**Doxygen cheat sheet**

**Advanced Markup**

Text within comments can be marked up, which may effect how Doxygen parses the text in question, and/or how its appears in any output. Markup terms are preceded by ampersands @ (or backslashs \). Generally, tags can be used multiple times for the same entity (e.g. multiple authors, multiple warnings) and will be combined into a single paragraph.

Markup for global information:

**@file <file-name>**

Make comments for a source or header file called <file-name>. The file name may include part of the path if the name alone is not unique. If no file name is given it is assumed this documentation belongs to the current file. Remember that global functions, variables, typedefs etc. will only be included in the output if the file they are in is documented as well.

**@mainpage**

Include this information on the main page of the output, perhaps for important classes etc.

**@page [name] <title>**

Indicates that this comment block is not specific to a class, file or member. but should appear as a new page.

Markup for general information:

**@bug [description]**

Has the obvious meaning.

**@todo [description]**

Describe an item or issue that must be resolved. A seperate "To Do" list will be generated and will cross-reference this entity.

**@deprecated [description]**

Indicates this entity is deprecated and gives reasons, describes alternatives etc.

**@invariant [description]**

Describes an invariant property of an entity. Multiple invariants can be mentioned in a single or separate paragraphs.

**@warning [description]**

Describe a situation or detail that needs attention. As a suggested rule, this should be information the reader must heed, such as gotchas, limitations etc.. Similar to @attention.

**@note [text]**

Make a comment. As a suggested rule, this should only be information the reader can safely ignore, and of marginal interest like credits, inspirations and reasoning behind code. Similar to @remarks.

**@author [author-list]**

Has the obvious meaning. Multiple authors can be listed together or in separate statements.

**@see [name-list]**

Make references to other entities or URLs. Names joined by either :: are intrepreted as a member of a class. Overloaded methods can be selected by incuding a list of argument types.

Some markup for functions:

**@param <name> <description>**

For documenting function parameters.

**@return <description>**

For documenting function return values.

**@overload [function-declaration]**

Generates boilerplate documentation that refers to the earlier function.

**@pre { description of the precondition }**

Describe the precondition (requirements) of a function. Multiple preconditions can be listed together or in separate statements.

**@post [description]**

Describe the postcondition (required results) of a function. Multiple postcondition can be listed together or in seperate statements.

Some markup for appearance:

**@code ... @endcode**

For enclosing a literal block of code

**@internal**

Marks paragraph out as internal information that will not appear in the output.

**Groups**

Entities may be grouped into global *modules*, which appear together on a new page. Modules can include files, namespaces, classes etc. The group is created by using @defgroup <name> in a special comment block. An entity can be made part of the group by using @ingroup <name> inside its documentation block. Entities can also be grouped by enclosing them with @{ and @}. @addtogroup can be used instead of @defgroup to avoid accidentally defining groups multiple times.

Members of a file or class are automatically grouped on type and protection level, but can be made into *member groups* by use of //@{ and //@} or /\*@{\*/ and /\*@}\*/. Note these are different to the module grouping tags. Before the opening marker, a separate comment block may be placed that contains @name and other information.