RunProgram_Program_echo_out.txt	Optional - as needed if Program is specified.	
Program argument 4:		Optional - as needed if Program is specified.	
Program argument 5:		Optional - as needed if Program is specified.	
Program argument 6:		Optional - as needed if Program is specified.	
Program argument 7:		Optional - as needed if Program is specified.	
Program argument 8:		Optional - as needed if Program is specified.	
Use command shell:	<u></u>	Optional - use command shell (default=True).	
Timeout (seconds):		Optional - default is no timeout.	
Exit status indicator:		Optional - output string to indicate status (default=use process exit status).	
Command:	RunProgram(Program=echo,ProgramArg1=Hello,ProgramArg2=>,ProgramArg3=\${WorkingDir}/Results/Test_RunProgram_Program_echo_out.txt)		
	Cancel OK		
RunProgram_Program			

RunProgram() Command Editor when Specifying Program and Arguments

The following figure illustrates how a command can be run without a command shell and using the program output to determine the exit status. The *testecho.exe* program is a compiled executable and can therefore be run without a command shell. Because the standard output is being evaluated for the exit value, the output cannot be redirected to a file with > (this would result in no output being available to StateDMI to evaluate), and > is only recognized if running with a command shell in any case.

The following approach is suitable, for example, when running a compiled model or data analysis tool. However, if the tool is run using a script or batch file, then a command shell must be used.



RunProgram\_Program\_ExitStatusIndicator

RunProgram() Command Editor when Specifying Program, Arguments, and Exit Status Indicator