Main Output

Before getting into the details of the main output, first the location of the output and its calculation need to be described.

Output Paths

T3D takes the base output path from the **outputDirectory** element in the XML configuration. In this directory, depending on the input a sub-directory will be created for each successful T3D execution. The main output includes the following entities:

- XML files containing the information extracted from the input TTCN-3 file(s)
- An `html` sub-directory containing the generated HTML, CSS, and JavaScript files

The **real** output directory is calculated from the input. If the input is a directory then the name of the last directory in the input path (the directory that is actually to be documented) is selected as destination the name of the sub-directory: For example, given input path `a/b/c` (relative) and base output path `c:\Temp\T3D\`, the generated documentation will be placed in `c:\Temp\T3D\c\`. If the input is a filename, then a sub-directory will be created with the name of that file in the base output directory: For example, given input filename `templates.ttcn`, the output will be generated in `c:\Temp\T3D\templates\`. Any subdirectories leading to the input filename will be disregarded, e.g. generated documentation for inputs such as `Libraries\templates.ttcn` or `c:\TTCN\Libraries\templates.ttcn` will still be placed in the `c:\Temp\T3D\templates\` destination directory. Particular cases, such as documenting the current directory, e.g. "`.`" or "`./`", will result in documentation being generated in a sub-directory in the base output directory with the name of the current directory, that is calling `t3d --config myConfig.xml --profile myProfile .` while in `c:\TTCN\Libraries\` will generate the documentation in `c:\Temp\T3D\Libraries\Libraries\` (assuming the same base output path is used as in the above examples). If multiple inputs are provided (e.g. multiple files, directories, or by means of wildcards), the name of the first entry in the list is taken for the destination sub-path.

XML Output

The XML output consists of (currently) three XML files storing the relevant data extracted from the TTCN-3 sources in a structured intermediate format. The currently generated XML files are:

- `project.xml` - This file contains the most comprehensive information and represents an abstracted version of the source TTCN-3 files, including module structure, the individual elements, their bodies (if configured so), and T3Doc comments for every module, group, and element of the processed TTCN-3 files. It serves as a basis for the main and the module parameter / test case views (for details on the separate views, see Section Views below).

- `imports.xml` - This file contains information about the imports and dependency relations of the processed TTCN-3 modules. It serves as a basis for the import / module dependency view.
- `dependencies.xml` - This file can be thought of as a blend between an abstracted version of the main `project.xml` file and a low-level version of the `imports.xml` file, featuring a compact representation of the low-level dependencies at the module definition (element) level - it contains all the module definitions and all the known and unknown elements referenced directly within each module definition, where unknown elements are currently prefixed with `unresolvedReference---`. There is no separate view associated to this file, since it is basically a compact representation of the main view and its main intent is the use with third-party tools to perform custom tasks, such as slicing or markup of definitions related to a particular module definition (e.g. approved / locked definitions, etc.).

These XML files serve as a basis for the HTML generation. They can also be utilized by custom third-party tools to produce different output depending on the particular needs. More information on this is available in the T3D technical documentation.

HTML

The main output consists of interlinked HTML files. There are three different views serving different purposes.

There is the **Main View**, which serves as a browseable representation of the source code, with a separate page for every every module and every construct. The output currently includes and shows the source code for every module and construct and all references to known constructs are linked to the corresponding page for these construct.

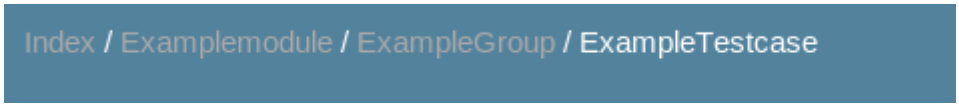
Then there is the **Module Parameter/Testcase? View** that shows a summary of the module parameters related to a particular test case and vice versa. Optionally it can also display the path to each reference, which is useful for indirect references (i.e. a module parameter is referenced in a function or an altstep called within another function, which in turn is called within the test case).

Finally, there is the **Import View** which should serve as an overview of the dependency relations between modules. This view is currently based on the import statements only and does not take into account whether imported definitions are actually used in the importing module. Apart from direct dependencies, it also lists distinct indirect dependencies (i.e. when module1 imports from module2 which in turn imports from module3, when module1 is selected, module3 will be listed as an indirect dependency, since for consistent use of module1 both module2 and module3 will be necessary).

Layout

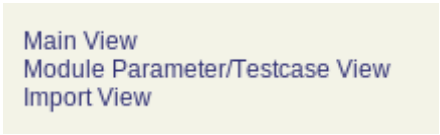
All three views are embedded into a more or less identical layout. The current default layout consists of several sections:

- A header section, which currently includes a *path navigation*. The path navigation currently features the view index and the structural path to the selected construct, including the selected / containing module, any (nested) groups the construct may be in and the construct itself. All the elements up to the selected construct are references to the respective entities:



T3D Path Navigation

- A left hand side navigation bar consisting of a view selector, presentation options, a module index and a construct index, where:
  - The *view selector* allows the user to switch between different views on certain constructs, e.g. switch between the the main view, the module parameter view, and the import view on the currently selected module, or between the main view and the module parameter view on the currently selected test case. If a certain view is not available for a particular construct, then upon selecting that view, the module index for that view will be displayed.

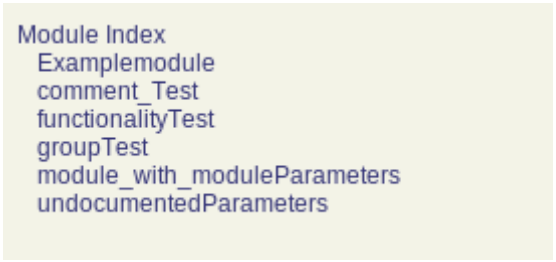


T3D View Selector

  - The *presentation options* control whether for example the bodies of the module definitions are displayed or not in the main view, or whether the contents of the import statements are displayed or not in the import view.

T3D Presentation Options

  - The *module index* allows the user to display the module documentation in all different views, for all the documented modules.



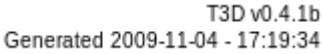
T3D Module Index

  - The *construct index* lists all the construct categories in the main view, and only test cases, module parameters, and groups in the module parameter view. The contents in each category depend on the currently selected module or group. If no module or group is selected (i.e. the module index of a view is selected), then all the documented constructs in all modules are listed. If a module or a group is selected, then only the constructs defined within that module or group are listed. To make the presentation more compact and manageable, the construct index categories are all collapsed. By clicking on a category, the list of constructs of that category is expanded below the category in a

familiar expand-collapse fashion. In the import view, this area is used to display clarifications on the color scheme used to mark the module dependencies.
Both the module index and the construct lists under the separate categories, as well as elements in the other views are sorted alphabetically (currently sorting is broken though).

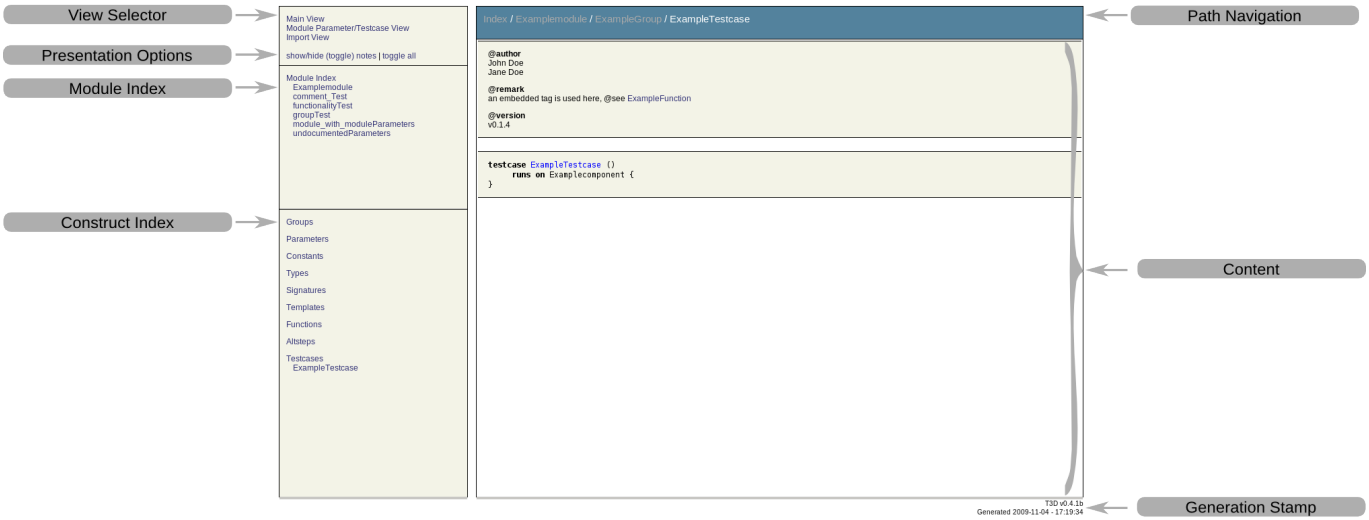
T3D Construct Index

- A *content section* on the right hand side, which contains the actual content of the selected view on the selected construct.
- A *generation stamp* in the lower-right corner indicating the tool name and version, and the generation date and time for the documentation page.



T3D Generation Stamp

The annotated screenshot below summarizes the complete layout for the main view of an example module.



T3D Annotated Main View
The main output has been tested so far under:

- Windows: Firefox 3.x, Opera 9.51, SeaMonkey 1.1, Safari 3.1.2, Internet Explorer 7
- Linux: Firefox 3.x, Opera 10.00, Konqueror 4.2.2

CSS and Layout Customization

This default layout can be further customized with the help of a Cascading StyleSheet (?CSS) file. The file is named `doc.css` and is placed in a `css` sub-directory under the `html` directory where the generated HTML files are located. The default CSS file is copied from the location specified in the configuration profile, which by default is `$T3D_HOME/css/doc.css`. Once changes are made to the CSS file in the output folder, they can be transfered back to the source CSS file or saved in a separate location and referenced in the documentation profile, so that the updated CSS file will be used the next time this profile is selected for the documentation of TTCN-3 test suites. Be aware that if the same output path is specified for the same input, all files will be overwritten, including any customized CSS files.

The default CSS file shall serve as a reference for customized CSS files. It is still in development and not yet complete. Comments within the file provide hints about the selectors. Further finer details can be deduced by examining the generated HTML code.

The CSS customizations are mostly for the presentation of the generated documentation. To influence the actual generated contents, the Extensible StyleSheet Language Transformations (?XSLT) file can be customized. It also needs to be customized shall new CSS selectors be introduced. More on XSLT in section *The Documentation Generation Process* below

JavaScript

The HTML output also makes use of a basic JavaScript (?JS) file (by default located in `T3D_HOME/js/doc.js` and copied to a `js` sub-folder in the output directory), which provides some basic functionalities necessary for the enhanced navigation and toggling of details. Same rules as with the CSS file apply, customizations need to be moved upstream if they are to be reused in future documentation generations. Also adaptation of the XSLT file may be necessary if major new JS functionalities are introduced. Additionally, a second JavaScript file called index.js is generated at runtime and then copied to the same folder as `doc.js`. This file is used to generate the *construct index* navigation in the main and module parameters views of the HTML documentation. Since this file is automatically generated with each documentation generation, it makes less sense to manipulate it upstream, instead it can be used for fine-tuning the final output. Due to the use of JavaScript, if support for it is not available or disabled in the user's browser (due to security or other concerns), the navigation will not work properly.