Create a single executable from a Python project

I want to create a single executable from my Python project. A user should be able to download and run it without needing Python installed. If I were just distributing a package, I could use pip, wheel, and PyPI to build and distribute it, but this requires that the user has Python and knows how to install packages. What can I use to build a self-contained executable from a Python project?

python compilation exe packaging software-distribution

edited Jan 18 at 14:21
davidism
47.5k 10 92 122

asked Aug 21 '12 at 16:47
ShadowFlame
597 2 9 24

I used py2exe about 8 months ago with success. It hasn't changed because it hasn't needed to, assuming you're still writing python 2.x, not python 3.x – Endophage Aug 21 '12 at 16:49

I am still writing in 2.x(2.7.2 to be exact), but might see the need to transfer to 3.x(if modules and such are compatible) and would need a solution then too. – ShadowFlame Aug 21 '12 at 16:51

1 py2exe is dead, cx_Freeze is great. – Oleh Prypin Aug 21 '12 at 16:52

Note that packaging will complicate problem diagnostics greatly. Not only you (or `pdb`) don't have easy access to the files being executed, `py2exe` (dunno about the others) doesn't include `.py` files so you won't see source lines in stack traces. – ivan_pozdeev Jan 16 '15 at 17:08

2 Answers

actively maintained as of my last edit (September 2017).

I've also included links to their respective websites, repos, and PyPi listings in case you want to check for yourself on how frequently they've been updated.

Also, unless otherwise noted, all programs listed below will produce an exe specifically for the operating system it's running in. So for example, running Pyinstaller in Windows will produce a Windows exe, but running Pyinstaller in Linux will produce a Linux exe. If you want to produce an exe for multiple operating systems, you will have to look into using virtual machines or look into using something like Wine.

The following programs all work similarly -- they bundle together Python and your program, effectively combining them to produce an executable.

- **PyInstaller:**
  - **Links**: Website || Repo || PyPi
  - **Supports**: Python 2.7 and Python 3.3 - 3.6 on Windows, Mac, and Linux.
- **cx_Freeze:**
  - **Links:** Website || Repo || PyPi
  - **Supports:** Python 2.7 and 3.0 (?) - 3.6 on Windows, Mac, and Linux.
  - **Note:** The last version of cx_Freeze to support Python 2.6 appears to be 4.3.4, released in December 2016.
- **pyapp:**
  - **Links:** Website || Repo || PyPi
  - **Supports:** Python 2.7 and 3.3 - 3.4 (?) on Mac only.
  - **Note:** As of my last edit, I've been unable to conclusively determine if pyapp supports Python 3.5 - 3.6.
- **py2exe**:
  - **Links:** Website || Repo || PyPi
  - **Supports:** Windows only. Version 0.6.9 supports Python 2.4 - 2.7, and version 0.9.2.2

was last updated on October 2014. The version on PyPI supports Python 3 only.

- **bbfreeze**:
  - **Links:** Website || Repo || PyPi
  - **Supports:** Python 2.4 - 2.7 but not Python 3 on Windows and Linux.
  - **Other notes:** Currently does not have an active maintainer, and was last updated in 2014.

You can find a slightly (?) more in-depth comparison at The Hitchhiker's Guide to Python: Freezing your code.

Of course, that's not the only way of doing things:

- **pynsist:**
  - **Links:** Website || Repo || PyPi
  - **Supports:** All Python versions? Note -- will create **Windows installers** only.
  - **Other notes:** Pynsist seems to support bundling together any arbitrary Python version with your application. However, the tool itself requires Python 2.7 or 3.3+ to run.

    Pynsist will create a Windows installer for your program which will directly install Python on the user's computer instead of bundling it with your code and create shortcuts that link to your Python script. Although this program produces only Windows installers, it appears that you can still run Pynsist on Mac and Linux computers.
- **Nuitka:**
  - **Links:** Website || Repo (Github mirror) || PyPi
  - **Supports:** Python 2.6 - 2.7 and Python 3.2 - 3.6 on Windows, Mac, and Linux.
  - **Other notes:** Nuitka will literally compile your Python code and produce an exe (as opposed to the other projects, which simply include Python) to try and speed up your code. As a side effect, you'll also get a handy exe you can distribute. Note that you need to have a C++ compiler available on your system.

- **Supports:** Python 2.6 - 2.7 and Python 3.2 - 3.6 (?) on Windows, Mac, and Linux.
- **Other notes:** Cython is similar to Nuitka in that it is a Python compiler. However, instead of directly compiling your code, it'll compile it to C. You can then take that C code and turn your code into an exe. You'll need to have a C compiler available on your system.

My personal preference is to use PyInstaller since it was the easiest for me to get up and running, was designed to work nicely with various popular libraries such as numpy or pygame, and has great compatibility with various OSes and Python versions.

However, I've also successfully built various exes using cx_Freeze and py2exe without too much difficulty, so you should also definitely consider checking those out.

I haven't yet had a chance to to try pynist, Nuitka, or Cython extensively, but they seem like pretty interesting and innovative solutions. If you run into trouble using the first group of programs, it might be worthwhile to try one of these three. Since they work fundamentally differently then the Pyinstaller/cx_freeze-style programs, they might succeed in those odd edge cases where the first group fails.

In particular, I think pynist is a good way of sidestepping the entire issue of distributing your code altogether: Macs and Linux already have native support for Python, and just installing Python on Windows might genuinely be the cleanest solution. (The downside is now that you need to worry about targeting multiple versions of Python + installing libraries).

Nuitka and Cython (in my limited experience) seem to work fairly well. Again, I haven't tested them extensively myself, and so my main observation is that they seem to take much longer to produce an exe then the "freeze" style programs do.

As a final note, if you want to support Linux only, you may want to look into creating a distro package for your package, instead of turning it into an executable. Listing tools that can help you do this is beyond the scope of this post, but here's some reading material if you decide to explore this route.

edited Sep 22 at 17:50 answered Aug 21 '12 at 16:59

Note that you also need to include inside your bundle the Microsoft C++ Redistributables (aka `msvcrxx.dll` , where `xx` is a version number) to make the .exe run (or ask users to install it by themselves). This is necessary because Python is built on Windows using Visual Studio. – gaborous Oct 21 '15 at 18:40

So it looks like none of the python IDE's (pycharm or pydev - eclipse pluggin) have yet to incorporate the compile-to-single-executable functionality into the IDE itself. Is that correct? :-) – Nelda.techspiress Oct 26 '16 at 16:32

If you care about keeping safe your source code, probably Cython is the best choice because it compiles C code. Those programs that use *freezing* technique generally can be easily decompiled. – Martín De la Fuente Apr 30 at 19:06

pyinstaller is still under active development. You can see the latest changes on GitHub.

It has support for all three major platforms:

- Windows (32-bit and 64-bit)
- Linux (32-bit and 64-bit)
- Mac OS X (32-bit and 64-bit)

and it supports Python versions 2.6 and 2.7. It does not support Python 3, but there is an experimental Python 3 branch.

Update

As of version 3.2.1 it supports Python 2.7, 3.3-3.5

edited Apr 7 at 0:55 answered Aug 21 '12 at 16:58

protected by davidism Oct 21 '15 at 18:52

Thank you for your interest in this question. Because it has attracted low-quality or spam answers that had to be removed, posting an answer now requires 10 reputation on this site (the association bonus does not count).

Would you like to answer one of these unanswered questions instead?