

# How to use Dev-C++

## ● Introduction

Dev-C++ is a full-featured integrated development environment (IDE), which is able to create Windows or DOS-based C/C++ programs using the Mingw compiler system (included with the package), or the Cygwin compiler.

These are the recommended requirements of Dev-C++:

**Microsoft Windows 98, NT or 2000**  
**32 MB RAM**  
**233 Mhz Intel compatible CPU**  
**45 MB free disk space**

Dev-C++ allows you to write, compile and run a C or C++ program.

C++ programming language is an enhanced version of C language that provides **object-oriented programming (OOP)** capabilities. It is a superset of C, which means that you can use a C++ compiler to compile C programs. Object oriented programming techniques differ significantly from the sequential programming used in C programming language. Although a C++ compiler like Dev-C++ allows you to compile a C program that includes some features of C++, in this course we will concentrate on **C programming language**. A program written in pure C language may be compiled and run using other C compilers, like Turbo C etc.

## ● Dev-C++ interface

When you click on the Dev-C++ icon



on your desktop, the program window opens (Figure 1).

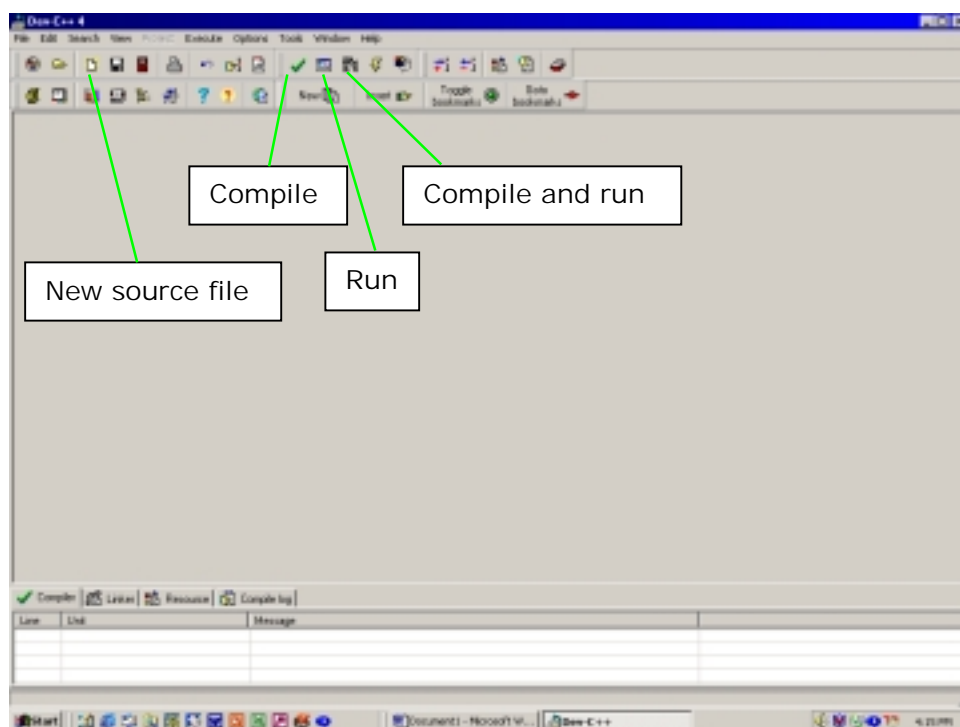


Figure 1.

Click on each menu on the Menu bar and study the menu options. Note the icons that are shown next to the menu items. You can find the identical icons on the toolbar.

In the first part of the course you will write, compile and run a program that is written as a single source file. For this you will use only a few buttons (or menu options) from the toolbar. Later you will learn how to create a project that consists of many separate files.

Before you write your first C program we customise some of the Environmental settings of the IDE.

1. Choose **Options => Compiler Options** from the toolbar. The following dialog box appears (Figure 2):

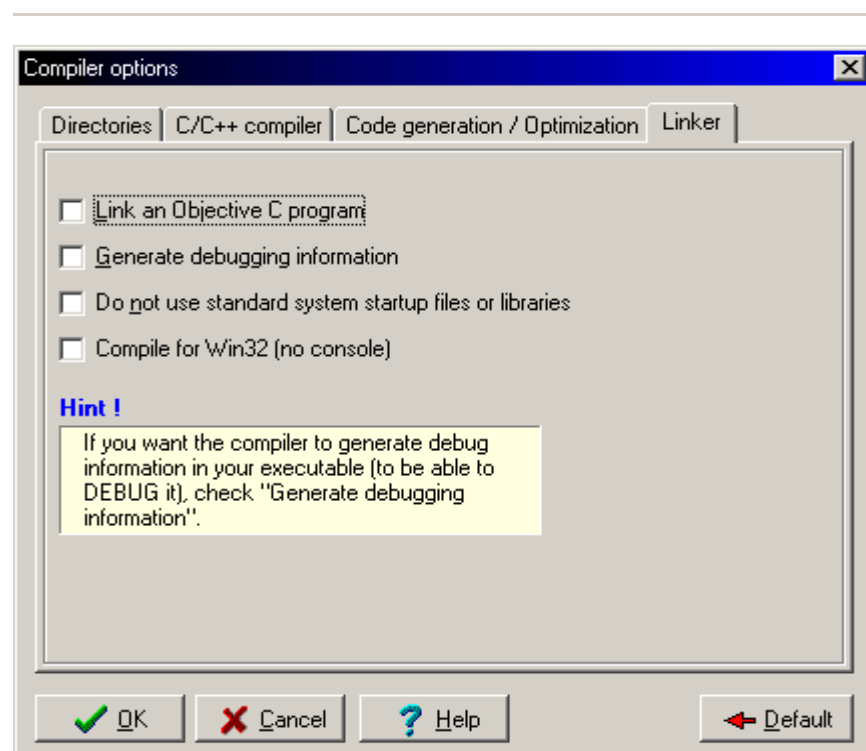


Figure 2.

Click on the **Linker** tab on the top and **uncheck** all the boxes. Most importantly make sure that the **Compile for Win32 (no console)** box is unchecked.

2. Click on the **Options => Environment Options** on the Menu bar. From the dialog box that appears choose **Editor** (Figure 3).

Choose any background colour you like. Select font "**Courier New**" size **10**. Check on "**Line Numbers**". Uncheck "**Auto-Indent**" and "**Use smart Tabs**". Check "**Tabs to Spaces**" and type or select **4** in the "**Number of space for a tab**" box. The dialog box now should look like the one in Figure 3.

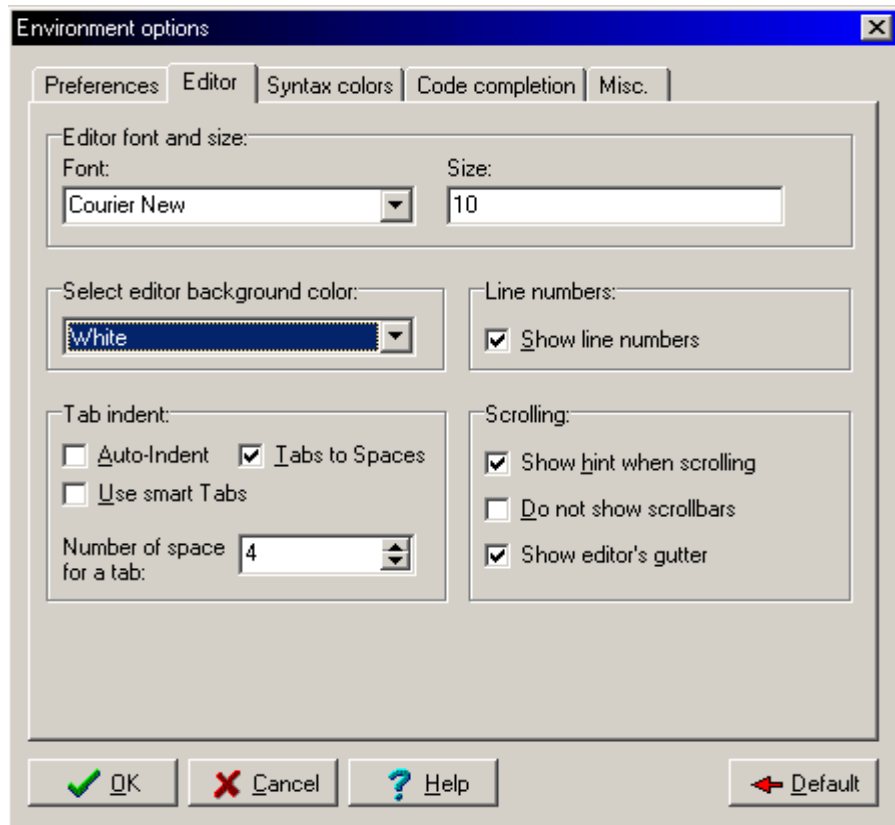


Figure 3.

3. Click on the "Misc." tab on the top. The dialog box changes as shown (Figure 4).

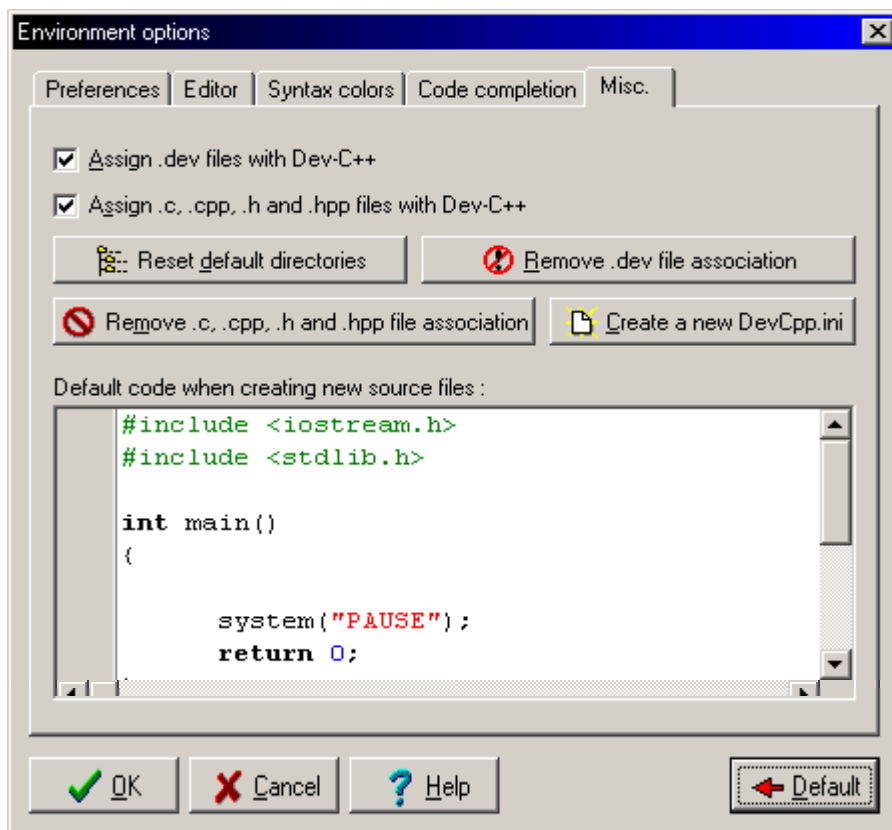


Figure 4.

Make sure that the two options at the top are checked. Now if you double-click on a C source file (for instance, myProgram1.c) in the File Manager, the Dev-C++ program window will open automatically.

The few lines of code that appear in the center are the default templates. This is a C++ template. To change it to C template replace the first line of the code with **#include <stdio.h>**. Move this first line down and at the very top type the following: **/\* Insert comments here \*/** and click **OK**. You can also adjust indent to 4 spaces.

Now you are ready to write your first C program.

## ● Using Dev-C++ compiler

- Select **File => New source file** from the Menu bar or click on the “**New source file**” button on the toolbar. The editor window opens with the template of a C source code.
- Replace the words “*Insert comments here*” with your name and the name of the program. State the purpose of this program inside of the comments.
- Type your code between the opening brace of the **main()** function { and the statement **system("PAUSE")**. This last statement will prevent the DOS-prompt window from closing automatically when your program finishes. It will ask the user to press **Enter** to continue. The “**pause**” command is specific for Windows/DOS and will not work in UNIX environment.

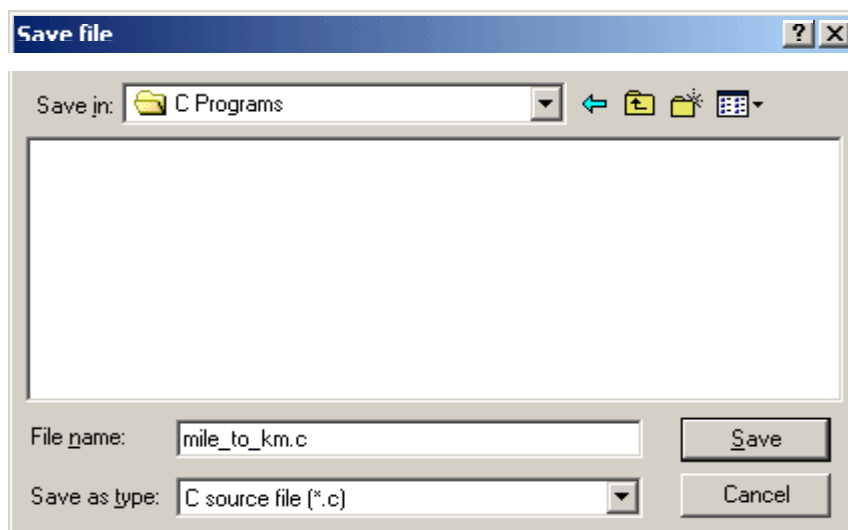


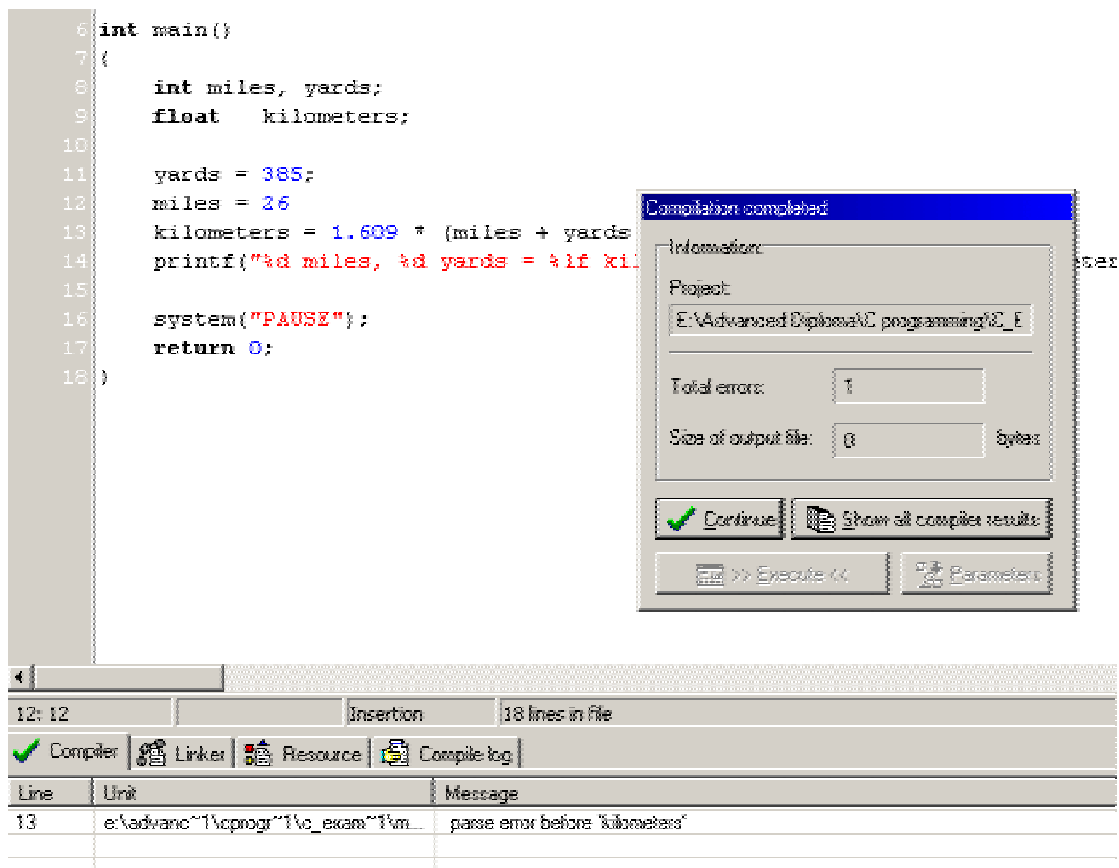
Figure 5

- Click on the **Save** button or select **File => Save unit as...** (Figure 5). On your home drive create a directory **C Programs** and save the file as **C source file** (not C++!).



**NOTE** that you may not be able to create an executable file on a floppy disk because of the memory limitations. Always save your C program on a hard drive. Also, **DO NOT** save your programs in the Dev-C++ \bin directory.

- Click **"Compile"** button or select **Execute => Compile** option from the Menu bar. The Compilation dialog box appears (Figure 5). In this case a program has one syntax error. The Compiler results are shown at the bottom of the window. The number at the left indicates the line number where the error occurs; in this case this is line 13. **"Parse"** error means that this is a syntax error. In this case the semicolon is missing after **26**.
- Click **"Continue"** to return to the editor window, correct the error(s) and compile the program again.



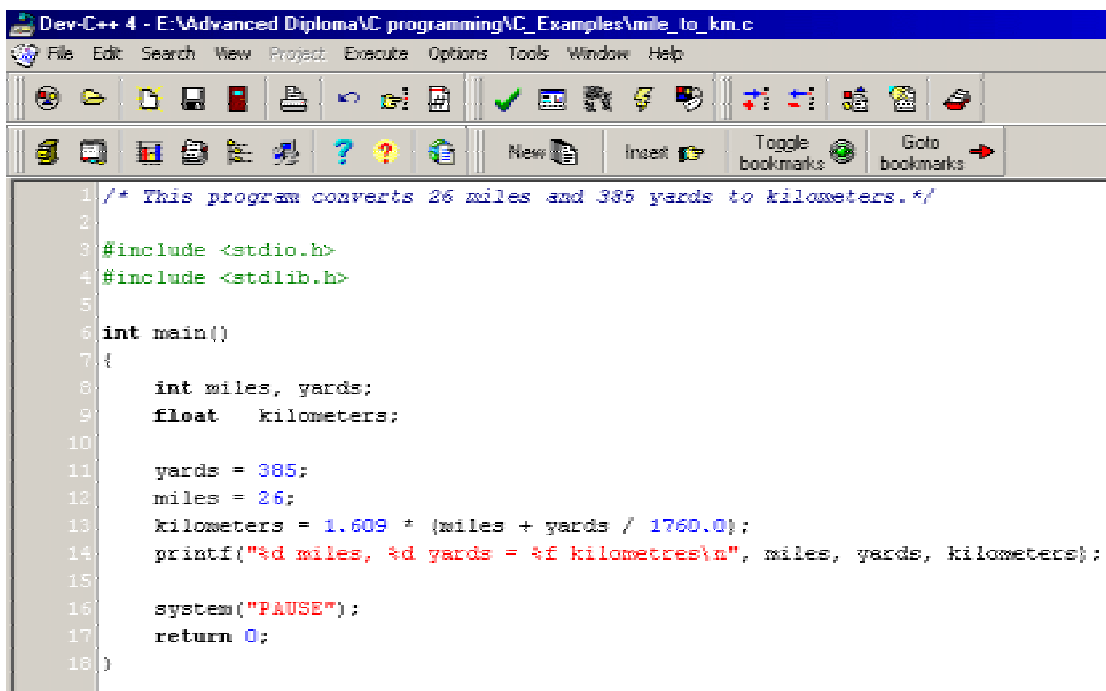
When your program compiles successfully the executable file is created in the same directory where the source file is. It has the same name as the source code file and extension .exe. You can use the **Execute** button on the **"Compilation Completed"** window to run this executable file. Also you can click on **"Continue"** to return to the editor and press the **"Run project"** button on the toolbar.

As with any other Windows application, you can run your program from the DOS-PROMPT by typing the complete path to the executable file and the file name and pressing Enter:

**H:> \C Programs\miles\_to\_km**

The complete program, which is used in this example, is shown below (Figure 6), and the output of this program is presented in the Figure 7.

To close the DOS window press Enter.



```
1 /* This program converts 26 miles and 385 yards to kilometers.*/
2
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 int main()
7 {
8     int miles, yards;
9     float kilometers;
10
11     yards = 385;
12     miles = 26;
13     kilometers = 1.609 * (miles + yards / 1760.0);
14     printf("%d miles, %d yards = %f kilometres\n", miles, yards, kilometers);
15
16     system("PAUSE");
17     return 0;
18 }
```

Figure 6.

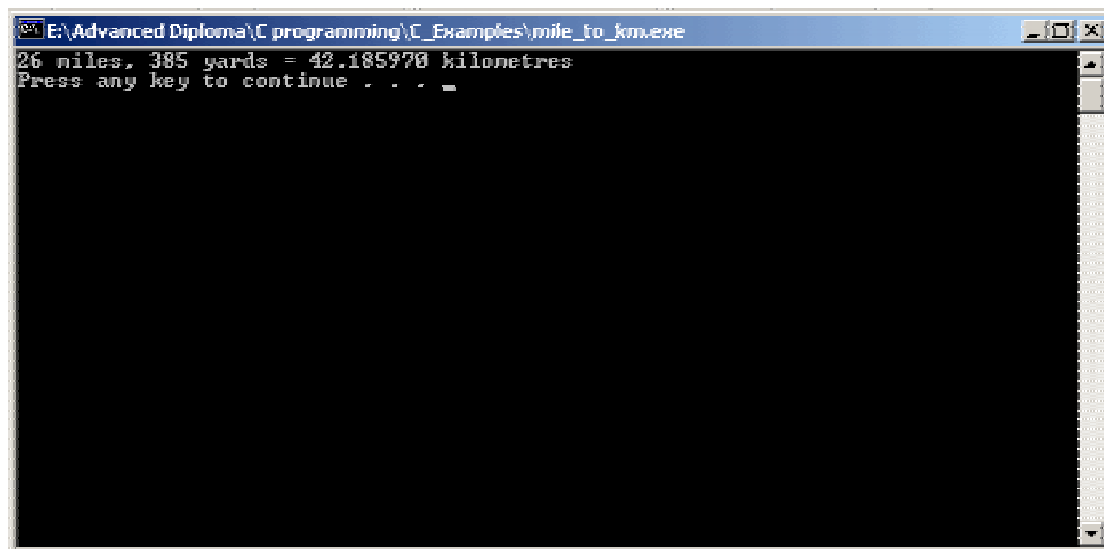


Figure 7.

## ● Making a Project

Using DEV-C++ IDE (Integrated Development Environment) you can develop a **project** that includes as many separate source files as necessary.

Usually a project includes several source files (with extension `.c` or `.cpp`). Each of these files contains one or more functions. One of the files must include the `main()` function. A custom header file usually contains all the function prototypes, references to the library header files and constants' definitions. Each of the source files may now *include* a single custom header. Remember that `#include` statement for the custom header has double quotation marks instead of angle brackets, for example

```
#include "myHeader.h"  
#include <stdio.h>
```

To create a project click on [File => New Project](#). A dialog window appears. Click on [C Project => Console Application](#). Give the project the name, for instance, *Practice1\_1a*, and save it in a new directory on your home drive. The file *Practice1\_1a.dev* is created, and the name of the project appears on the left side of the screen. It has a new source file attached and the same file is open in the editor window. You can now type your program.

If you want to add an existing source file to the project, click on [Project => Add to project](#) on the main menu and choose a file. The name of the file appears, attached to the project name, on the left panel. Double click on the source file will open a separate editor window.

You can remove any file from the project using [Project=> Remove from project](#) option of the menu or clicking the right mouse button on the file at the left side of the screen.

Any time when you make changes to any of the source files, you can compile and build your application with a single mouse click. All files included in the project will be compiled and linked.