Site Server Command Line Functionality

Site Server contains limited command line functionality to control execution of an existing project.

The command line functionality is provided by TCRUNNER.EXE or TCCONSOLE.EXE.
Both programs are contained in the installation directory for Site Server.

Either executable can be accessed using a command line call from the Run line in Windows, or from a program execute function from most products. Note that command lines are limited by the operating system to 250 characters. The character limitation may pose a problem, as you may not be able to type in all of the information that you need. To resolve this issue, you can use alternative mechanisms that make it possible to pass in larger amounts of command data, as described in this document. Whichever mechanism you choose, the basic forms of the commands are common.

The fundamental difference between TCConsole and TCRunner is that TCRunner is compiled as a "Windows-based" application, and TCConsole is compiled as a "console" application.

The most significant characteristics are:

TCRunner:

- Will return control to the calling process immediately.
- Will then proceed to do the work.
- Will not display any visible window.

TCConsole:

- Will do the work synchronously.
- Will not return to the calling process until the work is done.
- Will display (if required) a "command shell" window.

TCConsole is more suited to a batch environment where strict synchronous processing is required and where the command shell environment is already present.

Commands can be entered directly on the command line, redirected via `stdin`, or contained in a steering file.

For example, you could say:

```
TCRUNNER.EXE Project="Technical Docs.TCP"
```

or

```
TCRUNNER.EXE < params.txt
```

or

```
TCRUNNER.EXE CommandFile="params.txt"
```

Note: You must run **PswdCfg.exe** to set the credentials for the user running the project in the registry. (For more information, see "Registering a username and password for Site Server" in Chapter 3 of the Installation Guide, in the Documentation folder.)

Functionality

Using the command line interface, the following attributes can be controlled:

Project File
Task Log File
Task Logging Level
Task Logging Reset

These settings can be controlled via command line parameters, which can be added cumulatively to the command line. These commands are grouped with the value that is being passed. So, for the Project File, the command line would include:

```
TCRUNNER.EXE Project="Technical Docs.TCP"
```

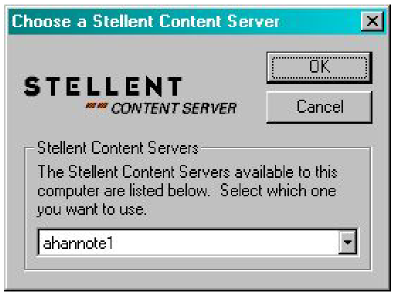
The keywords for the commands are:

Project=	filename or "filename" Specifies the file path for a file system based project. Use quotes if the path or filename contains spaces. Default is "".
LogFile=	filename or "filename" Use quotes if the path or filename contains spaces. Default is "".
LogLevel=	A number to specify the log level to set – from 0 to 5. Default is 4
LogReset=	yes or no Setting to control resetting of the log file. Default is yes.
DataDir= or DataBase=	pathname or "pathname" The path to the incremental build database directory Default is "".
UpdateProject=	yes or no Setting to control automatic updating of old project files. Default is no.
CommandFile=	filename or "filename" a file containing commands. Default is "".
ShowParams=	yes or no Primarily for diagnostic purposes, writes the accumulated input parameter values to `stdout` *after* the results report. Default is no.

AllowFileOverwrites=	yes or no Controls behavior w.r.t. filename collision avoidance at the translate stage. By default, the engine will avoid overwriting files that don’t belong to the project. This option allows you change that behavior. Default is no.
Steps=	a bit significant decimal number This parameter controls which project "steps" to execute. 1 = update 2 = translate 4 = stage 8 = publish Default is 15 (update + translate + stage + publish) If there are no stage or publish settings at all, then no error message will be generated if you request to stage or publish. If stage or publish settings are specified, you may get an error message if some previous step (e.g. translate) is not up-to-date. **Note**: If you have not set staging options for your project, the default setting of 15 should not be used. Instead, the parameter for Steps is 11 (update + translate + publish).

For a repository such as Content Server, replace the "Project=" parameter with "ClassID=" and "DocID="

ClassID=	conveys identity information about the repository type. This is either a built-in keyword or the CLSID of the RepoProject object. For Content Server, you can simply say ClassID=Stellent
DocID=	contains a repository-specific identifier of the project file. For Content Server, the form is: DocID=Intra.doc!^server^dDocName^^AutoRev Where ‘server’ is the name of the content server connection and dDocName is the content ID of the publisher project file. AutoRev enables automatic revisioning of the project file. Omit to disable.
Example:	Here, the content server connection is ‘ahannote1’:



Choose a Stellent Content Server

Stellent Content Servers

The Stellent Content Servers available to this computer are listed below. Select which one you want to use.

ahannote1

For TCConsole.exe ClassID=Stellent
DocID=Intra.doc!^ahannote1^SCPTest^^AutoRev

Invalid parameters will leave the settings unchanged. The commands must be separated by spaces.

Examples:

```
TCRUNNER.EXE Project=\\Operations\HR\Human Resources.tcp
LogFile=\\UserMachine\Shared\LogFile.txt LogReset=no

"C:\Program Files\Stellent Content Publisher\TCRUNNER.EXE" Project="C:\Marketing.TCP"
LogFile="C:\ MktgLog.txt"
Logreset=no
```

When you use a text file to store the commands, they can be separated by a space or a carriage return.

Example:

1. Create a text file and name it c:\TCConsole.txt
   ```
   LogFile=c:\TCConsoleLog.txt
   LogLevel=5
   ClassID=Stellent
   DocID=Intra.doc!^server^dDocName^^AutoRev
   ```

2. Start the Command Prompt and run TC Console with the CommandFile= parameter like this:
   ```
   c:
   cd \program files\stellent content publisher
   tcconsole.exe CommandFile=c:\TCConsole.txt
   ```

Setting Hierarchy

The settings have the following hierarchy, with the later items taking precedence:

1. Defaults built into the product.
2. Defaults set in the registry
3. Command Line Parameters
4. Command File Parameters
5. Parameters directed via `stdin` (only in TCRunner)

The minimal entry would be a Project File, with everything else using the default settings.

Built in defaults

The following settings are the built in default settings:
LogFile = ""
LogLevel = 4
LogReset=yes
Steps=15 (update + translate + stage + publish)
ShowParams=no
AllowFileOverwrites=no
UpdateProject=no

Registry Settings

The following registry settings allow you to specify the defaults for running the command line interface.

HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\Project	(string)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\ClassID	(string)

HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\DocID	(string)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\LogFile	(string)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\LogLevel	(dword)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\LogReset	(string)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\DataBase	(string)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\UpdateProject	(string)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\ShowParams	(string)
HKEY_LOCAL_MACHINE\Software\Stellent\Content Publisher\Runner\6.0\CommandFile	(string)

Return Values

Result information gets written to `stdout`. This will be a single line of text of the form:

```
result="Success" hr=0 infiles=42 outfiles=99 stagefiles=99 pubfiles=99 errors=0 warnings=0
```

If there is an error, the “result” value will display a very short error message that describes the stage in the processing where the error occurred. If there is an error, the “hr” value will be an 8-digit hexadecimal value that communicates the reason for failure.

Note that the returned hex representation does not include a “0x” prefix.
So you will see something like hr=80041016, not hr = 0x80041016
You should interpret the value as a 32 bit HRESULT.

Anticipated error values are:

E_IA_FILELOCKFAILURE	80041019	(Project Open)
E_IA_FILEDOESNOTEXIST	80041016	(Project Open)
E_IA_TRANSLATIONERRORS	80041023	(Project Translate)
E_IA_OUTOFMEMORY	80041002	(any)
E_IA_FAILED	80041000	(any)

Note that TCRunner is a Windows application, not a Console application, so when you run TCRunner from the MSDOS command line, the DOS prompt returns immediately and the `stdout` return is not displayed there.

To capture the return value, redirect the output to a text file. For example:

```
TCRunner < Params.txt  > Results.txt
```

If you use the “more” command, then you can delay the return of the DOS prompt. For example:

```
TCRunner Project=c:\dir1\Project1.tcp | more
```

If processing was attempted normally, the program will return 0 even if errors/warnings occur during that processing. In that case, detailed result information will be available from `stdout`. In exception cases, the program will return –1.