Programming Notes 15: Dev C++ Projects ‘n’ Stuff

As usual, more examples of the notes are in the example program.

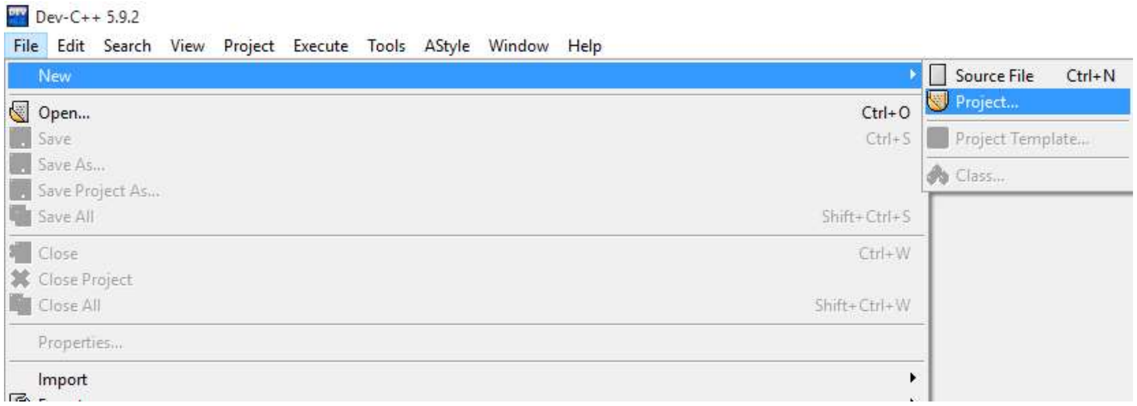
1. Files

As of yet, we’ve been making all of our classes in the same .cpp file as our main function. However, it is conventional and useful to split classes into multiple files. Generally, the definition of your class (meaning the actual “class name” part) goes in its own .h file, and the class implementation (all your name::func functions) go in their own .cpp file. There are several benefits of doing this: first and most obvious, it allows you to much more easily organize your code. Second, if other people are using your classes—or you, to be honest—you can easily open the .h file of your class to see what and how to use the member functions of your class. Finally, because your classes are in separate files, the compiler only has to compile the code that has changed. This means that the first time you compile your program, it will compile each class (and the main file) individually, and henceforth will only need to recompile specific parts of your program that change. For example, if you change one class, only that one class will need to be recompiled, vastly speeding up compile times. Finally, because your class definition is included in an .h file, simply #include that file wherever you need to use your class. Note that because it’s a file that you’ve created, you should use quotes (“”) instead of pointy brackets (<>). This also means that because you only have to include class definitions where you use them, you may not need to include all of your classes in your main file. For example, if you had a “card” and a “deck” class, the “card” class would obviously need to be included in the “deck” class, but if your main file does not have to create cards, it does not have to be included in your main file.

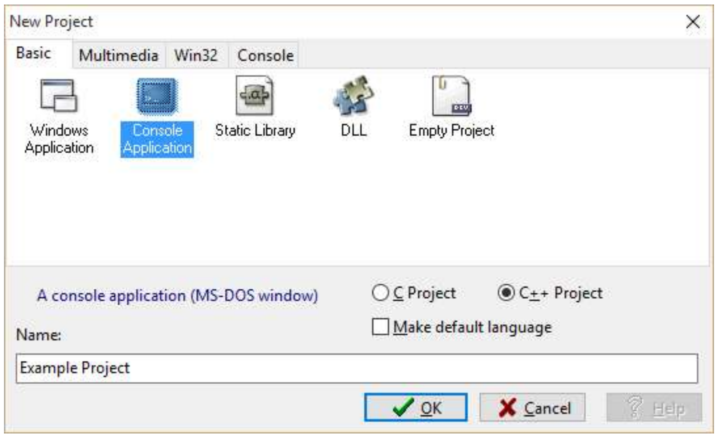
2. Dev C++ Projects

Dev C++ makes it pretty easy to manage projects with many files using their “project” feature. Here are a bunch of screenshots showing how to set up a project:

Create a project:



Select "console application," "C++ project," and name your project:



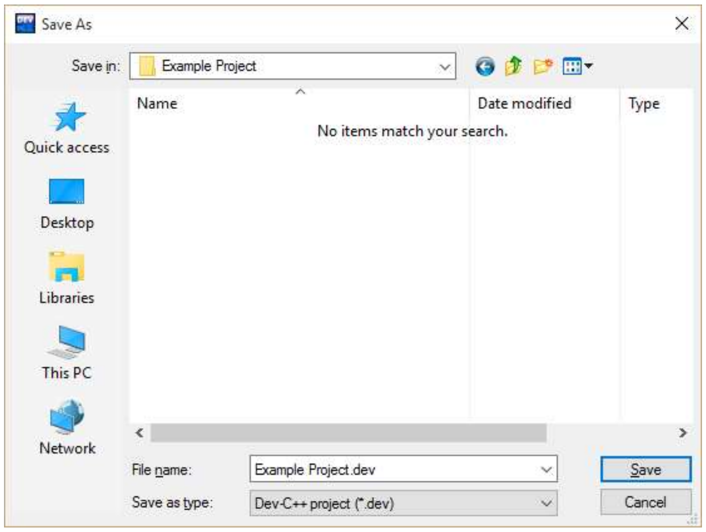
A console application (MS-DOS window)

C Project [✓ C++ Project]

Make default language

Name: Example Project

Save your project in a new folder:



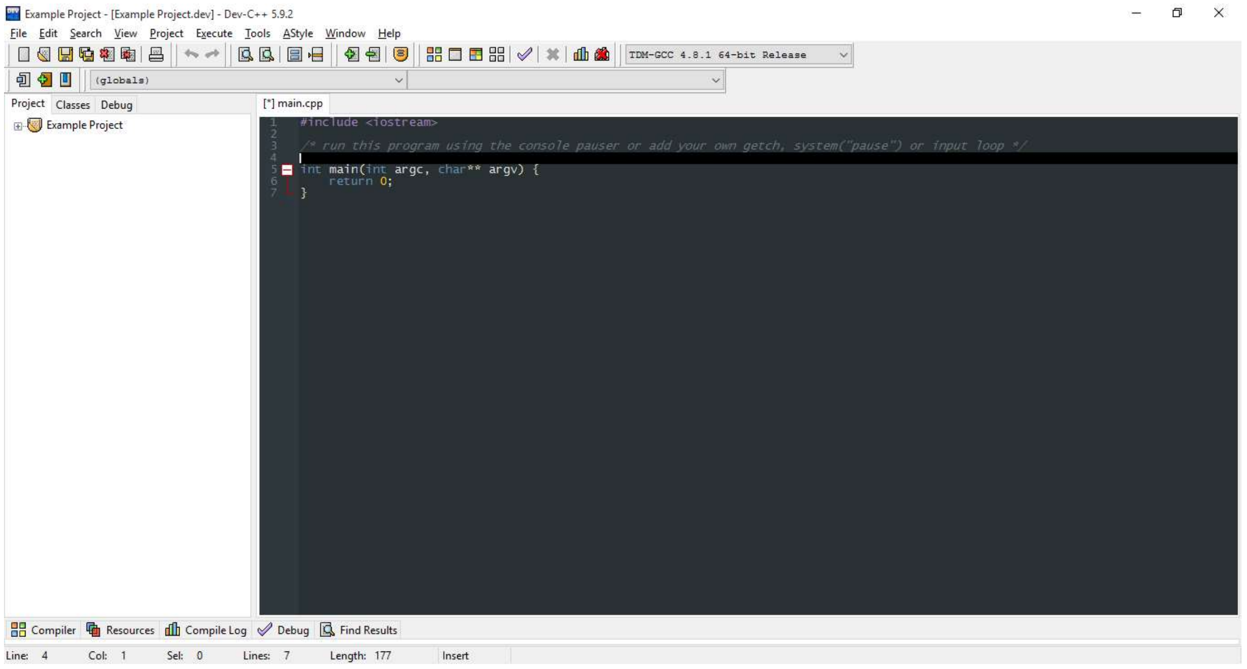
Save in: Example Project

No items match your search.

File name: Example Project.dev

Save as type: Dev-C++ project (*.dev)

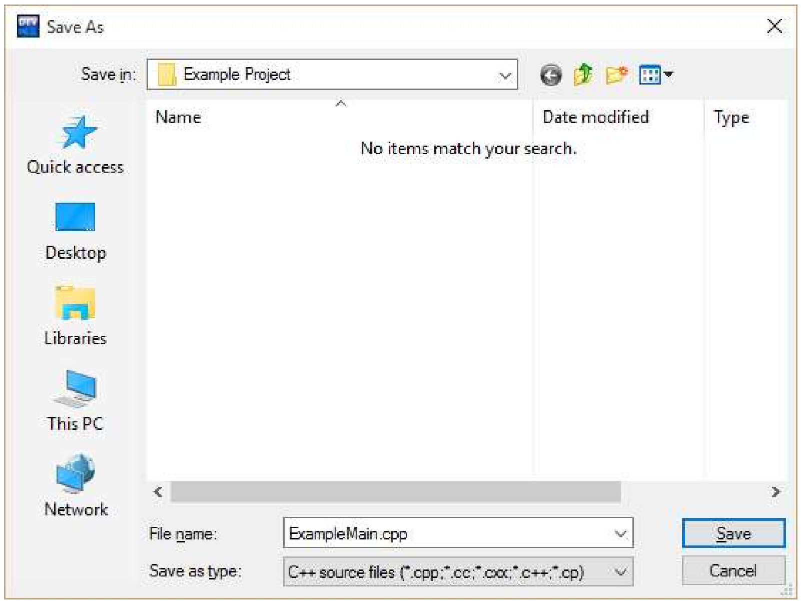
Your project should now look like this:


Example Project - [Example Project.dev] - Dev-C++ 5.9.2
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.8.1 64-bit Release
(globals)
Project Classes Debug
Example Project
[*] main.cpp
Compiler Resources Compile Log Debug Find Results

```
#include <iostream>

/* run this program using the console pauser or add your own getch, system("pause") or input loop */

int main(int argc, char** argv) {
	return 0;
}
```

```
Line: 4    Col: 1    Sel: 0    Lines: 7    Length: 177    Insert
```

Save the main.cpp file (to the same folder), rename it if you'd like:



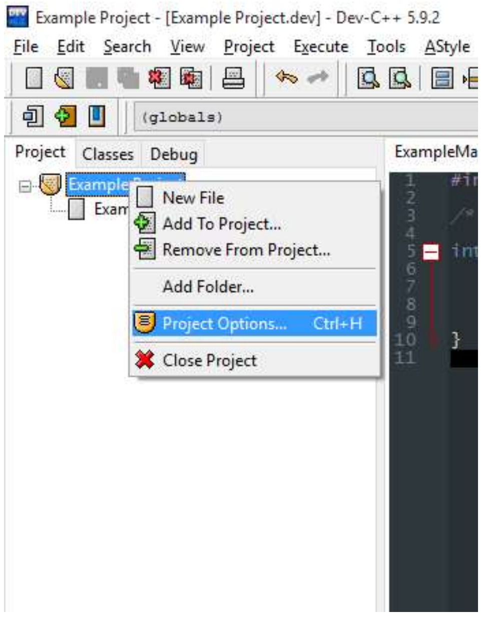
Save in: Example Project

No items match your search.

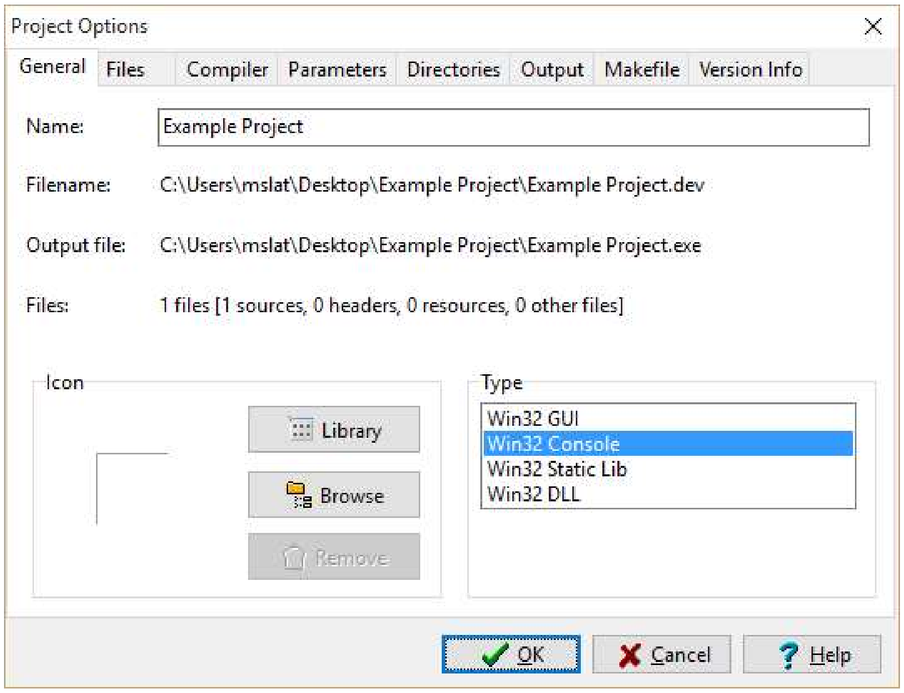
File name: ExampleMain.cpp

Save as type: C++ source files (*.cpp;*.cc;*.cxx;*.c++;*.cp)

You should now have a working project. To change compiler/linker settings:


Example Project - [Example Project.dev] - Dev-C++ 5.9.2
File Edit Search View Project Execute Tools AStyle
(globals)
Project Classes Debug
ExampleMa
New File
Add To Project...
Remove From Project...
Add Folder...
Project Options... Ctrl+H
Close Project

Here you can change project settings like name, type, icon, etc. For now you will only be interested in the "General," "Output," and *maybe* the "Compiler" and "Directories" tabs.



Name: Example Project

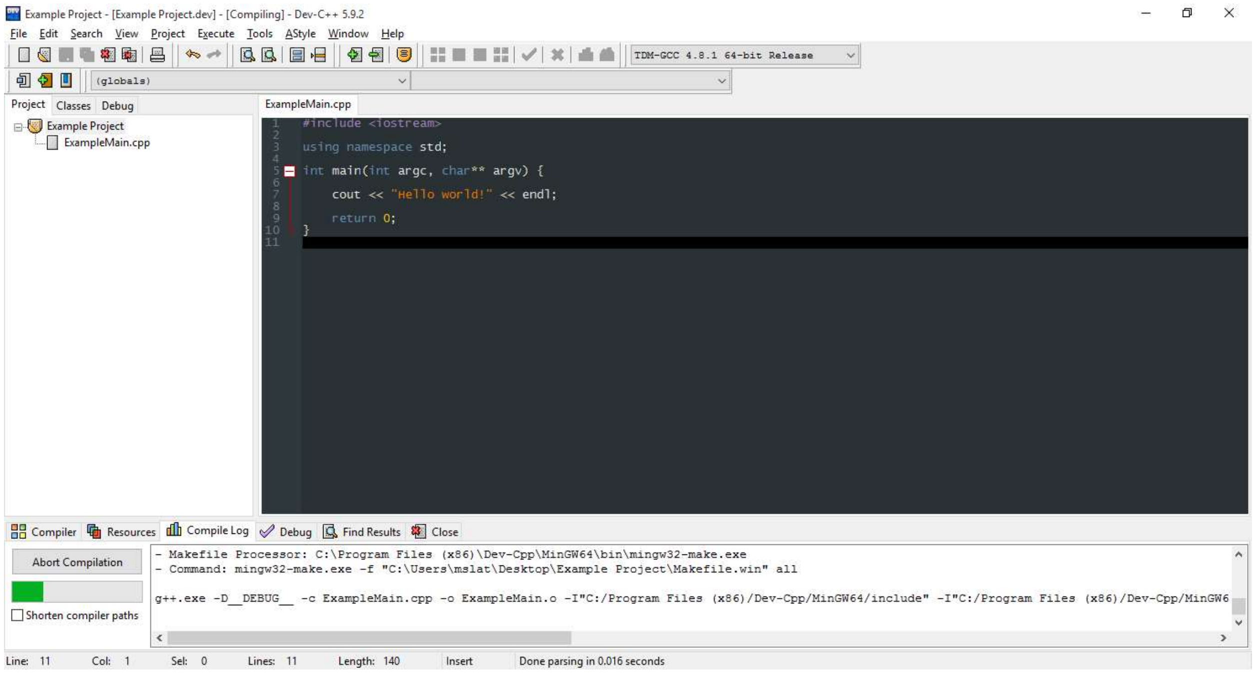
Filename: C:\Users\mslat\Desktop\Example Project\Example Project.dev

Output file: C:\Users\mslat\Desktop\Example Project\Example Project.exe

Files: 1 files [1 sources, 0 headers, 0 resources, 0 other files]

Type: Win32 Console

If you are satisfied with your settings, write a “hello world” program, and press F11 to compile and run:

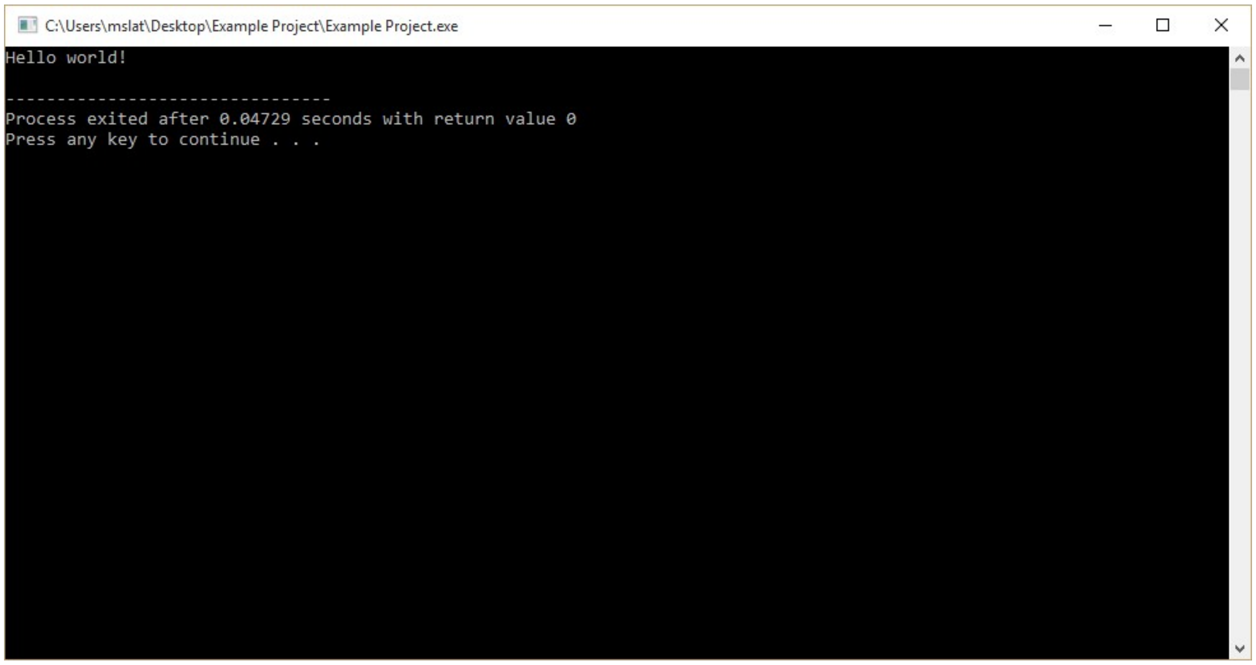

Example Project - [Example Project.dev] - [Compiling] - Dev-C++ 5.9.2
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.8.1 64-bit Release
(globals)
Project Classes Debug
Example Project
ExampleMain.cpp
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Shorten compiler paths

```
#include <iostream>

using namespace std;

int main(int argc, char** argv) {

    cout << "Hello world!" << endl;

    return 0;
}

```

```
- Makefile Processor: C:\Program Files (x86)\Dev-Cpp\MinGW64\bin\mingw32-make.exe
- Command: mingw32-make.exe -f "C:\Users\mslat\Desktop\Example Project\Makefile.win" all

g++.exe -D__DEBUG__ -c ExampleMain.cpp -o ExampleMain.o -I"C:/Program Files (x86)/Dev-Cpp/MinGW64/include" -I"C:/Program Files (x86)/Dev-Cpp/MinGW6
```

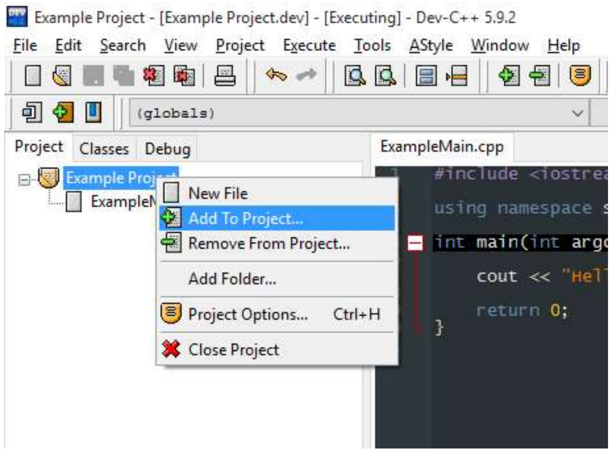
```
Line: 11    Col: 1    Sel: 0    Lines: 11    Length: 140    Insert    Done parsing in 0.016 seconds
```

If you set up your project correctly, you should get what you expect.



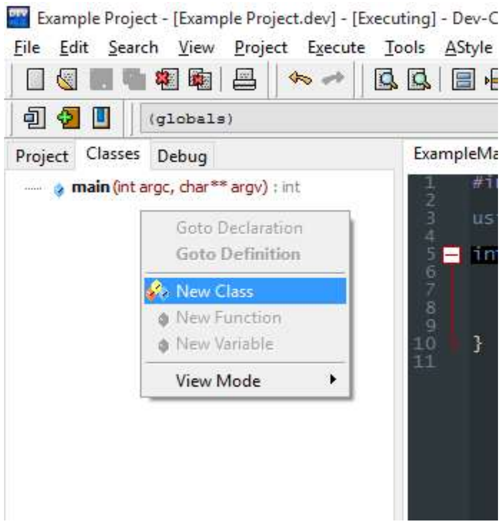
```
Hello world!

--------------------------------
Process exited after 0.04729 seconds with return value 0
Press any key to continue . . .
```

To add non-code files to your project, such as text or data for input:

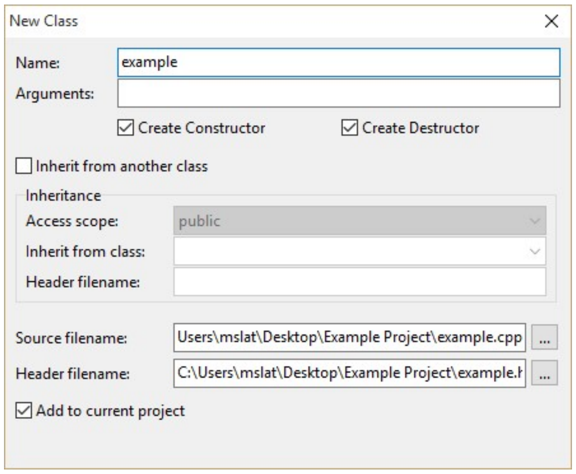

Example Project - [Example Project.dev] - [Executing] - Dev-C++ 5.9.2
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug
ExampleMain.cpp
Example Proj
ExampleM
New File
Add To Project...
Remove From Project...
Add Folder...
Project Options... Ctrl+H
Close Project

```
#include <iostrea
using namespace s
int main(int argc
    cout << "Hel
    return 0;
}
```

Finally, to add a class:

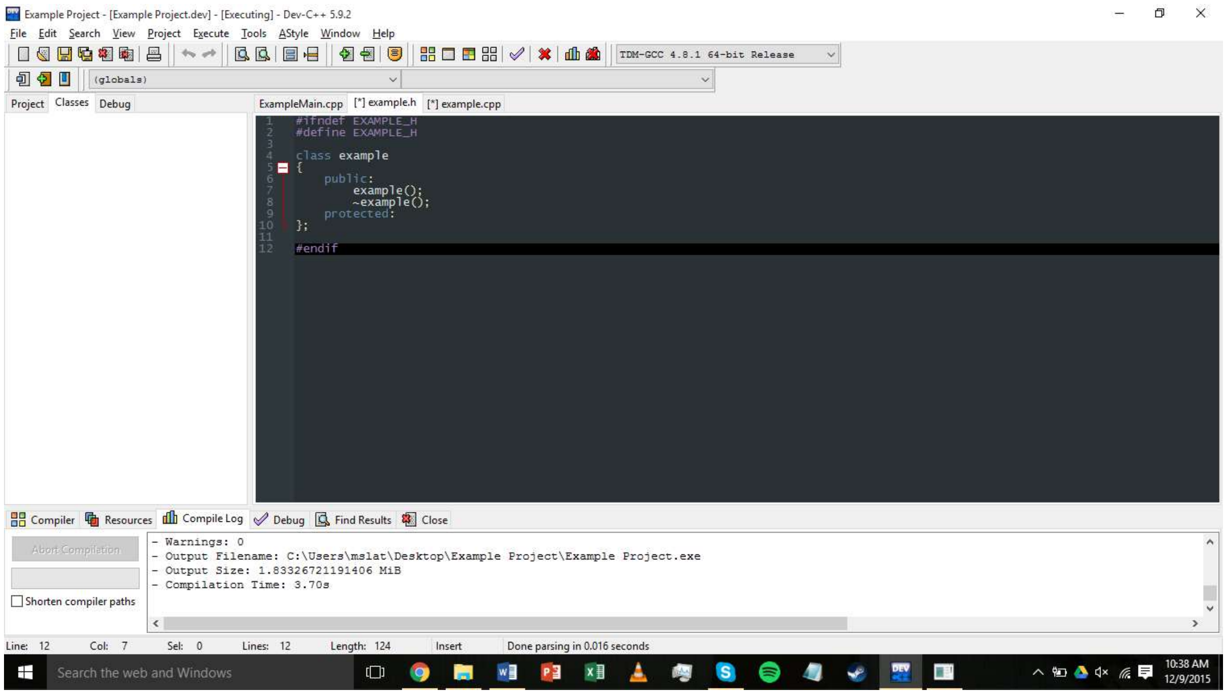

Example Project - [Example Project.dev] - [Executing] - Dev-C
File Edit Search View Project Execute Tools AStyle
(globals)
Project Classes Debug
ExampleMa
main (int argc, char** argv) : int
Goto Declaration
Goto Definition
New Class
New Function
New Variable
View Mode

Here, just name your class, and add a default constructor/destructor:



Name: example
Arguments:
☑ Create Constructor ☑ Create Destructor
☐ Inherit from another class
Inheritance
Access scope: public
Inherit from class:
Header filename:
Source filename: Users\mslat\Desktop\Example Project\example.cpp
Header filename: C:\Users\mslat\Desktop\Example Project\example.h
☑ Add to current project

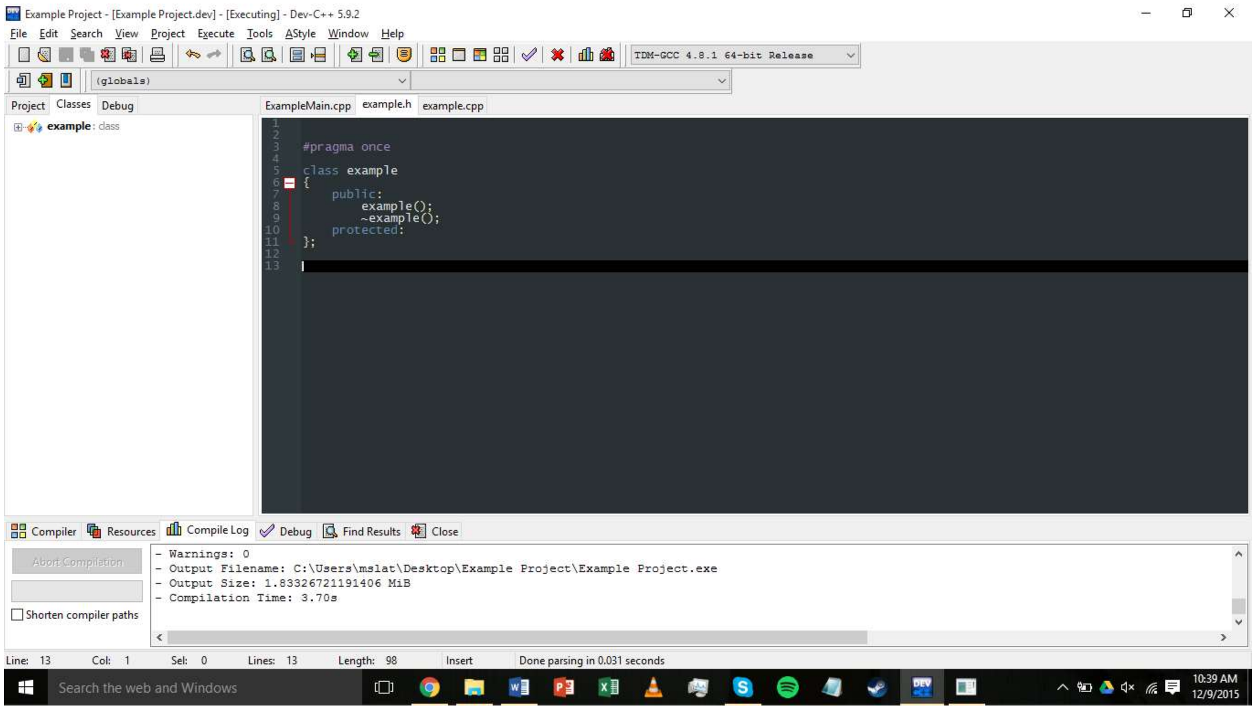
An .h and .cpp file for your class should appear. Save them both to your project folder.


Example Project - [Example Project.dev] - [Executing] - Dev-C++ 5.9.2
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.8.1 64-bit Release
(globals)
Project Classes Debug
ExampleMain.cpp [*] example.h [*] example.cpp
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Shorten compiler paths
Line: 12 Col: 7 Sel: 0 Lines: 12 Length: 124 Insert Done parsing in 0.016 seconds
Search the web and Windows
10:38 AM 12/9/2015

```
#ifndef EXAMPLE_H
#define EXAMPLE_H

class example
{
    public:
        example();
        ~example();
    protected:
};

#endif
```

```
- Warnings: 0
- Output Filename: C:\Users\mslat\Desktop\Example Project\Example Project.exe
- Output Size: 1.83326721191406 MiB
- Compilation Time: 3.70s
```

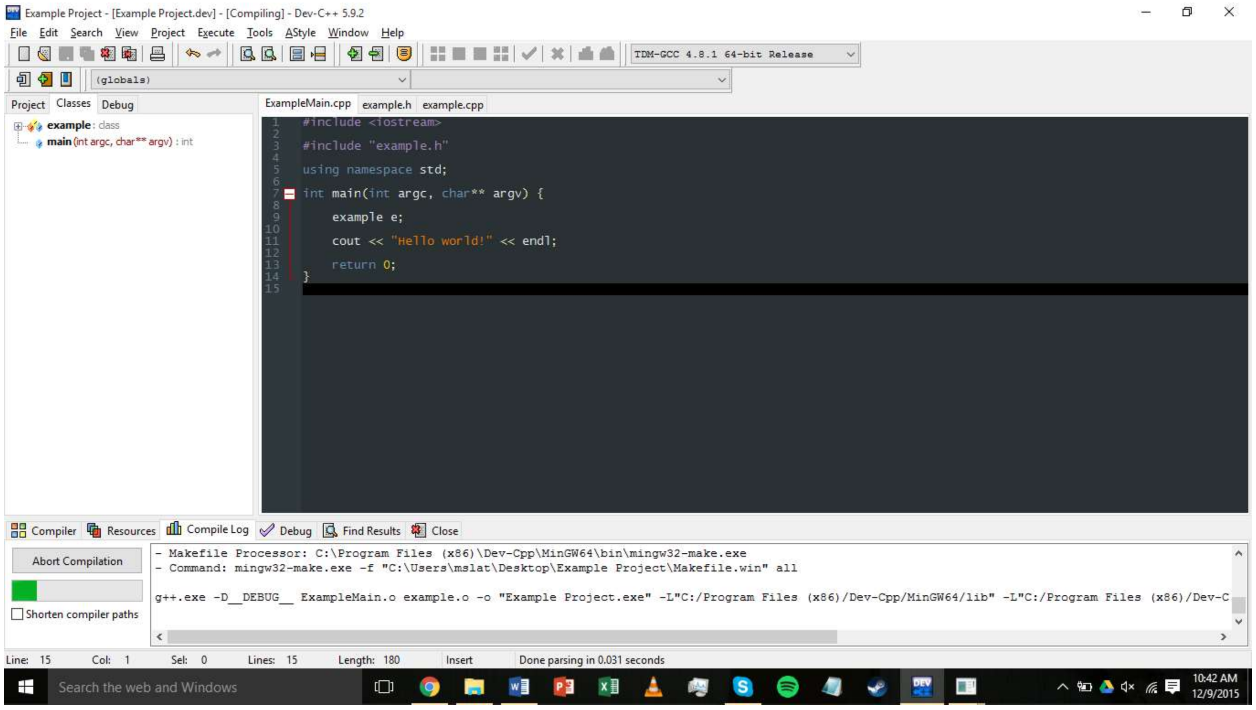
In the .h file, you can replace all of the # statements with "#pragma once" if you'd like. This makes sure that if your .h file would be included more than once, it is only included once:


Example Project - [Example Project.dev] - [Executing] - Dev-C++ 5.9.2
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.8.1 64-bit Release
(globals)
Project Classes Debug
example : class
ExampleMain.cpp example.h example.cpp
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Shorten compiler paths
Line: 13 Col: 1 Sel: 0 Lines: 13 Length: 98 Insert Done parsing in 0.031 seconds
Search the web and Windows
10:39 AM 12/9/2015

```


#pragma once

class example
{
    public:
        example();
        ~example();
    protected:
};

```

```
- Warnings: 0
- Output Filename: C:\Users\mslat\Desktop\Example Project\Example Project.exe
- Output Size: 1.83326721191406 MiB
- Compilation Time: 3.70s
```

To use your class in the main file, simply include your new .h file, using quotes. Try compiling this, to make sure everything is working. To add more classes, simply repeat this process.

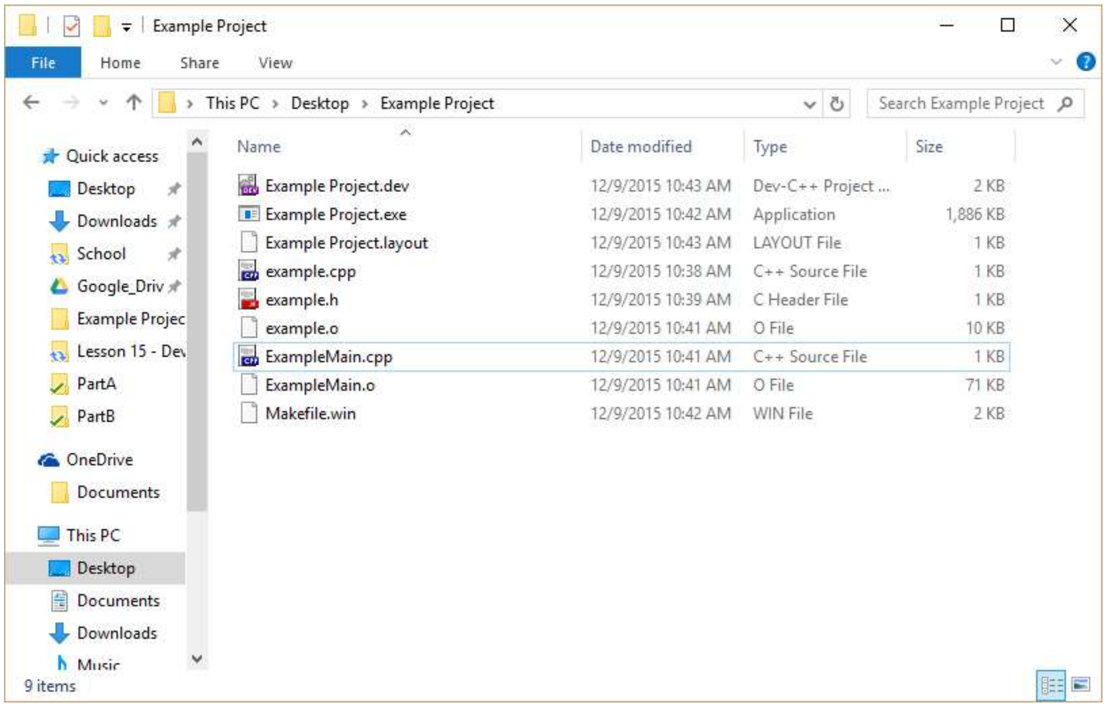


```
#include <iostream>

#include "example.h"

using namespace std;

int main(int argc, char** argv) {

    example e;

    cout << "Hello world!" << endl;

    return 0;
}
```

```
- Makefile Processor: C:\Program Files (x86)\Dev-Cpp\MinGW64\bin\mingw32-make.exe
- Command: mingw32-make.exe -f "C:\Users\mslat\Desktop\Example Project\Makefile.win" all

g++.exe -D__DEBUG__ ExampleMain.o example.o -o "Example Project.exe" -L"C:/Program Files (x86)/Dev-Cpp/MinGW64/lib" -L"C:/Program Files (x86)/Dev-C
```

Line: 15 Col: 1 Sel: 0 Lines: 15 Length: 180 Insert Done parsing in 0.031 seconds

Finally, you are finished! Remember to save your project settings when you close Dev C++. Your project folder should now have several unfamiliar files. The .dev file is your project, and is what you will open when you want to work on your project. The .layout, .o, and makefile files are all byproducts of compiling with multiple files, so leave them be and you should be good.



Name	Date modified	Type	Size
Example Project.dev	12/9/2015 10:43 AM	Dev-C++ Project ...	2 KB
Example Project.exe	12/9/2015 10:42 AM	Application	1,886 KB
Example Project.layout	12/9/2015 10:43 AM	LAYOUT File	1 KB
example.cpp	12/9/2015 10:38 AM	C++ Source File	1 KB
example.h	12/9/2015 10:39 AM	C Header File	1 KB
example.o	12/9/2015 10:41 AM	O File	10 KB
ExampleMain.cpp	12/9/2015 10:41 AM	C++ Source File	1 KB
ExampleMain.o	12/9/2015 10:41 AM	O File	71 KB
Makefile.win	12/9/2015 10:42 AM	WIN File	2 KB

9 items