Supplement: Visual C++ Debugging

For Introduction to C++ Programming

By Y. Daniel Liang

Note: The screen shots are taken from VC++ 2010. It is the same for the later version.

1 Introduction

The debugger utility is integrated in VC++. You can pinpoint bugs in your program with the help of the VC++ debugger without leaving the IDE. The VC++ debugger enables you to set breakpoints and execute programs line by line. As your program executes, you can watch the values stored in variables, observe which functions are being called, and know what events have occurred in the program. Let us use Listing 2.11, ShowCurrentTime.cpp, to demonstrate debugging. Create a new program named ShowCurrentTime.cpp. The source code for ShowCurrentTime.cpp can be obtained from the book.

2 Setting Breakpoints

You can execute a program line by line to trace it, but this is time-consuming if you are debugging a large program. Often, you know that some parts of the program work fine. It makes no sense to trace these parts when you only need to trace the lines of code that are likely to have bugs. In cases of this kind, you can use breakpoints.

A breakpoint is a stop sign placed on a line of source code that tells the debugger to pause when this line is encountered. The debugger executes every line until it encounters a breakpoint, so you can trace the part of the program at the breakpoint. Using the breakpoint, you can quickly move over the sections you know work correctly and concentrate on the sections causing problems.

There are several ways to set a breakpoint on a line. One quick way is to click the cutter of the line on which you want to put a breakpoint. You will see the line highlighted, as shown in Figure 1. You also can set breakpoints by choosing Run, Add Breakpoint. To remove a breakpoint, simply click the cutter of the line.

As you debug your program, you can set as many breakpoints as you want, and can remove breakpoints at any time during debugging. The project retains the breakpoints you have set when you exit the project. The breakpoints are restored when you reopen it.
A breakpoint is set in ShowCurrentTime.cpp.

3 Starting the Debugger

To start debugging, set a break point at the first line in the main function and choose Debug, Start (or F5). If the program compiles without problems, debugging starts. You will see two debugging windows (Autos and Call Stack) appearing at the bottom, as shown in Figure 2. If these windows do not appear, choose Debug, Windows to open these windows.
The debugging windows appear in the IDE.

- The Autos window displays variables and expressions from the current line of code, and the preceding line of code. See Figure 3. This window has tabs Locals, Thread, Module, and Watch at the bottom.

- The Locals tab (see Figure 4) displays all local variables in the current block of code. If you are inside of a function, the locals tab will display the function parameters and locally defined variables.

- The Watch tab (see Figure 5) display a variable state until you explicitly remove it from the window. You can add a variable to the watch window by right clicking a variable and selecting “Add Watch”.

Figure 2

The debugging windows appear in the IDE.
Figure 3

The Autos window displays the variable in the current line.
Figure 4

The Locals window displays all variables in the block.
Figure 5

The Watches window displays the variables you want to watch for.

4 Controlling Program Execution

The program pauses at a line called the current execution point. This line is highlighted and has a yellow arrow to the left. The execution point marks the next line of source code to be executed by the debugger.

When the program pauses at the execution point, you can issue debugging commands to control the execution of the program. You also can inspect or modify the values of variables in the program.

When VC++ is in the debugging mode, the Debug menu contains the debugging commands (see Figure 6). Most of the commands also appear in the toolbar under the message pane. The toolbar contains additional commands that are not in the Run menu. Here are the commands for controlling program execution:

- **Step Over** executes a single statement. If the statement contains a call to a function, the entire function is executed without stepping through it.
• **Step Into** executes a single statement or steps into a function.

• **Step Out** executes all the statements in the current function and returns to its caller.

![Image of debugging commands](image)

**Figure 6**

The debugging commands appear under the Debug menu.

NOTE: The debugger is an indispensable, powerful tool that boosts your programming productivity. It may take you some time to become familiar with it, but your investment will pay off in the long run.