

# Walkthrough: Compile a C program on the command line

 06/21/2018  9 minutes to read Contributors 

## In this article

[Prerequisites](#)

[Open a developer command prompt](#)

[Create a C source file and compile it on the command line](#)

[Next steps](#)

[See also](#)

Visual C++ includes a C compiler that you can use to create everything from basic console programs to full Windows Desktop applications, mobile apps, and more.

This walkthrough shows how to create a basic, "Hello, World"-style C program by using a text editor, and then compile it on the command line. If you'd rather work in C++ on the command line, see [Walkthrough: Compiling a Native C++ Program on the Command Line](#). If you'd like to try the Visual Studio IDE instead of using the command line, see [Walkthrough: Working with Projects and Solutions \(C++\)](#) or [Using the Visual Studio IDE for C++ Desktop Development](#).

## Prerequisites

To complete this walkthrough, you must have installed either Visual Studio and the optional Visual C++ components, or the Build Tools for Visual Studio.

Visual Studio is a powerful integrated development environment that supports a full-featured editor, resource managers, debuggers, and compilers for many languages and platforms. For information on these features and how to download and install Visual Studio, including the free Visual Studio Community edition, see [Install Visual Studio](#).

The Build Tools for Visual Studio version of Visual Studio installs only the command-line toolset, the compilers, tools, and libraries you need to build C and C++ programs. It's perfect for build labs or classroom exercises and installs relatively quickly. To install only the command-line toolset, download [Build Tools for Visual Studio](#) and run the installer.

Before you can build a C or C++ program on the command line, you must verify that the tools are installed, and that you can access them from the command line. Visual C++ has complex requirements for the command-line environment in order to find the tools, headers, and libraries it uses. **You can't use**

**Visual C++ in a plain command prompt window** without some preparation. You need a *developer command prompt* window, which is a regular command prompt window that has all the required environment variables set. Fortunately, Visual C++ installs shortcuts for you to launch developer command prompts that have the environment set up for command line builds. Unfortunately, the names of the developer command prompt shortcuts and where they are located are different in almost every version of Visual C++ and on different versions of Windows. Your first walkthrough task is to find the right shortcut to use.

### Note

A developer command prompt shortcut automatically sets the correct paths for the compiler and tools, and for any required headers and libraries. Some of these values are different for each build configuration. You must set these environment values yourself if you don't use one of the shortcuts. For more information, see [Set the Path and Environment Variables for Command-Line Builds](#). Because the build environment is complex, we strongly recommend you use a developer command prompt shortcut instead of building your own.


## Open a developer command prompt

1. If you have installed Visual Studio 2017 on Windows 10, open the Start menu, and then scroll down and open the **Visual Studio 2017** folder (not the Visual Studio 2017 app). Choose **Developer Command Prompt for VS 2017** to open the command prompt window.

If you have installed Microsoft Visual C++ Build Tools 2015 on Windows 10, open the **Start** menu, and then scroll down and open the **Visual C++ Build Tools** folder. Choose **Visual C++ 2015 x86 Native Tools Command Prompt** to open the command prompt window.

If you are using a different version of Visual Studio or are running a different version of Windows, look in your Start menu or Start page for a Visual Studio tools folder that contains a developer command prompt shortcut. You can also use the Windows search function to search for "developer command prompt" and choose one that matches your installed version of Visual Studio. Use the shortcut to open the command prompt window.

2. Next, verify that the Visual C++ developer command prompt is set up correctly. In the command prompt window, enter `cl` and verify that the output looks something like this:

| Output  |  Copy |
|---|--|
| <pre>C:\Program Files (x86)\Microsoft Visual Studio\2017\Enterprise&gt;cl Microsoft (R) C/C++ Optimizing Compiler Version 19.10.25017 for x86</pre> |  |

Copyright (C) Microsoft Corporation. All rights reserved.

```
usage: cl [ option... ] filename... [ /link linkoption... ]
```

```
C:\Program Files (x86)\Microsoft Visual Studio\2017\Enterprise>
```

There may be differences in the current directory or version numbers, depending on the version of Visual C++ and any updates installed. If this is similar to what you see, then you are ready to build C or C++ programs at the command line.

### **Note**

If you get an error such as "'cl' is not recognized as an internal or external command, operable program or batch file," error C1034, or error LNK1104 when you run the `cl` command, then either you are not using a developer command prompt, or something is wrong with your installation of Visual C++. You must fix this issue before you can continue.

If you can't find the developer command prompt shortcut, or if you get an error message when you enter `cl`, then your Visual C++ installation may have a problem. If you're using Visual Studio 2017, try reinstalling the **Desktop development with C++** workload in the Visual Studio installer. For details, see [Install C++ support in Visual Studio](#). Or, reinstall the [Build Tools for Visual Studio](#). Don't go on to the next section until this works. For more information about installing and troubleshooting Visual Studio, see [Install Visual Studio](#).


### **Note**

Depending on the version of Windows on the computer and the system security configuration, you might have to right-click to open the shortcut menu for the developer command prompt shortcut and then choose **Run as Administrator** to successfully build and run the program that you create by following this walkthrough.


## Create a C source file and compile it on the command line

1. In the developer command prompt window, enter `cd c:\` to change the current working directory to the root of your C: drive. Next, enter `md c:\simple` to create a directory, and then enter `cd c:\simple` to change to that directory. This is the directory that will contain your source file and the compiled program.

2. Enter **notepad simple.c** at the developer command prompt. In the Notepad alert dialog that pops up, choose **Yes** to create a new simple.c file in your working directory.
3. In Notepad, enter the following lines of code:

|   |  |
|---|--|
| C   |  Copy |
| <pre>#include &lt;stdio.h&gt;  int main() {     printf("Hello, World! This is a native C program compiled on the command line.\n");     return 0; }</pre> |  |

4. On the Notepad menu bar, choose **File, Save** to save simple.c in your working directory.
5. Switch back to the developer command prompt window. Enter **dir** at the command prompt to list the contents of the c:\simple directory. You should see the source file simple.c in the directory listing, which looks something like this:

|  |   |
|--|---|
| Output   |  Copy |
| <pre>C:\simple&gt;dir Volume in drive C has no label. Volume Serial Number is CC62-6545  Directory of C:\simple  10/02/2017  03:46 PM    &lt;DIR&gt;          . 10/02/2017  03:46 PM    &lt;DIR&gt;          .. 10/02/2017  03:36 PM                143 simple.c                1 File(s)                143 bytes                2 Dir(s)  514,900,566,016 bytes free</pre> |   |

The dates and other details will differ on your computer. If you don't see your source code file, simple.c, make sure you've changed to the c:\simple directory you created, and in Notepad, make sure that you saved your source file in this directory. Also make sure that you saved the source code with a .c file name extension, not a .txt extension.

6. To compile your program, enter **cl simple.c** at the developer command prompt.

You can see the executable program name, simple.exe, in the lines of output information that the compiler displays:

|  |  |
|--|--|
|  |  |
|--|--|

Output

 Copy

```
c:\simple>cl simple.c
Microsoft (R) C/C++ Optimizing Compiler Version 19.10.25017 for x86
Copyright (C) Microsoft Corporation. All rights reserved.

simple.c
Microsoft (R) Incremental Linker Version 14.10.25017.0
Copyright (C) Microsoft Corporation. All rights reserved.

/out:simple.exe
simple.obj
```

### Note

If you get an error such as "'cl' is not recognized as an internal or external command, operable program or batch file," error C1034, or error LNK1104, your developer command prompt is not set up correctly. For information on how to fix this issue, go back to the **Open a developer command prompt** section.

### Note

If you get a different compiler or linker error or warning, review your source code to correct any errors, then save it and run the compiler again. For information about specific errors, use the search box at the top of this page to look for the error number.

7. To run your program, enter **simple** at the command prompt.

The program displays this text and then exits:

Output

 Copy

```
Hello, World! This is a native C program compiled on the command line.
```

Congratulations, you've just compiled and run a C program by using the command-line.

## Next steps

This "Hello, World" example is about as simple as a C program can get. Real world programs have header files and more source files, link in libraries, and do useful work.

You can use the steps in this walkthrough to build your own C code instead of typing the sample code shown. You can also build many C code sample programs that you find elsewhere. To compile a program that has multiple source code files, enter them all on the command line, like this:

```
cl file1.c file2.c file3.c
```

The compiler outputs a program called file1.exe. To change the name to program1.exe, add an [/out](#) linker option:

```
cl file1.c file2.c file3.c /link /out:program1.exe
```

And to catch more programming mistakes automatically, we recommend you compile by using either the [/W3](#) or [/W4](#) warning level option:

```
cl /W4 file1.c file2.c file3.c /link /out:program1.exe
```

The compiler, cl.exe, has many more options you can apply to build, optimize, debug, and analyze your code. For a quick list, enter `cl /?` at the developer command prompt. You can also compile and link separately and apply linker options in more complex build scenarios. For more information on compiler and linker options and usage, see [C/C++ Building Reference](#).

You can use NMAKE and makefiles, or MSBuild and project files to configure and build more complex projects on the command line. For more information on using these tools, see [NMAKE Reference](#) and [MSBuild](#).

The C and C++ languages are similar, but not the same. The Visual C++ compiler uses a simple rule to determine which language to use when it compiles your code. By default, the Visual C++ compiler treats all files that end in .c as C source code, and all files that end in .cpp as C++ source code. To force the compiler to treat all files as C regardless of file name extension, use the [/Tc](#) compiler option.

The Visual C++ C compiler is generally compatible with the ISO C99 standard, but not strictly compliant. In most cases, portable C code will compile and run as expected. Visual C++ does not support most of the changes in ISO C11. Certain library functions and POSIX function names are deprecated by the Visual C++ compiler. The functions are supported, but the preferred names have changed. For more information, see [Security Features in the CRT](#) and [Compiler Warning\\_\(level 3\) C4996](#).

## See also

[Walkthrough: Creating a Standard C++ Program \(C++\)](#)  
[C Language Reference](#)

[Building C/C++ Programs](#)

[Compatibility](#)