Compiling Python extensions on Windows

Sun 21 December 2014, blog.ionelmc.ro

For Python 2.7

For Python 2.7 you need to get Microsoft Visual C++ Compiler for Python 2.7. It's a special package made by Microsoft that has all the stuff. It is supported since setuptools *6.0* [1].

Unfortunately the latest virtualenv, *1.11.6* as of now, still bundles setuptools *3.6*. This means that if you try to run `python setup.py build_ext` in an virtualenv it will fail, because setuptools can't detect the compiler. The solution is to force upgrade setuptools, example: `pip install "setuptools>=6.0"`.

If you're using tox then just add it to your `deps`. Example:

```
[testenv]
deps =
    setuptools>=6.0
```

This seems to work fine for 64bit extensions.

Note

Probably works for Python 3.3 too.

For Python 3.4

This one gave me a headache. I've tried to follow this guide but had some problems with getting 64bit extensions to compile. In order to get it to work you need to jump through these hoops:

1. Install Visual C++ 2010 Express (download).

2. **Before** installing the *Windows SDK v7.1* (these gave me a bad time):

    - Do **not** install Microsoft Visual Studio 2010 Service Pack 1 yet. If you did then you have to reinstall everything. Uninstalling the Service Pack removes components so you have to reinstall *Visual C++ 2010 Express* again.
    - Remove all the *Microsoft Visual C++ 2010 Redistributable* packages from *Control Panel\Programs and Features*.

    If you don't do those then the install is going to fail with an obscure "*Fatal error during installation*" error.

3. Install Windows SDK for Visual Studio 2010 (v7.1) (download). This is required for 64bit extensions.

    ⚠ On Windows 10 the web installer gets confused:

    > Some Windows SDK components require the RTM .NET Framework 4. Setup detected a pre-release version of the .NET Framework 4.

    You certainly don't need any of the .NET stuff so you can work around by getting the offline version of the SDK (ISO) (download).

    - Make sure you download `GRMSDKX_EN_DVD.iso` (`X` means 64bit).

  - Windows has builtin mounting for ISOs. Just mount the ISO and run `Setup\SDKSetup.exe` instead of `setup.exe`.

4. Create a `vcvars64.bat` file in `C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC\bin\amd64` that contains [2]:

```
CALL "C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin\SetEnv.cmd" /x64
```

   If you don't do this you're going to get a mind-boggling error like this:

```
Installing collected packages: whatever
  Running setup.py develop for whatever
    building 'whatever.cext' extension

Traceback (most recent call last):
  File "<string>", line 1, in <module>
  File "C:/whatever\setup.py", line 112, in <module>
    for root, _, _ in os.walk("src")
  File "c:\python34\Lib\distutils\core.py", line 148, in setup
    dist.run_commands()
  File "c:\python34\Lib\distutils\dist.py", line 955, in run_commands
    self.run_command(cmd)
  File "c:\python34\Lib\distutils\dist.py", line 974, in run_command
    cmd_obj.run()
  File "C:\whatever\.tox\3.4\lib\site-
packages\setuptools\command\develop.py", line 32, in run
    self.install_for_development()
  File "C:\whatever\.tox\3.4\lib\site-
packages\setuptools\command\develop.py", line 117, in
install_for_development
    self.run_command('build_ext')
  File "c:\python34\Lib\distutils\cmd.py", line 313, in run_command
    self.distribution.run_command(command)
  File "c:\python34\Lib\distutils\dist.py", line 974, in run_command
    cmd_obj.run()
  File "C:/whatever\setup.py", line 32, in run
    build_ext.run(self)
  File "C:\whatever\.tox\3.4\lib\site-
packages\setuptools\command\build_ext.py", line 54, in run
    _build_ext.run(self)
  File "c:\python34\Lib\distutils\command\build_ext.py", line 339, in run
    self.build_extensions()
  File "c:\python34\Lib\distutils\command\build_ext.py", line 448, in
build_extensions
    self.build_extension(ext)
  File "C:/whatever\setup.py", line 39, in build_extension
    build_ext.build_extension(self, ext)
  File "C:\whatever\.tox\3.4\lib\site-
packages\setuptools\command\build_ext.py", line 187, in build_extension
    _build_ext.build_extension(self, ext)
  File "c:\python34\Lib\distutils\command\build_ext.py", line 503, in
build_extension
```

```
    depends=ext.depends)
  File "c:\python34\Lib\distutils\msvc9compiler.py", line 460, in compile
    self.initialize()
  File "c:\python34\Lib\distutils\msvc9compiler.py", line 371, in
initialize
    vc_env = query_vcvarsall(VERSION, plat_spec)
  File "C:\whatever\.tox\3.4\lib\site-
packages\setuptools\msvc9_support.py", line 52, in query_vcvarsall
    return unpatched['query_vcvarsall'](version, *args, **kwargs)
  File "c:\python34\Lib\distutils\msvc9compiler.py", line 287, in
query_vcvarsall
    raise ValueError(str(list(result.keys())))
ValueError: ['path']
Complete output from command C:\whatever\.tox\3.4\Scripts\python.exe -c
"import setuptools, tokenize; __file__='C:/whatever\\setup.py';
exec(compile(getattr(tokenize, 'open', open)
(__file__).read().replace('\r\n', '\n'), __file__, 'exec'))" develop --no-
deps:
```

`msvc9_support.py` will run `vcvarsall.bat amd64`:

```
c:\Program Files (x86)\Microsoft Visual Studio 10.0\VC>vcvarsall.bat amd64
The specified configuration type is missing.  The tools for the
configuration might not be installed.
```

Basically that is caused by `vcvarsall.bat` not being able to run `vcvar64.bat` because, surprise, the Windows SDK is missing that file.

5. Now everything should work, go and try `py -3 setup.py clean --all build_ext --force`.

6. Install Microsoft Visual Studio 2010 Service Pack 1 (download). This is optional, however, if you do this you also have to do the following too [3]:

7. Install Microsoft Visual C++ 2010 Service Pack 1 Compiler Update for the Windows SDK 7.1 (download).

8. Remove all those SQL Server packages that you don't need from *Control Panel\Programs and Features*.

Alternative: Victor Stinner argues that just installing the Windows SDKs (*v7.0* for

Python 2.7, *v7.1* for Python 3.3 and 3.4) is enough if you prepare the environment (inconvenient in my opinion):

```
setenv /x64 /release
set MSSDK=1
set DISTUTILS_USE_SDK=1
```

See his notes. You still need to install those fiddly SDKs :-)

For Python 3.5

It seems *Visual Studio 2015 Community Edition* just works without any fuss. Get it here. It's quite big and doesn't let you skip the irrelevant SQL Server and .NET tools so I had to manually remove them from *Control Panel\Programs and Features*.

[1] Support added in https://bitbucket.org/pypa/setuptools/issue/258

[2] See: http://stackoverflow.com/a/26513378

[3] Installing the *Microsoft Visual Studio 2010 Service Pack 1* removes some components from the SDK. The solution is to also install the *Microsoft Visual C++ 2010 Service Pack 1 Compiler Update for the Windows SDK 7.1*. See: http://support.microsoft.com/kb/2519277

This entry was tagged as python windows

Subscribe or Search