# Using T3Q

T3Q is a command line tool which can be called by simply typing "t3q" in Windows or on Unix systems. The Windows command-line can be accessed by starting the program "cmd". For Unix systems, we only support bash environments.

## Configuration

T3Q uses an XML-based configuration format. Starting with version v0.4.2, the location of the configuration file must be supplied as a command line option during execution (the location was previously based in the default user's data location - depending on the system, a file `t3q.xml` was stored in `%APPDATA%\T3Q` (Windows) or in `~/.t3q/` (Unix)). The location of the `%APPDATA%` directory depended on the username, the Windows version and possibly the localization. For example, for a user 'foobar' on a German Windows XP machine, the resulting configuration path would be `C:\Documents and Settings\foobar\Application Data\T3Q`. In version v0.4.2 and onwards, the location of the configuration file has to be supplied every time T3Q is called. Additionally, starting with version v1.0.1, if there is no existing configuration file in the selected location, the tool will prompt the user to use the appropriate option to generate a new configuration with the default settings. Thus, the tool has to be started with the appropriate parameter to generate a new configuration prior to actual usage. This way, it will be possible to modify the configuration file to accommodate the particular needs of the user prior to performing any analysis. By location here the path, filename, and extension of the configuration file are meant, which implies that any filename and any extension can be used. It is probably best to retain certain norms in naming the configuration files, at least in preserving the file extensions (`.xml`) to avoid confusion.

Please note that after an update, during the development stage, the configuration files will often be extended. Thus, it may be necessary to generate a new configuration file and transfer the custom settings from the old configuration file. The configuration file has two sections:

- A configuration profiles section
- A main section

The **ConfigurationProfiles** section contains a list of **QualityCheckProfile** elements. Its elements contain the tags **profilename** as well as check* elements and other specific elements that are needed for the configuration of some of the quality checks. In the default `t3q.xml` that is created, there is a profile called *defaultProfile* that initializes all profile configuration values to the default values and enables all checks. This is also the configuration of an implicitly given profile called *all*. The *all* profile is not part of the XML configuration, but it exists and is known to the tool.

Since v0.3.1, configuration profiles also have a version (**profileVersion** element), that regulates the profile compatibility. If a profile from an older version is used, T3Q will throw and error and recommend profile upgrade or the selection of another profile. Also, further robustness checks have been introduced to provide hints if the configuration profile is otherwise incompatible or corrupted. In case of a problem that can be localized, a corresponding error message is provided suggesting the location of the problem.

Also, there are several generic options, that affect the overall behavior of the T3Q tool. The known extensions for files that are to be processed can be specified by means of a regular expression in the **resourceExtensionsRegExp** option. By default, *ttcn*,*ttcn3* and *3mp* file extensions are recognized. Since v1.0.1, there is also support for supplying input by means of what will be referred to as *project* files. These files specify a list of input files and directories (including wildcards). The **projectExtension** option regulates what is to be considered a project file. Note that this is not a regular expression option, therefore only a single value is currently supported. Since v1.0.3, the **ignoredResourceRegExp** option enables the specification of regular expressions for resources within the set of input files that will be ignored during the processing steps. Note that all resources in the input set will be parsed and pre-processed so that definitions within ignored resources will be still be correctly resolved during the analysis phase. The ignored resources will be skipped during the analysis and formatting steps (annotated with **(Skipped)** in the output during analysis and formatting). The default setting for ignored resources is **.*IGNORED.***, meaning that resources which contain **IGNORED** in their full path will be skipped. There is an option to switch recursive processing on and off (**settingRecursiveProcessing**). There is an option to switch aborting on (parsing) errors on and off (**settingAbortOnError**). Both of these options are turned on by default. Turning them off may result in unreliable output. Files that contain syntax errors will be analyzed only up to the point of the first syntax error occurrence. As a result, imported definitions may not be resolved correctly, which in turn may affect some of the quality checks. The same situation may occur if not all the resources in a test suite are processed.

Additionally, there are options to regulate the output. As as of version 0.4.1, these are grouped under **loggingConfiguration**. **statShowFullPath** can be used to enable / disable the display of the path of a file in the output, when a quality check is violated. Additionally, if desired, **statShowFilename** can be set to **false** to disable outputting the filename with each output message. Further options include **showMessageClass** - to display the message class (or also quality check class, e.g. naming convention, code style, etc), **showDetails** - to display further details about the output message, which in the current state generally shows the internal name of the quality check that generated the message (the internal name is generally corresponding to the configuration entry for easy traceability), and the reference requirement ID (although the latter may be omitted in subsequent releases). There is also an option **logOutputPrefix** to add a custom prefix to every log message (at the beginning of the line, before the path / filename). By default, this option adds three empty spaces before each logging message in the output.

There are two further settings related to the output - one that shows a brief statistics summary with basically counters of the occurrences of each message class (**statShowSummary**) and one that enables / disables output statistics about the length of the files being analyzed (in lines of code) - **statShowLOC**.

In the *main section*, there is currently one single configuration element **defaultConfigurationProfile**. It points to the implicit *all* profile by default. The **defaultConfigurationProfile** is the profile that will be used if no specific profile is provided as command-line parameter. If the specified **defaultConfigurationProfile** does not exist in the configuration, T3Q will fall back on the implicit *all* profile.

## Command-Line Usage

T3Q is used as follows:

```
t3q [options] ( path | filename )+
```

This means that apart from any required options (see below), there always has to be at very least one parameter that needs to be specified. This parameter is the input parameter, which can be either a path that contains the TTCN-3 files that should be analyzed, the name of an individual file, or any combination of these (a mixed list), including wildcards.

- If a path provided as input (and recursive processing is enabled in the configuration, which is the default setting), T3Q will recursively parse and analyze all files in this directory that match the provided file extensions (which are specified in the configuration file). For the current path a simple '.' is sufficient. If no files match these extensions, T3Q will output a corresponding message and quit.
- If an individual file is provided as input, then only that file will be processed.
- If the path or filename contains spaces, you need to put the path into quotation marks or use the auto-completion feature of the environment (provided there is one), which should take care of escaping spaces or enclosing the complete path in quotation marks.
- If a list of individual files and/or paths are provided as input, these will be combined and processed together.
- If wildcards are used, these will be expanded by the command-line environment into a list and subsequently analyzed as such.

The following options can be provided currently and can be specified in any order:

```
--generate-config <FILE NAME>   Generate a new default configuration
                                file at the specified location
--config <FILE NAME>            Configuration file location
--profile <PROFILE NAME>        Configuration profile

--format                        Apply code formatting
--verbosity <LOG LEVEL>         Verbosity level (currently supports ERROR,
                                WARNING and INFORMATION values)

--local-dependencies            Generate local dependencies

--output-path <PATH>            Destination path for the output (if
                                applicable, otherwise ignored).
                                Overrides the profile setting.
```

```
    --help                              Show this usage information screen
```

- `--help` will provide brief usage information listing the expected syntax and the available options. T3Q will stop and analyze no files when the help screen is called.
- The `--generate-config` option (new as of v1.0.1) allows the generation of new default configuration files at the location specified. T3Q will then quit.
- The `--config` option (new as of v0.4.2) is mandatory and has to be specified every time T3Q is run (except when `--help` or `--generate-config` are used). It specifies the location of the configuration file. Starting with v1.0.1, if no configuration file is found at the specified location, the user will be prompted to use the appropriate option (`--generate-config`) to produce a new default configuration. To use the default location from previous versions one will have to specify it manually, e.g.

  ```
  t3q --config ~/.t3q/t3q.xml
  ```

  on a UNIX based system or

  ```
  t3q --config %APPDATA%\T3Q}\t3q.xml
  ```

  on a Windows based system. The `--generate-config` and the `--config` options are mutually exclusive, with `--generate-config` having precedence, meaning that if both are specified, T3Q will still only generate the new default configuration and quit. Please not that if the location of the configuration file contains spaces, it has to be either enclosed in quotation marks or the auto-completion feature of the environment has to be used to take care of escaping the spaces.

- The `--profile` option overrides the **defaultConfigurationProfile** in the XML configuration. This means that you can specify multiple profiles in the XML configuration and run T3Q using another existing profile without the need to change the XML configuration. If the profile specified on the command-line does not exist, T3Q will automatically fall back to the default profile provided in the main section of XML configuration. In turn, if this default profile does not exist as well, T3Q will fall back to the implicit *all* profile. If the configuration profile name contains any empty spaces, they need to be escaped or the profile name needs to be enclosed in quotation marks, depending on the environment.

- The `--format` option (newly introduced in v0.4.2) enables the code formatting feature (see below), which had to be enabled by a configuration entry in the configuration profile in previous versions. That configuration entry is now superseded by the command line option, although the output path where the formatted output is to be placed still needs to be specified in the configuration profile (this may also be superseded by a command line argument for convenience in future releases).

- The `--verbosity` option (also newly introduced in v0.4.2) regulates the verbosity level of the output messages based on their type. The possible values are *ERROR*, *WARNING* and *INFORMATION* (in an ascending inclusive order, meaning that when *INFORMATION* is selected, the output will include both *ERROR* and *WARNING* verbosity level messages as well). *INFORMATION* is the default setting. More information about the message types is available in the next section.

- The `--local-dependencies` option (newly introduced in v2.0.0b23) enables the generation of local-dependencies. Currently, when the feature is activated, the other checks are deactivated. The location of the generated output can be specified with the **dependencyOutputPath** in the configuration profile (*DOCUMENTATION* by default). The output path can also be overridden by the command line argument for the `--output-path` option.

- The `--output-path` option (newly introduced in v1.0.1) makes it possible to supply the output path at the command line interface (overriding the profile setting). If affects only the code formatting feature and if it is not used, this optionr is simply ignored. It is added purely for convenience in case the same profile has to be used on different projects with different destination paths.

The native binary executable for Windows supplied with v1.0.1 is discarded again as of v1.0.2, due to the fact that it does not grant the desired advantages. Thus, the basic usage is reverted to the batch scripts. Starting with v1.0.2, the batch scripts include a "hidden" `--echo` option, which simply outputs the deployment specific call to the Java virtual machine, **without** any command line arguments, meaning that it makes no sense to provide any further command line parameters besides the `--echo` option. The sole purpose of this option is to output the full command line necessary for starting the tool, which may be necessary for embedding into third party tools. This is why this option is considered hidden (it also does not show in the standard help screen). It should not affect the general usage of the tool. If, for whatever reason, other command line arguments are supplied together with the `--echo` option, then the `--echo` option needs to be the first command line argument.

## Output Format

The output format is configurable to a certain degree. Individual files being analyzed are listed, and if any of the quality checks are violated an output message is provided. The message format is:

```
[Prefix][[Path]FileName: ]LineNumber: MessageType: [MessageClass: ]Message[ (MessageDetails)]
```

where whether the path is displayed can be switched on or off in the T3Q configuration using the **statShowFullPath** boolean switch. If **statShowFullPath** is turned off, only the filename is displayed, otherwise the path is displayed as supplied to T3Q, including the subdirectories. The filename provides the name of the file where the violation occurred. It is also optional. Whether the filename is displayed or not is determined by the **statShowFilename** boolean switch. It can be handy to reduce output clutter because all the output messages are grouped under the files they occurred in anyway. The line number(s) indicates the line or scope of lines (separated by "-") where the violation occurred. The message type indicates the type of message. There are currently three types of messages *WARNING* - if a quality check is violated, *INFORMATION* in case a processing problem has occurred, and *ERROR* in case a serious error (such as a parsing error) has occurred (note that currently parsing errors may have a different format, they are in the process of being unified in format with the other message types). After the message type, if enabled (via the **showMessageClass** switch), the message class will show information about the type of quality check violated. There have been seven message classes initially defined:

- *General* - for general messages
- *Naming Conventions* - for naming conventions-related violations and messages
- *Code Documentation* - for documentation-related violations and messages
- *Log Statements* - for violations and messages related to log statements
- *Structure of Data* - for violations and messages related to the structure of data
- *Code Style* - for violations and messages related to coding style
- *Test Suite Modularization* - for violations and messages related to modularization

In addition there is also the *Universal* class, for messages that do not fall into any of the other categories. These message classes are also used for the brief statistics summary at the end of the processing (if enabled). These classes can be used for further filtering of the output. The improved logging interface will allow the generation of structured file format log (such as XML), which may subsequently be used with other tools for automated processing, filtering, and sorting based on any of the output fields.

The message contains the details about the occurrence. Finally, the optional message details contain further details about the occurrence, which in the general case amount to a reference to the internal name of the quality constraint being violated, which is identical to the configuration tag.

Examlpes:

with path:

```
ETSI 3GPP/IWD_09wk04/7_1/MAC.ttcn: 2107-2117: WARNING: Code Style: A nested
alt statement is used! (6.3, checkNoNestedAltStatements)
```

without path, class, and details:

```
MAC.ttcn: 2107-2117: WARNING: A nested alt statement is used!
```

## Performance and Memory Usage

Large TTCN-3 test suites tend to take quite a while to process (both for parsing and for analysis). Therefore, it is generally a good idea to set larger memory limits (as far as the system allows) in order to improve processing time and avoid possible memory problems. The default setting is to set the **upper** memory limit to 512MB, which by today's standards is rather conservative, however, it should be sufficient for smaller to medium-sized TTCN-3 test suites. The optimal memory limits are not easy to determine, and ways to automatically calculate and set these depending on the available system resources, the size of the input TTCN-3 test suite, and the configuration in use are currently being investigated. In the meantime, should processing take too long or memory errors occur, it is advisable to set a higher upper memory limit, depending on the available system resources. This can be done by manually editing the parameters in the start scripts (`t3q.bat` and `t3q` for Windows and Unix respectively). These files should be edited very carefully, as mistakes may prevent T3Q from starting. Under Windows, in `t3q.bat`, set the `-Xmx` parameter in following line:

```
set JAVA_CMD=%JAVA% -Xmx512m -Xss128m -cp "%CLASSPATH%" org.etsi.t3q.T3Q
```

to the desired upper limit (e.g. to `-Xmx1024m` for a 1GB upper memory limit).

Under Unix, the `t3q` file can be changed in a similar fashion by setting the `-Xmx` parameter in the

```
JAVA_CMD="$JAVA -Xmx512m -Xss128m -cp $CLASSPATH org.etsi.t3q.T3Q"
```

line to the desired upper limit.

In general, the optimal setting depends on the size of the TTCN-3 test suite and to a degree on the selected quality checks.

As of v1.0.3, a tool to guesstimate the optimal memory settings is included with T3Q. This tool is launched prior to the actual tool execution and attempts to detect the maximum memory settings with which T3Q can be started at that particular moment, aiming to both reduce processing time and avoid potential out of memory errors. It should be noted that this tool can be considered in beta status, as in some edge cases it may cause T3Q to crash or to fail at start. Such cases should be reported so that the memory detection tool can be further adjusted to avoid such issues in the future. It is still possible to select preferred memory settings manually by adjusting the start-up scripts as described above, where the particular line has been changed to:

```
set JAVA_CMD=%JAVA% -Xmx%HEAP%m -Xss128m -cp "%CLASSPATH%" org.etsi.t3q.T3Q
```

under Windows and to

```
JAVA_CMD="$JAVA -Xmx`$MT_CMD`m -Xss128m -cp $CLASSPATH org.etsi.t3q.T3Q"
```

under Unix, and the `%HEAP%` and `` `$MT_CMD` `` parts of the line should be substituted with the desired settings.