Contents

Even though Python is an interpreted language, you may need to install Windows C++ compilers in some cases. Unlike Linux, compilers for Windows are not included by default in the OS.

For example, you will need to use them if you wish to:

- Install a non-pure Python package from sources with Pip (if there is no Wheel package provided).
- Compile a Cython or Pyrex file.

Microsoft provides official C++ compilers called *Visual C++*, you can find them bundled with *Visual Studio* or, for some versions, in standalone distributions. Some alternative compilers exist like MinGW, but incompatibilities may occur with a CPython official distribution that is built with Microsoft Visual C++.

The compiler's architecture must be the same as Python's (for example: if you use Python 64bit, you have to use an x64 compiler).

Which Microsoft Visual C++ compiler to use with a specific Python version ?

Each Python version uses a specific compiler version (e.g. *CPython 2.7* uses *Visual C++ 9.0*, *CPython 3.3* uses *Visual C++ 10.0*, etc). So, you need to install the compiler version that corresponds to your Python version :

Visual C++	CPython
14.X	3.5, 3.6, 3.7, 3.8
10.0	3.3, 3.4
9.0	2.6, 2.7, 3.0, 3.1, 3.2

Distutils notes

If the package's *setup.py* (still) uses *distutils* rather than the recommended *setuptools*, you may need extra steps:

- *distutils* only supports the very minimum of compiler setups. The sections in this guide corresponding to them explicitly mention *distutils*.
- For other setups, you need to run the compilation from the "SDK prompt" of the corresponding toolchain and set the *DISTUTILS_USE_SDK* environment variable to a non-empty value.

Compilers Installation and configuration

Compatible architectures are specified for each compiler in brackets.

Before do anything, install or upgrade the *Setuptools* Python package. It contain compatibility improvements and add automatic use of compilers:

```
pip install --upgrade setuptools
```

Microsoft Visual C++ 14.2 standalone: Build Tools for Visual Studio 2019 (x86, x64, ARM, ARM64)

This is a standalone version of *Visual C++ 14.2* compiler, you don't need to install *Visual Studio 2019*.

- Install *Microsoft Build Tools for Visual Studio 2019*.
- In Build tools, install *C++ build tools* and ensure the latest versions of *MSVCv142 - VS 2019 C++ x64/x86 build tools* and *Windows 10 SDK* are checked.
- The *setuptools* Python package version must be at least 34.4.0.

Build Tools also allows to install any previous Visual C++ 14 version (Including 2015, 2017 ones).

Microsoft Visual C++ 14.2 with Visual Studio 2019 (x86, x64, ARM, ARM64)

Visual Studio 2019 contains *Visual C++ 14.2* compiler. The *setuptools* Python package version must be at least 34.4.0.

Microsoft Visual C++ 14.1 standalone: Build Tools for Visual Studio 2017 (x86, x64, ARM, ARM64)

This is a standalone version of *Visual C++ 14.1* compiler, you don't need to install *Visual Studio 2017*.

- Install *Microsoft Build Tools for Visual Studio 2017*.
- The *setuptools* Python package version must be at least 34.4.0.

⚠ Build Tools for Visual Studio 2017 was upgraded by Microsoft to Build Tools for Visual Studio 2019. See the previous paragraph to install it.

Microsoft Visual C++ 14.1 with Visual Studio 2017 (x86, x64, ARM, ARM64)

Visual Studio 2017 contains *Visual C++ 14.1* compiler. The *setuptools* Python package version must be at least 34.4.0.

⚠ Visual Studio 2017 was upgraded by Microsoft to Visual Studio 2019. See the previous paragraph to install it.

Microsoft Visual C++ 14.0 standalone: Visual C++ Build Tools 2015 (x86, x64, ARM)

This is a standalone version of *Visual C++ 14.0* compiler, you don't need to install *Visual Studio 2015*.

- Install *Microsoft Visual C++ Build Tools 2015*. Check *Windows 8.1 SDK* and *Windows 10 SDK* options.
- The *setuptools* Python package version must be at least 24.0.

⚠ Visual C++ Build Tools 2015 was upgraded by Microsoft to Build Tools for Visual Studio 2017. See the previous paragraph to install it.

Microsoft Visual C++ 14.0 with Visual Studio 2015 (x86, x64, ARM)

Visual Studio 2015 contains *Visual C++ 14.0* compiler. *Distutils* will automatically detect the compiler and use it.

⚠ Visual Studio 2015 was upgraded by Microsoft to Visual Studio 2017. See the previous paragraph to install it.

Microsoft Visual C++ 10.0 standalone: Windows SDK 7.1 (x86, x64, ia64)

This is a standalone version of *Visual C++ 10.0* compiler, you don't need to install *Visual Studio 2010*.

- Uninstall *Microsoft Visual C++ 2010 Redistributable* if present (all versions and architectures). If present, it can cause an error on Windows SDK 7.1 installation.
- Install *Microsoft .NET Framework 4* if not present.
- Install *Microsoft Windows SDK for Windows 7 and .NET Framework 4*. Check *Windows headers and libraries*, *Visual C++ Compilers* and *Windows Native Code Development\Tools* options only.
- Install *Microsoft Visual C++ 2010 Service Pack 1 Compiler Update for the Windows SDK 7.1*. This updates the compiler to Visual C++ 10.0 SP1.
- reinstall *Microsoft Visual C++ 2010 Redistributable* (for all previously installed architectures).
- The *setuptools* Python package version must be at least 24.0.

Microsoft Visual C++ 10.0 with Visual Studio 2010 (x86, x64, ia64)

Visual Studio 2010 contains *Visual C++ 10.0* compiler. *Distutils* will automatically detect the compiler and use it. The *Express* edition of *Visual Studio 2010* only bundles a compiler for x86.

Microsoft Visual C++ 9.0 standalone: Visual C++ Compiler for Python 2.7 (x86, x64)

This is a standalone version of *Visual C++ 9.0* compiler, you don't need to install *Visual Studio 2008*.

- Install *Microsoft Visual C++ Compiler for Python 2.7*.
- The *setuptools* Python package version must be at least 6.0.

Even though this package's name refers to Python 2.7 specifically, you can use it with all Python versions that use *Visual C++ 9.0*.

This package always installs its start menu shortcuts for the installing user (i.e. an administrator) only. To get them for all users, run the installation like this: *msiexec /i <full path to .msi> ALLUSERS=1*.

Microsoft Visual C++ 9.0 standalone: Windows SDK 7.0 (x86, x64, ia64)

This is a standalone version of *Visual C++ 9.0* compiler, you don't need to install *Visual Studio 2008*.

The use of *Microsoft Visual C++ Compiler for Python 2.7* is recommended (If you don't need to compile for ia64). See the previous paragraph to install it.

- Install *Microsoft .NET Framework 3.5 SP1* if not present.
- Install *Microsoft Windows SDK for Windows 7 and .NET Framework 3.5 SP1*. Check *Windows headers and libraries*, *Visual C++ Compilers* and *Win32 Development Tools* options only.
- The *setuptools* Python package version must be at least 24.0.

Microsoft Visual C++ 9.0 standalone: Windows SDK 6.1 (x86, x64, ia64)

This is a standalone version of *Visual C++ 9.0* compiler, you don't need to install *Visual Studio 2008*.

Windows SDK 6.1 was upgraded by Microsoft to Windows SDK 7.0. See the previous paragraph to install it.

- Install *Microsoft .NET Framework 3.5 SP1* if not present.
- Install *Microsoft Windows SDK for Windows Server 2008 and .NET Framework 3.5*. Check *Windows headers and libraries*, *Visual C++ Compilers* and *Win32 Development Tools* options only.
- The *setuptools* Python package version must be at least 24.0.

Microsoft Visual C++ 9.0 with Visual Studio 2008 (x86, x64, ia64)

Visual Studio 2008 contains *Visual C++ 9.0* compiler. *Distutils* will automatically detect the compiler and use it. The *Express* edition of *Visual Studio 2008* only bundles a compiler for x86.

GCC - MinGW-w64 (x86, x64)

MinGW-w64 is an alternative C/C++ compiler that works with all Python versions up to 3.4.

- Install *Win-builds* into *C:\MinGW_w64*.

- Open *Win-builds*, switch to *install* at least *binutils*, *gcc*, *gcc-g++*, *getext*, *mingw-w64*, *win-iconv*, *winpthreads*, *zlib*, and click *Process*.
- Add *C:\MinGW_w64\bin* to the *PATH* environment variable.
- Create a *distutils.cfg* file with the following contents in the folder *\Lib\distutils* in Python install directory :

Toggle line numbers

```
1 [build]
2 compiler=mingw32
3
4 [build_ext]
5 compiler=mingw32
```

GCC - MinGW (x86)

MinGW is an alternative C/C++ compiler that works with all Python versions up to 3.4.

- Install *Minimalist GNU For Windows* into *C:\MinGW*.
- Open *MinGW Installation Manager*, check *mingw32-base* and *mingw32-gcc-g++*, and *Apply Changes* in the *Installation* menu.
- Add *C:\MinGW\bin* to the *PATH* environment variable.
- Create a *distutils.cfg* file with the following contents in the folder *\Lib\distutils* in Python install directory :

Toggle line numbers

```
1 [build]
2 compiler=mingw32
3
4 [build_ext]
5 compiler=mingw32
```