Softwaredocumentation

1. What is software documentation?

Software documentation is the process of providing information about a software system to its users, developers, and other stakeholders. It can take many forms, such as user guides, API documentation, technical specifications, and more. The goal of software documentation is to make it easy for people to understand, use, and maintain the software.

1.1. There are several types of software documentation:

- **User documentation**: This type of documentation is intended for end-users of the software. It includes user manuals, help files, and tutorials. It explains how to install, configure, and use the software.
- **API documentation**: API (Application Programming Interface) documentation is intended for developers who will be using the software. It explains the different classes, functions, and data structures provided by the software and how to use them.
- **Technical documentation**: Technical documentation is intended for the software development team. It includes design documents, architecture diagrams, and code comments. It explains how the software works and how it was implemented.
- **Operations documentation**: Operations documentation is intended for system administrators, who are responsible for maintaining and operating the software. It includes instructions for deploying, monitoring, and troubleshooting the software.
- **Marketing documentation**: Marketing documentation is intended for the sales and marketing team. It includes brochures, data sheets, and case studies. It explains the features and benefits of the software to potential customers.

Overall, software documentation is an important aspect of the software development process, as it helps ensure that the software is well understood and can be easily maintained over time. It also helps to ensure that the software is easy for others to use, which can improve its overall effectiveness and adoption.

2. Doxygen:

Doxygen is a documentation generation tool for source code. It can be used to generate documentation in a variety of formats, such as HTML, LaTeX, and RTF, from source code written in a variety of programming languages, including C, C++, C#, Java, Python, and more. Doxygen uses special comments in the source code to extract documentation, and can also be configured to extract documentation from other sources, such as header files and documentation written in markup languages like Markdown or reStructuredText. Doxygen is widely used in software development to generate documentation from source code, and is particularly useful for generating documentation for large, complex codebases. It is also the standard for C, C++.

2.1. Key Benefits:

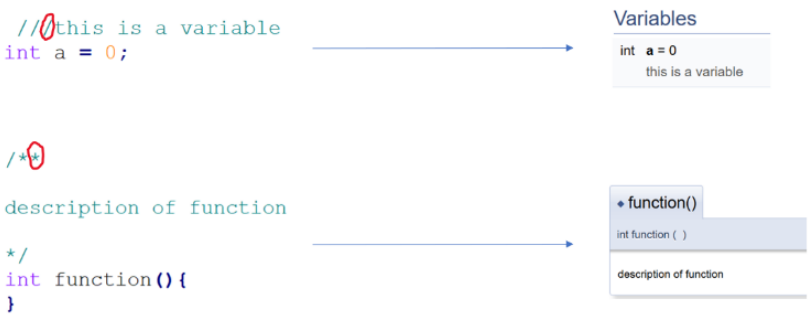
- Automatically update Documentation, if changes to the source code is made
- Makes it easy to document old code
- Makes it easy to document anything

2.2. How to configure

To run doxygen you need to install the version for your system. Then you need to modify the configuration file to fit your needs. These are common ways to configure Doxygen:

- INPUT: name of the project
- OUTPUT_DIRECTORY: directory to store the generated documentation
- FILE_PATTERNS: specifies list of file extensions that should be processed by doxygen
- EXTRACT_ALL: whether to extract documentation from all sources
- EXTRACT_PRIVATE: whether to extract documentation from private sources
- GENERAT_HTML: whether to generate HTML output

2.3. Syntax



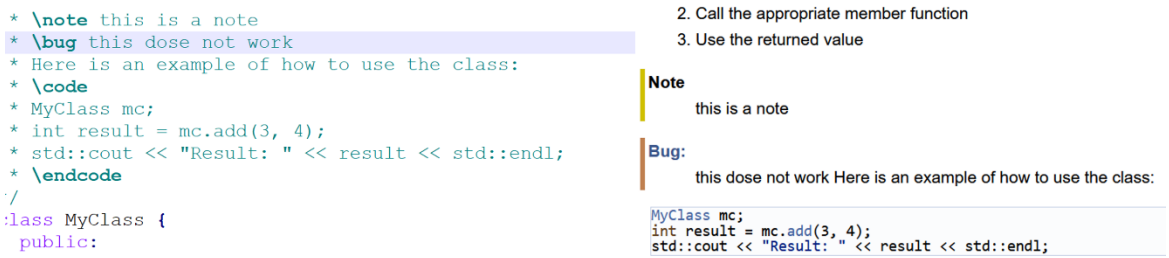
As shown in the figure above, it is very similar to c-comments syntax. Other syntax, similar to other languages is also possible. You should put the comment block before the entity you want to document, but you also can put it anywhere else in the code and use a structural command to identify it. For example: **\fn function** to identify the function in the figure.

2.4. Special Commands

Special commands in Doxygen are tags that provide additional information or formatting for the documentation. They are used within the specially formatted comment blocks to control the generation of the reference documentation. For example:

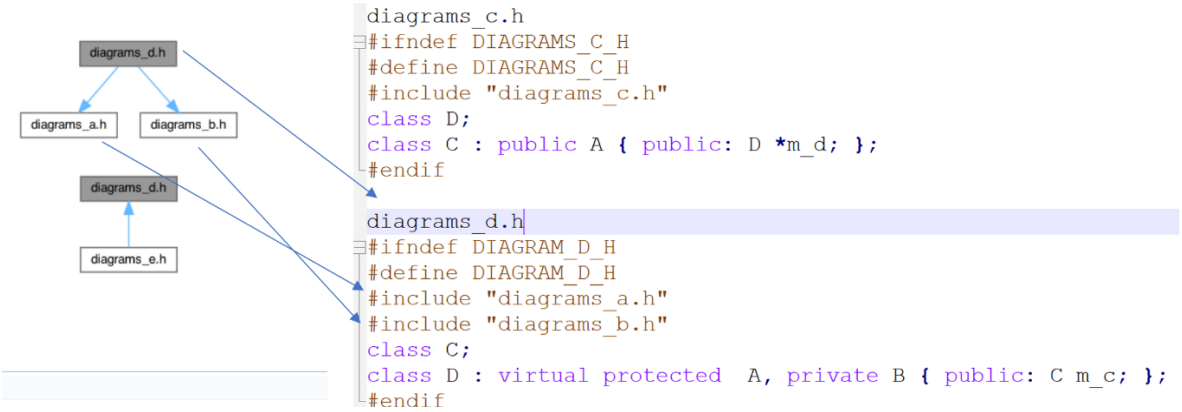
- **\author: Author of code**
- **\code: A section, specifically formatted for showing code**
- **\warning: A section for warnings**
- **\param: Shows the parameter of an entity**
- **\link**: link to an external document

Below is an example, which shows how the note, bug and code tags are used.

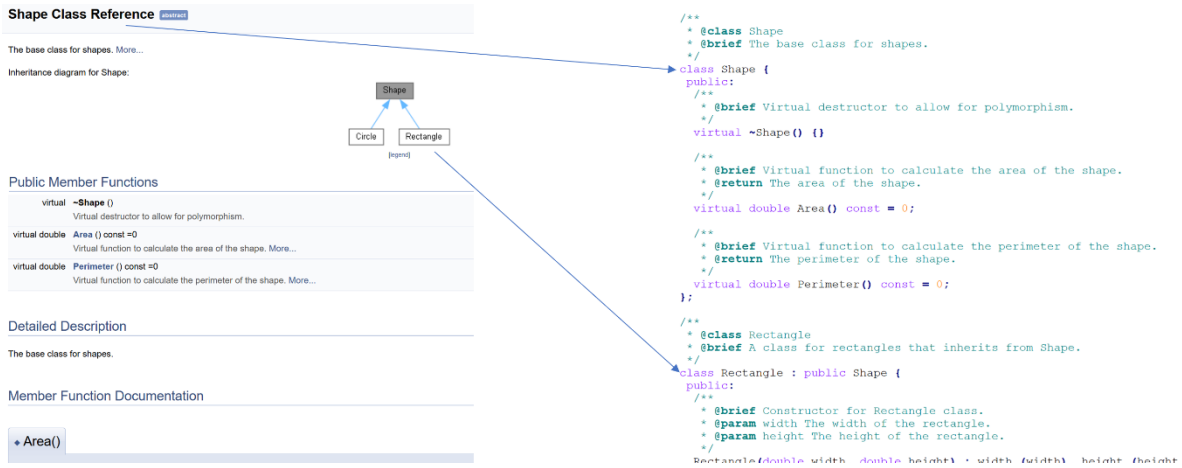


2.5. Diagrams

You can use diagrams in doxygen to visualize relationships between entities in the code, such as class inheritance and references. You need to configure DOT in the configuration file.



This figure demonstrates how you can show references in doxygen.



Here you can see how you can show class inheritance in doxygen using diagrams.