The source code files for the examples from the book are organized in folders by chapter. Where an example involves multiple files, the set of source files for the chapter is stored in a subfolder of the chapter folder. The name of the example subfolder has the example name that is used in the text. For example, the files for Ex9\_02 in Chapter 9 are stored in the subfolder Ex9\_02 to the chapter folder with the name Ch09.  Note that only source files are included in the code download; there are no project files. Ideally you should create  a project file for each example yourself and create the code files for the example as you progress through the text. Typing in the code will enable you to remember how things work much more effectively than if you just read the code or use the files from the code download directly.  If you get into difficulty with an example, you should first try to discover why it is not working as it should. Only when you have exhausted all possible avenues for fixing the example should you resort to the files in the code download. You have a couple of ways in which you can use these files:  - First, take a look at the file or files for the example and try to see what the differences are. Note   that typos in your code do not necessarily result in a failed compilation. For example, using = where you meant ==   typically will not cause a compiler error message, but the program is unlikely to produce the results you expect.  - Second, if the differences are not apparent to you, you can move your files from the project folder to a temporary   location and substitute the files from the code download for your files in the project folder. When you   have established that the example now works, you can compare your original files with those in the code   download once again to see the differences. You can open your source file alongside the corresponding   file in the project. By floating the two code windows in the editor, you will be able to view both   side-by-side.  Remember, the idea is not simply to get the example working, but to understand how the code works and how you fix it when it doesn't. You will learn much more from making mistakes than when everything just works. Keep in mind that it is not just the code that can be the cause of a problem. If you don't select the correct project type when you first create the example, things are unlikely to work as they should. If you have not set  the project options correctly, this can also result in your program not behaving as it should.  Best of luck!  Ivor Horton