

Forum:[Hack Forums](#)

Thread: [How to Convert Python Source code to exe effectively !??](#)

SIM-4791248 - June 5, 2021, 9:44 pm

i tried all public tools once again [auto py to exe - py2exe...Etc] Python 3.9 ?

any clear or solid solutions yet?? my program include tkinter GUI wich never run after compiling ???

Thanks again

truco - June 6, 2021, 5:52 am

what kind of error are you getting? you are probably not including something necessary to run the program, i mostly go for pyarmor or i'm converting the source to .pyd

Doctor. - June 6, 2021, 9:50 am

Wish I could help you bro but idk what that shit is xD

Sajuuk - June 6, 2021, 11:36 am

I'm assuming you're needing to pack Python code into a Windows PE for compact portability. The existing options are popular for a reason: they're essentially the only ones out there. As an additional step, you could pack the WinPE with something like UPX, but that is extremely common among the purpose. Python isn't meant to be shipped/exchanged as near-unreadable code. The readability aspect of Python was taken into account as a major focus for a reason, and (as far as I'm concerned; too lazy to e-mail and ask the developers of the language) that's not meant to change from source to any other form.

If you're wanting to pack Python code into a WinPE with a tiny Python interpreter (via aforementioned packers) in some attempt to protect the source code, you've no use for Python and would be better off using some language more fit for the task. I.e. a language which compiles to native. That is the straightest-forward process in protecting source code. Compiling code to processor/memory instructions makes it near impossible to tell what a program is doing (specifically, at least). Both C and C++ work great for this. Rust is a close competitor, albeit more "memory safe" (which you can do yourself when programming C/C++).

tl;dr (in either case)

<https://www.cprogramming.com/tutorial/c-tutorial.html>

<https://jmeubank.github.io/tdm-gcc/> (Windows)

<https://gcc.gnu.org/> (Mac/Linux)

If neither of my assumptions are correct, let alone close to what purpose you have in doing this, let me know. Yes, post here. I've zero clue why so many people ask to PM me for questions of any sort. Post it here in your thread. That way other/future board members can find the answer(s) for the same question.

TheCompany - June 7, 2021, 12:46 pm

(Jun 5, 2021 09:44 PM)MrRob0otV2 Wrote: [CITE](#) i tried all public tools once again [auto py to exe - py2exe...Etc] Python 3.9 ?

any clear or solid solutions yet?? my program include tkinter GUI wich never run after compiling ???

Thanks again

<https://medium.com/lifeandtech/executable-gui-with-python-fc79562a5558> ?

TOG6 - June 8, 2021, 3:28 am

try pip install pyinstaller and pip install auto-py-to-exe

then open cmd agn and type auto-py-to-exe, from there you'll 100% know what to do!

good luck!

Doomslayer - June 8, 2021, 6:58 am

Python is an interpreted language, not a compiled language. This means that instead of being compiled to machine code which is understood by the computer, python code is passed through a separate program known as an interpreter. This means that if you want a standalone windows executable, you will have to pack the entire python interpreter along with any external modules your program uses (in this case tkinter). This will never be efficient and your executable will end up being huge.

Out of all the tools that do this for you, I have had the most success with pyinstaller. You may want to enable the no upx flag because that fixed some problems I was having with it, but keep in mind python really wasn't made for portable desktop applications.

x_x - June 8, 2021, 10:57 am

Pyinstaller but you'll have to include every library otherwise it won't work.

?BADMAN? - June 9, 2021, 4:22 pm

You should check nuitka compiler.

Doc - June 11, 2021, 3:18 pm

Might wanna consider learning to use Visual Studio and learning C# if you're wanting to create compiled executable programs. Just a tip.

Sajuuk - June 11, 2021, 7:42 pm

(Jun 11, 2021 03:18 PM)The Doc Wrote: [\[CITE\]](#) Might wanna consider learning to use Visual Studio and learning C# if you're wanting to create compiled executable programs. Just a tip.

tsuki - June 13, 2021, 7:25 am

(Jun 8, 2021 06:58 AM)Doomslayer Wrote: [\[CITE\]](#) Python is an interpreted language, not a compiled language.

Languages are not interpreted or compiled. Implementations of them are.

AeroBlue - June 14, 2021, 2:29 pm

So I went through this issue a while back, and while it's a different version the general method should work. I would try using [PyInstaller](#) as it's the best option I've found and has a variety of different ways of compiling. Sometimes libraries don't do well but PyInstaller usually does a good job of resolving things.

Lindros - June 16, 2021, 12:50 pm

(Jun 13, 2021 07:25 AM)tsuki Wrote: [\[CITE\]](#) Languages are not interpreted or compiled. Implementations of them are.

If the implementation of a language is using a compiler, then it is referred to as a compiled language. If the implementation of a language uses an interpreter, it's referred to commonly as an interpreted language.

While I see why you're technically correct, there was nothing really wrong with Doomslayer's explanation.

Goodies - June 18, 2021, 10:46 am

(Jun 16, 2021 12:50 PM)Lindros Wrote: [\[CITE\]](#) If the implementation of a language is using a compiler, then it is referred to as a compiled language. If the implementation of a language uses an interpreter, it's referred to commonly as an interpreted language.

While I see why you're technically correct, there was nothing really wrong with Doomslayer's explanation.

I think what he was getting at is that there's no reason you can't compile Python.

You are right, the reference implementation of Python (the one from python.org) is interpreted, but of course languages are simply just a collection of rules and syntax. Implementations like Cython are an ecosystem for extending the existing Python language (in both syntax and functionality) as well as providing a transpiler (from Python to C) which allows you to compile the source code into a static executable.

kray - June 18, 2021, 10:11 pm

(Jun 18, 2021 10:46 AM)Goodies Wrote: [\[CITE\]](#) I think what he was getting at is that there's no reason you can't compile Python.

You are right, the reference implementation of Python (the one from python.org) is interpreted, but of course languages are simply just a collection of rules and syntax. Implementations like Cython are an ecosystem for extending the existing Python language (in both syntax and functionality) as well as providing a transpiler (from Python to C) which allows you to compile the source code into a static executable.

But have you heard of Brython? It's a great implementation of the Python interpreter that bring novel usage to the `<=` operator that nobody ever wanted.

Goodies - June 19, 2021, 3:13 am

(Jun 18, 2021 10:11 PM)kray Wrote: [\[CITE\]](#) But have you heard of Brython? It's a great implementation of the Python interpreter that bring novel usage to the `<=` operator that nobody ever wanted.

I actually have! I toyed with it once a few years ago, tried learning the API and browser module, then like 6 months ago or so I thought about it again in passing and found out it fully supports python 3 syntax including async/await. It's neat, but I haven't really played with it again. I know typescript and Vue.js well enough to cover all of my (scant) webdev needs, so unless a similar reactive framework could be created in Brython, it seems like it would be a lot more effort for the design. I hate tkinter for the same reason. I don't want to programmatically design shit... I want to code the functionality of the application and just include components that someone else has already designed xD

It's worth noting that RustPython can compile to WASM which is beyond neat.