Conditional Compilation*

Kenneth Leroy Busbee

Abstract

An introduction and example of conditional compilation as used within the C++ programming language.

1 Overview

As you proceed in your programming career, the problems/tasks that need solving become more complex. The documentation of the algorithm done in pseudo code (or some other method) will still need to be converted into a programming solution. Inevitably, when writing that source code mistakes will be introduced. When learning the syntax of a new programming language, programmers sometimes automatically think in their old language syntax, and make mistakes that are sometimes hard to detect.

The concept of using a flag to either activate or have remain dormant certain lines of code designed solely to help with the debugging of a program has existed since almost the beginning of modern computer programming (1950's). One of the debugging tools available within C++ is **conditional compilation**. For our flag, we would use a defined constant like:

```
#define DEBUG 1
```

Then using another compiler directive pair, the #if and #endif, we can have the compiler during the pre-processor either include or not include one or more lines of code.

```
#if DEBUG
  cout ≪ "\n***** DEBUG Code ** Hi mom!";
#endif
```

Of course saying "Hi mom!" is not very useful for debugging your code. However, you can use test data with conditional compilation. A series of input data values and a series of output predictors can be placed in the program. Then you can turn on the debug feature or turn them off with your debugging flag.

You should study the demonstration program in conjunction with this module.

*Version 1.5: May 11, 2010 11:42 am +0000
†http://creativecommons.org/licenses/by/3.0/

2 Demonstration Program in C++

2.1 Creating a Folder or Sub-Folder for Source Code Files

Depending on your compiler/IDE, you should decide where to download and store source code files for processing. Prudence dictates that you create these folders as needed prior to downloading source code files. A suggested sub-folder for the **Bloodshed Dev-C++ 5 compiler/IDE** might be named:

- Demo_Programs

If you have not done so, please create the folder(s) and/or sub-folder(s) as appropriate.

2.2 Download the Demo Program

Download and store the following file(s) to your storage device in the appropriate folder(s). Following the methods of your compiler/IDE, compile and run the program(s). Study the source code file(s) in conjunction with other learning materials. You may need to right click on the link and select "Save Target As" in order to download the file.

Download from Connexions: Demo_Conditional_Compliation.cpp[1]

3 Definitions

Definition 1: conditional compilation
A compiler directive that includes or excludes lines of code based on a Boolean expression.

[1] See the file at <http://cnx.org/content/m22204/latest/Demo_Conditional_Compilation.cpp>