Borland C++ Builder Tutorial

V22.0051

Fall 2002

1 Overview

Borland C++ Builder is a **integrated development environment** (**IDE**) used in all phases of C program development. Specifically, it "integrates" several of the programs used to develop C programs, including:

- the text editor;
- the C compiler;
- the C preprocessor;
- the linker; and
- the debugger.

Builder is designed to simplify the development of *large* applications that involve *elaborate* graphical user interfaces. Unfortunately, this makes it somewhat less than ideal for the purposes of this course. We will be developing simple C programs that use a text based interface. Builder has many features that we will never use; but their presence makes Builder more complicated than necessary.

This tutorial covers the *minimum* needed for students to use Builder for their programming assignments. Often there is more than one way of doing something; in situations like this, this tutorial will only mention one (the simplest). The goal is too spend as little time as possible discussing the complexities of Builder so that we have more time to discuss the complexities of C programming.

This is a "hands on" tutorial. To get the maximum value out of it, it is necessary to sit down in front of a PC, and go through it step by step.

2 Getting Started

Create a folder called CPrograms. This folder will contain *all* of the C programs that are written in this course. (It helps to be organized!) Students with their own PCs should create this folder on their hard drive. Students working in the Stern labs should use their personal network (H) drives. Students working in the ITS labs must use a floppy or ZIP diskette.

Download the ZIP file found at the following location:

```
http://cs.nyu.edu/~campbell/Tutorial.zip
```

Extract the contents of the zip file into the CPrograms folder.

3 Building a Program

Start the Borland C++ Builder application. By default, a number of file windows will be opened. Close all of these windows by choosing the menu item File → Close All.

Now open the file called Tutorial.c (this is one of the files contained in the ZIP archive). Choose the menu item File → Open ... – a dialog box will appear. Select the folder CPrograms in the field labeled Look in:. Select the item C file (*.cpp,*.hpp,*.c,*.h). At this point, the dialog box should look like Figure 1. Select the file Tutorial.c and click the Open button.

A message box (Figure 2), asking if you wish for Builder to create a project, will appear. Builder, like most IDEs, requires a project[1] to be created for each program being developed. Click Yes.

An edit window (Figure 3) will appear. The file Tutorial.c contains the C source for a very simple program. The next step is to translate the C source file to an executable file – this is called "building" the executable. Choose the menu item Project → Build Project.

Building actually involves three distinct programs: the preprocessor, the compiler, and the linker. If any one of these programs detects an error in the C source program, building will come to an immediate halt, without producing an executable. The program in the Tutorial.c file contains two (intentional) compiler errors. The error messages produced by the compiler appear in a pane attached to the edit window (Figure 4).

In some cases, but not all, the error will be obvious by examining the error message. Builder also highlights the line of the C source file where it *detects* the error. This is not necessarily the line where the error actually is; the only assumption that can be made is that the error is on *or before* this line.

[1] Technically, a **project** is the complete catalogue of files and resources used in building an application. Fortunately, we do not need to know exactly what that means.

The first error in the program can be repaired by editing the line

```
printf("I am a HAL Nine Thousand computer, Production Number "3.);
```

so that it becomes

```
printf("I am a HAL Nine Thousand computer, Production Number 3.");
```

The second error can be repaired by appending a semicolon (;) to the end of the line

```
return 0
```

After making both of these edits, choose Project → Build Project again. This time, the build will be successful, and an executable will be created.

4 Executing a Program

In principal, the program can be executed by choosing the menu item Run → Run. But doing this will reveal a rather serious problem – the program execution window will disappear the moment the program finishes, making it impossible to see the output produced by the program.

The file Execute.bat, which is the other file contained in the ZIP archive, contains a batch program[2] that can be used to solve this problem. To utilize this program within C++ Builder, a tool must be created for it. Choose the menu item Tool → Configure Tools; a dialog box (Figure 5) will appear. Click the Add button; another dialog box will appear. Fill in the fields of this dialog box as follows:

Title:	`Execute`
Program:	`Execute.bat`
Working dir:	
Parameters:	`$EXENAME`

The dialog box should look like the one that appears in Figure 6. Click OK, followed by Close to close both dialog boxes. Students who are working at home will only need to create this tool once; students working in either the ITS or Stern labs will need to do this *every* session.

Choose the menu item Tool → Execute to execute the program. A program execution window (Figure 7) will appear containing the output produced by the program, as well as the message

[2] A **batch** program is an ASCII file that contains one or more MS-DOS commands.

```
Press any key to continue ...
```

Dismiss this window by striking a key. Choose the menu item File → Quit to exit Builder. A message box (Figure 8) will appear asking if you wish to save the project. Since Builder is capable of recreating the project, there is no reason to do so. Click No.

A Figures

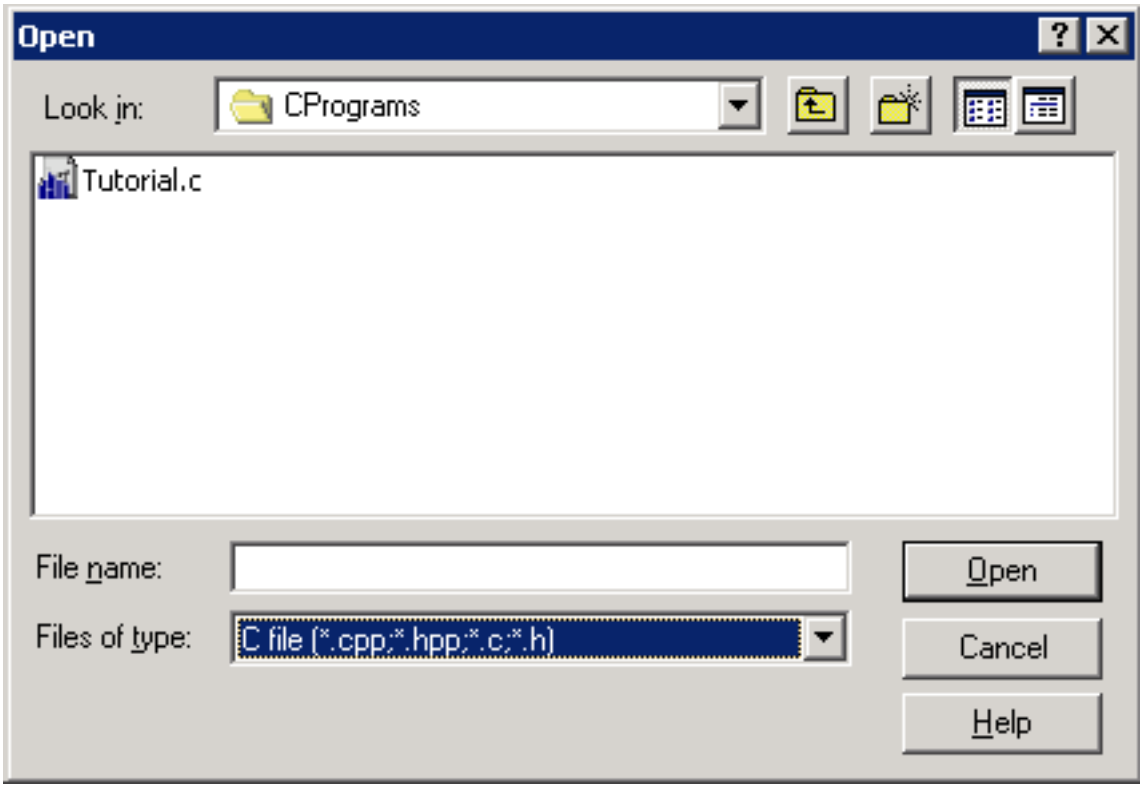

Figure 1: Open File Dialog Box

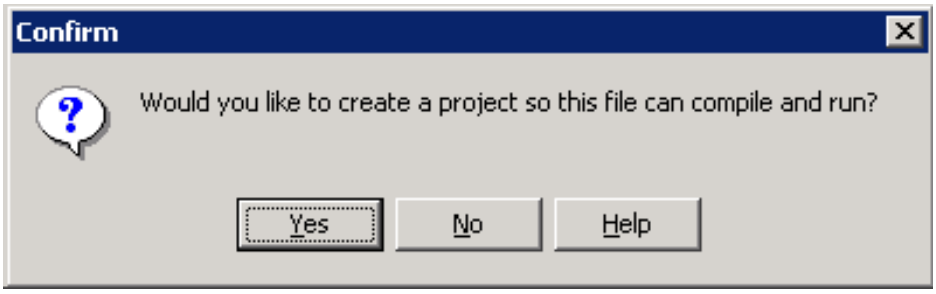

Figure 2: Create Project Message Box

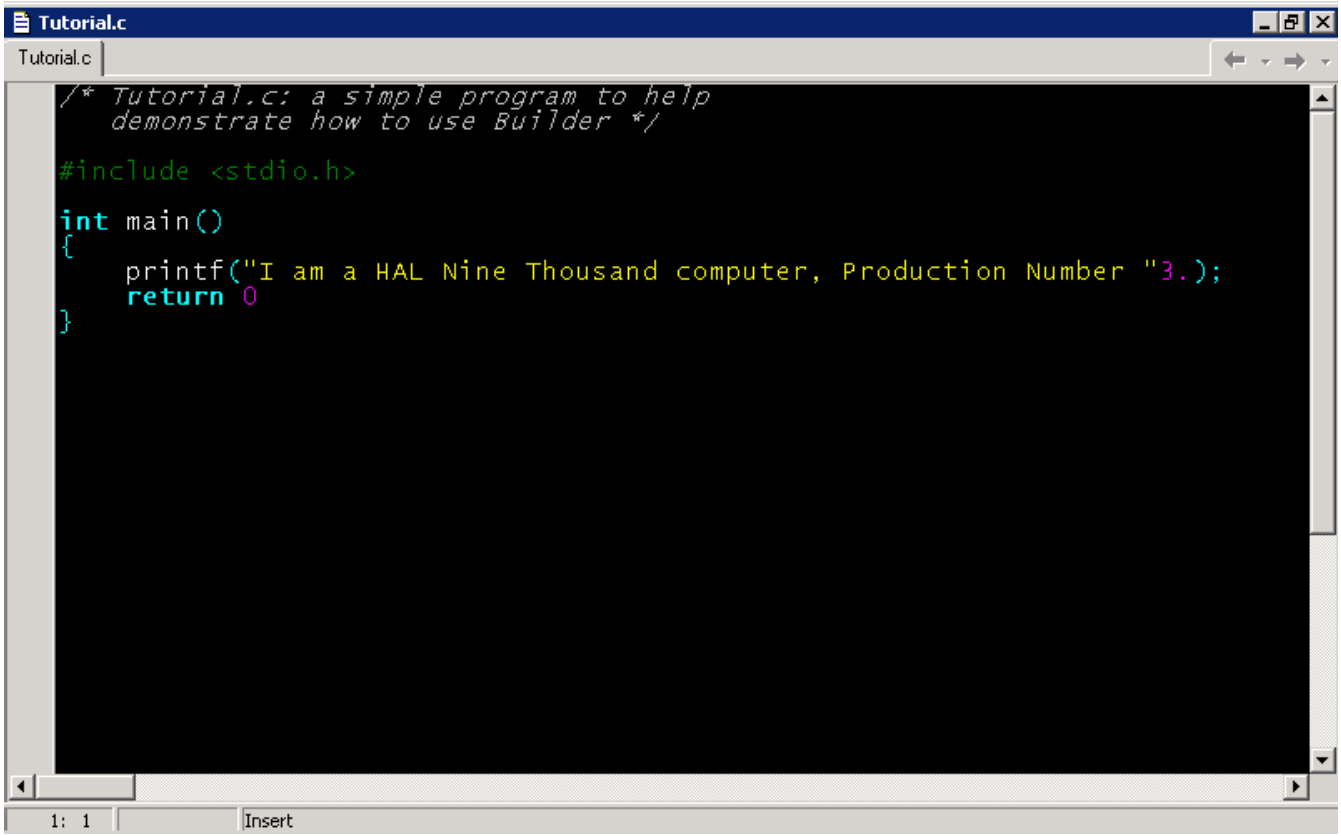

Figure 3: Editor Window

Figure 4: Error Messages Pane

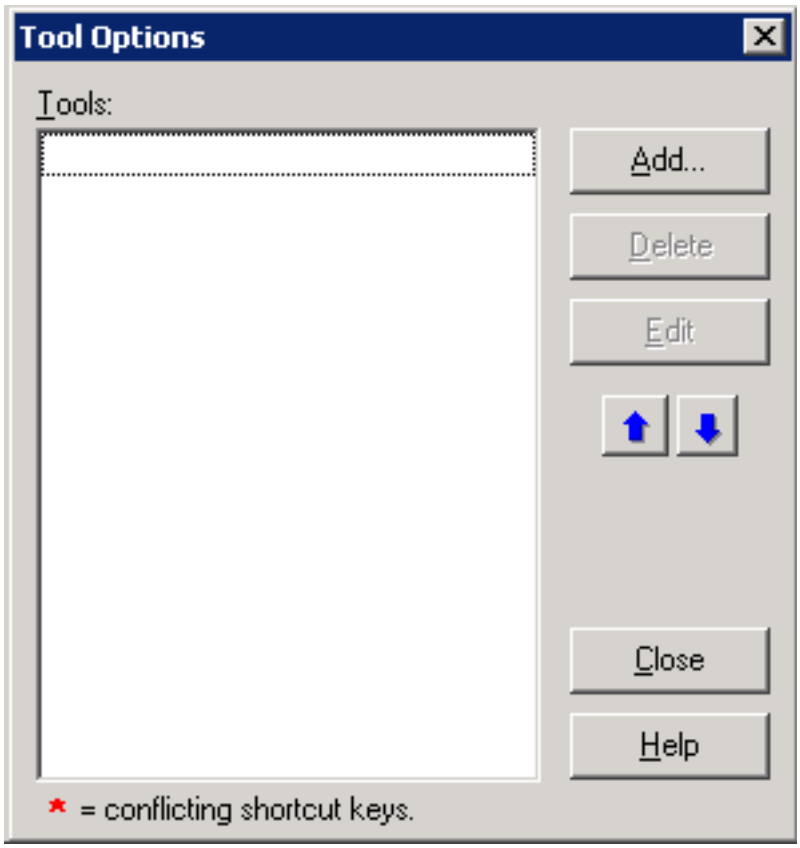

Figure 5: Tool Options Dialog Box

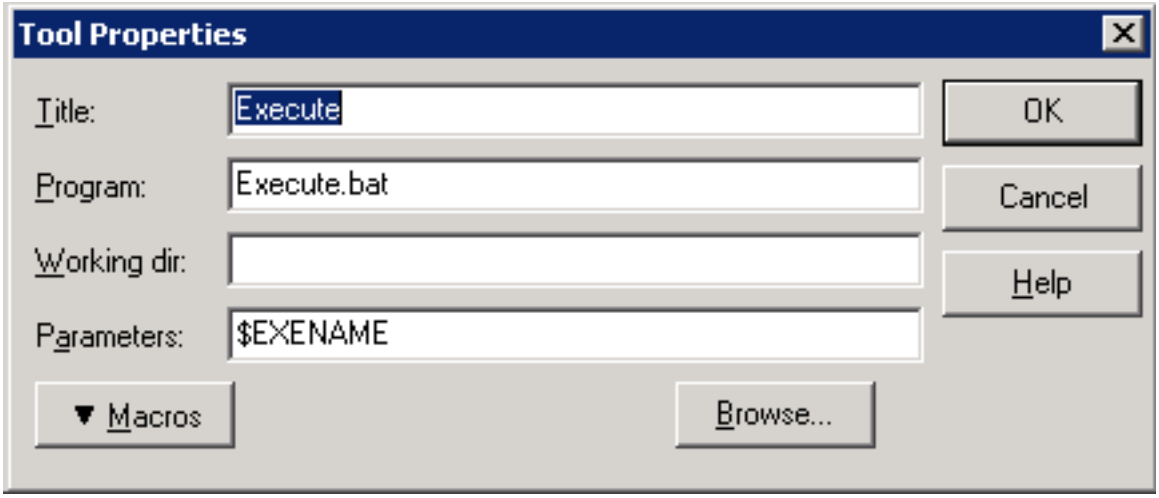

Figure 6: Tool Properties Dialog Box

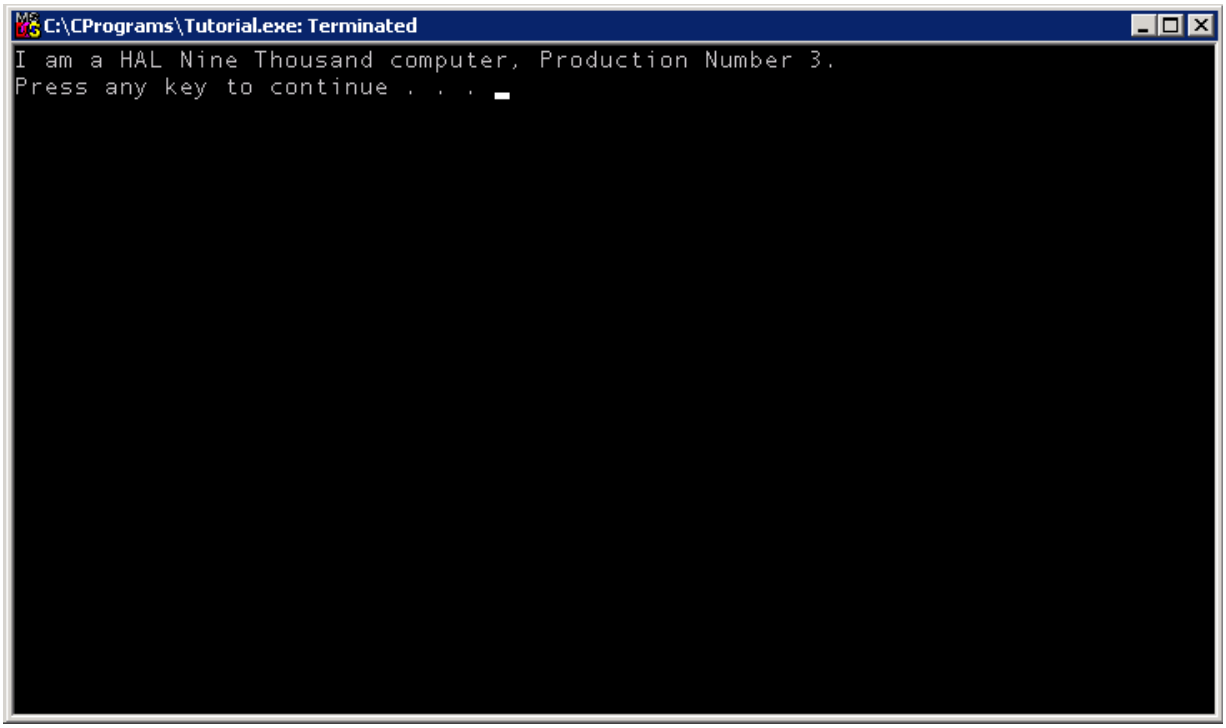

Figure 7: Program Execution Window

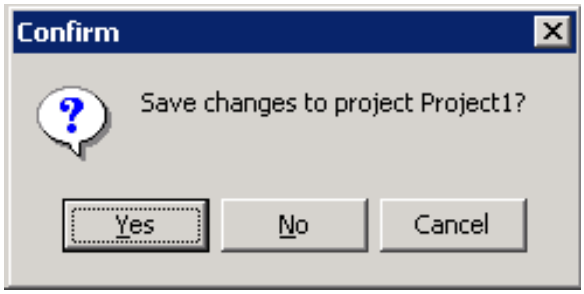

Figure 8: Save Project Message Box

B Summary

1. Start Borland C++ Builder.
2. Choose File → Close All.
3. If you are continuing work on an already existing program, skip to step 7.
4. Choose File → Open, and open an already existing program, such as Tutorial.c.
5. Choose File → Save As, choose a name and save the file.
6. Choose File → Close All.
7. Choose File → Open, and open the C source file (.c extension) you wish to work on.
8. Edit your program by making changes to it in the edit window.
9. Choose Project → Build Project to build your program; if the build results in errors, return to step 8.
10. Choose Tool → Execute to execute your program. If the program does not do what is desired, return to step 8.
11. If you wish to work on another program, return to step 2.
12. Quit Builder.

Whenever asked to create a project, answer Yes.
Whenever asked to save a project, answer No.

Note the suggested method of creating a new program is to make a copy of an already existing one. This can be done *without* leaving Builder (steps 4 to 6).