Using Doxygen: Quick Guide

Jignesh M. Patel
Computer Sciences Department
University of Wisconsin—Madison
jignesh@cs.wisc.edu

Doxygen is a popular tool to document your code, i.e. it is a "**documentation system**." Doxygen can be used to generate code for a variety of languages, including Java and C++. In this class we will use it extensively for the C++ projects.

Birds-eye view of how Doxygen works

There are two main steps in using Doxygen:

1. To use Doxygen, you write comments in code using the format that Doxygen understands. The comments are included in the header files (.h) files. But, you should still comment code in your .cpp files, though Doxygen won't use them extensively.

   So, you need to recognize the benefits and limitations of Doxygen. It is great at generating the documentation for the class definitions (the member variable, methods, etc.), class hierarchies (inheritance hierarchy), etc. But, it does not do much with documenting the algorithm (which is typically what you have in your .cpp files). You should still clearly outline your algorithm in the .cpp file, to make your code easy to read.

2. Then, you simply run Doxygen, which generates an html folder. Now you go to that folder and click on the index.html file. The documentation for your code is now in an easy to read html file.

   Doxygen is already installed on the CS instructional machines (e.g. the mumble machines).

Example using Doxygen

Here is a sample .h file commented with Doxygen[1] using the JavaDoc style of formatting. (You can use other styles of documentation with Doxygen, but the JavaDoc style is the most popular.)

```
/**
 * @file
 * @author Jan Doe <jandoe@example.com>
 * @version 1.0
 *
 * @section LICENSE
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License as
 * published by the Free Software Foundation; either version 2 of
 * the License, or (at your option) any later version.
 *
 * @section DESCRIPTION
 *
 * The time class represents a moment of time.
 */

/**
 * @brief A sample Time class
 * @author Future DB Guru
 *
 * This is a simple class to demonstrate how Doxygen is used.
 * It implements a dummy Time class.
 */
class Time {

    public:

        /**
         * Constructor that sets the time to a given value.
         *
         * @param timemillis Number of milliseconds
         *        passed since Jan 1, 1970.
         */
        Time (int timemillis) {
            // the code
        }

        /**
         * Get the current time.
         *
         * @return A time object set to the current time.
         */
        static Time now () {
            // the code
        }
};
```

[1] Adapted from the example at http://en.wikipedia.org/wiki/Doxygen

Copy the code above in a file called try.h. Then run the following command to generate a Doxygen configuration file:

```
doxygen -g
```

This command creates a file called "Doxyfile" This is a configuration file, which you can now edit. The main things to edit here are the name of the project, so edit the file to set PROJECT_NAME to

```
PROJECT_NAME = "CS564 Assignment 0: Try Doxygen"
```

Next, set the INPUT variable to point to the current directory.

```
INPUT = .
```

The final step is to generate the documentation, by running:

```
doxygen Doxyfile
```

The above command will create an html directory, with a file called index.html. Open that file in a browser, and you can read the documentation for your code.

There are many other parameters that you can customize and set in the Doxyfile, and I encourage you to take a look at the options by browsing the comments in that file.

As a next step, build on the simple program above to add classes with inheritance, and observe how Doxygen generates the documentation. Add a .cpp file and see how Doxygen handles that. Do you see what I mean by Doxygen not being a magical solution to documenting the algorithm? In other words, you still have to add comments in the main body of your function (usually in your .cpp file, but could be in your .h file for inline functions) to allow the reader to make sense of your code.

Doxygen Syntax

A quick way to understand the main syntax that you need for C++ documentation is at: http://www.stack.nl/~dimitri/doxygen/docblocks.html. As you see here, there are different styles for documenting. I like JavaDocs, but it's okay if you want to use another style.

The full manual is at: http://www.stack.nl/~dimitri/doxygen/manual. You can use the manual as a reference and go back to it as needed.