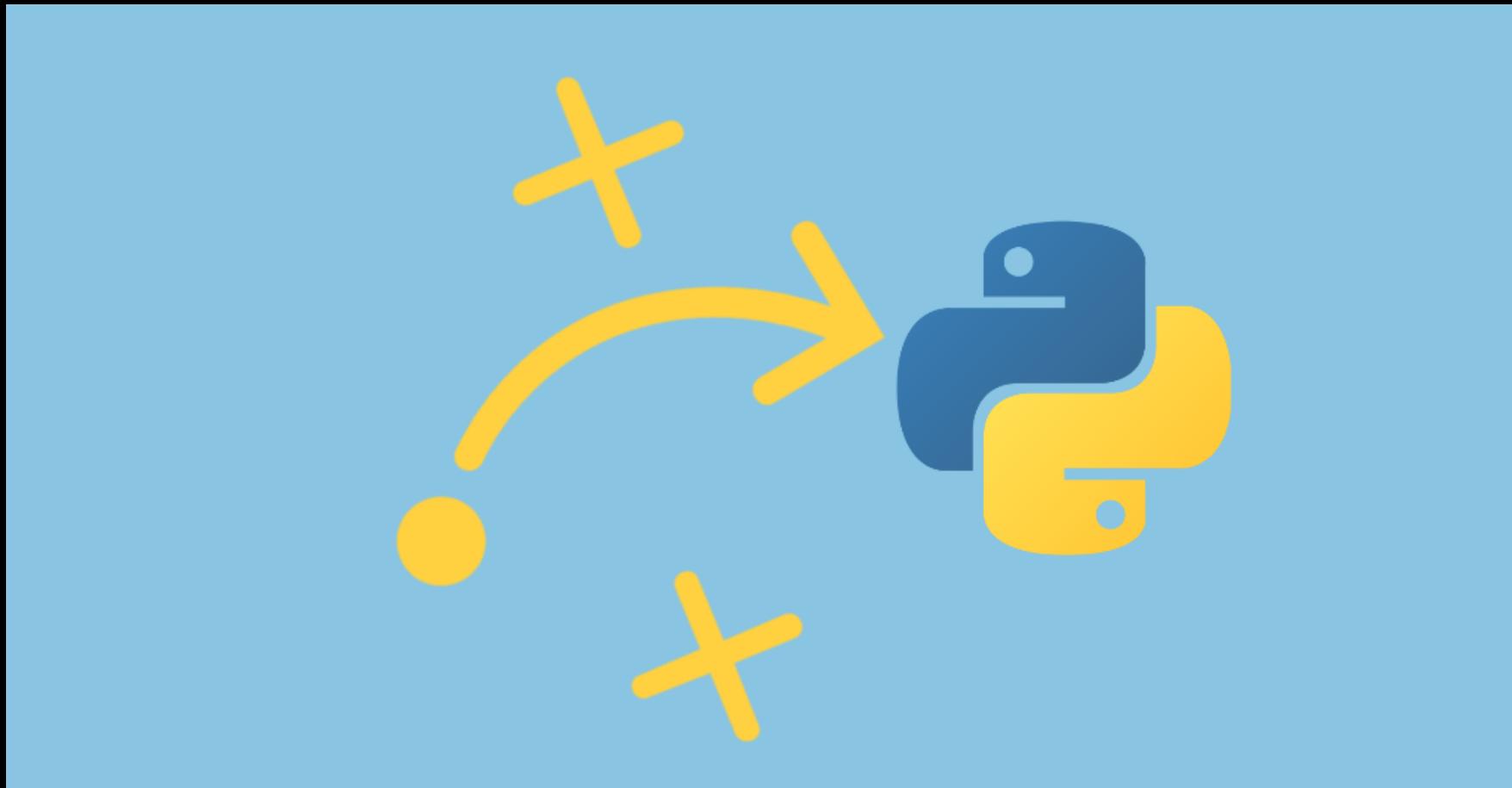


When **NOT** to Python



When **NOT** to Python



Python vs. . . .

Python	
Open source	Yes
Compiled	Typically no
Owned by a company	No
Base class library	Yes
Web app capabilities	Very strong
Database capabilities	Very strong
Mobile app capabilities	Poor
Desktop app capabilities	Moderate
Stack overflow Rank	1
TIOBE rank	3
Price	Free
General purpose language	Yes
Scientific computing level	Very strong

- Mobile



Talk Python Training Apps

Mobile apps - [Talk Python Train X] +

← → ⌛ ⌂ https://training.talkpython.fm/apps

TalkPython['Training'] Courses Apps Pricing Business Account Log out ⚙

Three devs are better than one 01:42

Julian's setup 02:43

Bob's setup 03:58

Michael's tool's and setup 02:46

PyBites code challenge platform 03:10

Talk Python Training Applications

Watch our courses on your device, on the go, or even offline! Our mobile apps are the best way to experience Talk Python courses on Android and iOS.

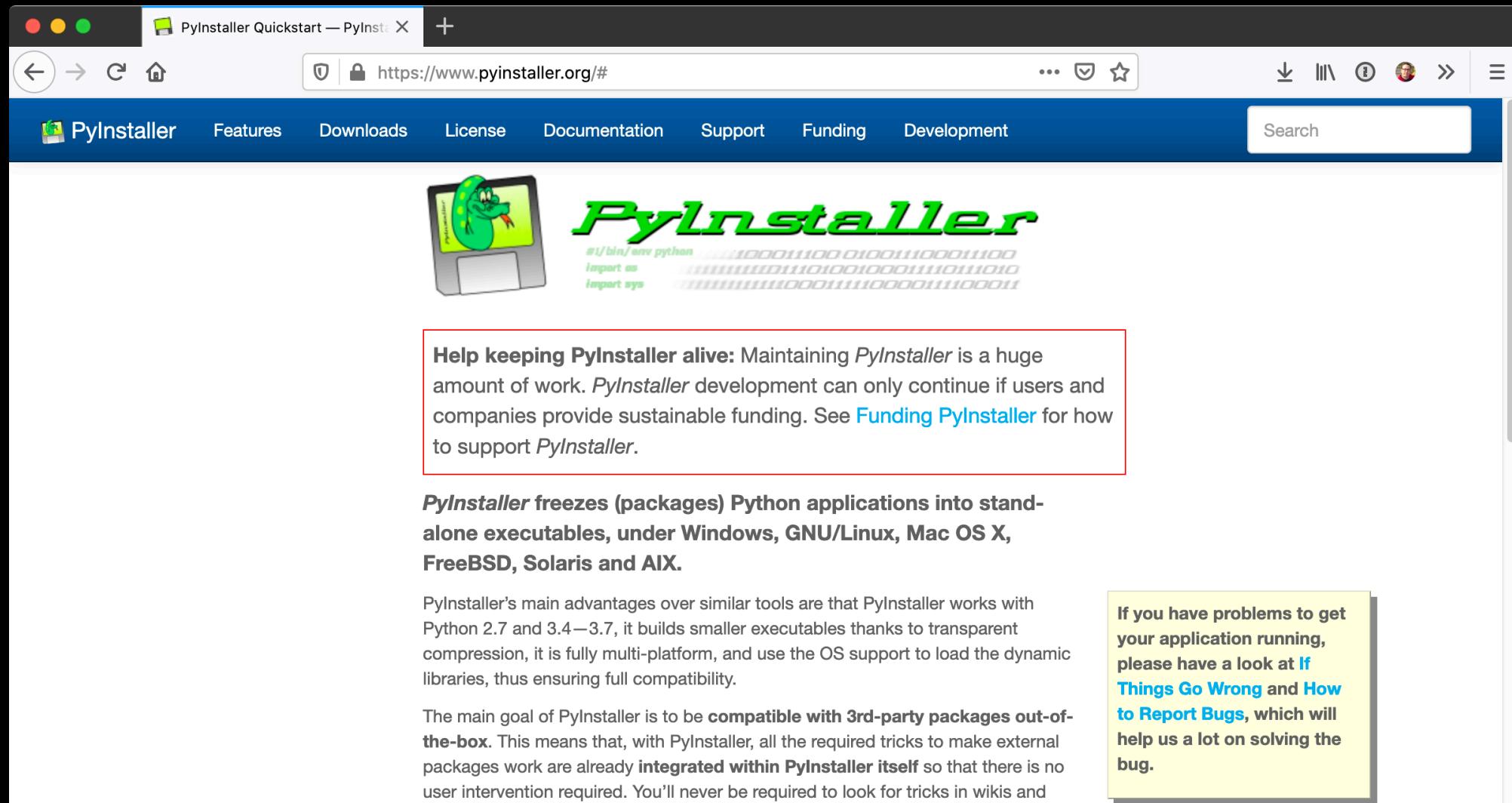
GET IT ON Google Play

Download on the App Store

Shipping consumer apps (probably)



PyInstaller



The screenshot shows a web browser displaying the PyInstaller homepage. The page has a dark blue header with the PyInstaller logo and navigation links for Features, Downloads, License, Documentation, Support, Funding, and Development. A search bar is also present. The main content area features a cartoon snake on a floppy disk icon next to the word "PyInstaller" in green. Below this is some Python code. A red-bordered box contains a message about funding. Another section describes PyInstaller's functionality. A sidebar on the right provides troubleshooting information.

PyInstaller Quickstart — PyInst... X +

https://www.pyinstaller.org/#

PyInstaller Features Downloads License Documentation Support Funding Development Search

 **PyInstaller**
#!/bin/python
import os
import sys

Help keeping PyInstaller alive: Maintaining PyInstaller is a huge amount of work. PyInstaller development can only continue if users and companies provide sustainable funding. See [Funding PyInstaller](#) for how to support PyInstaller.

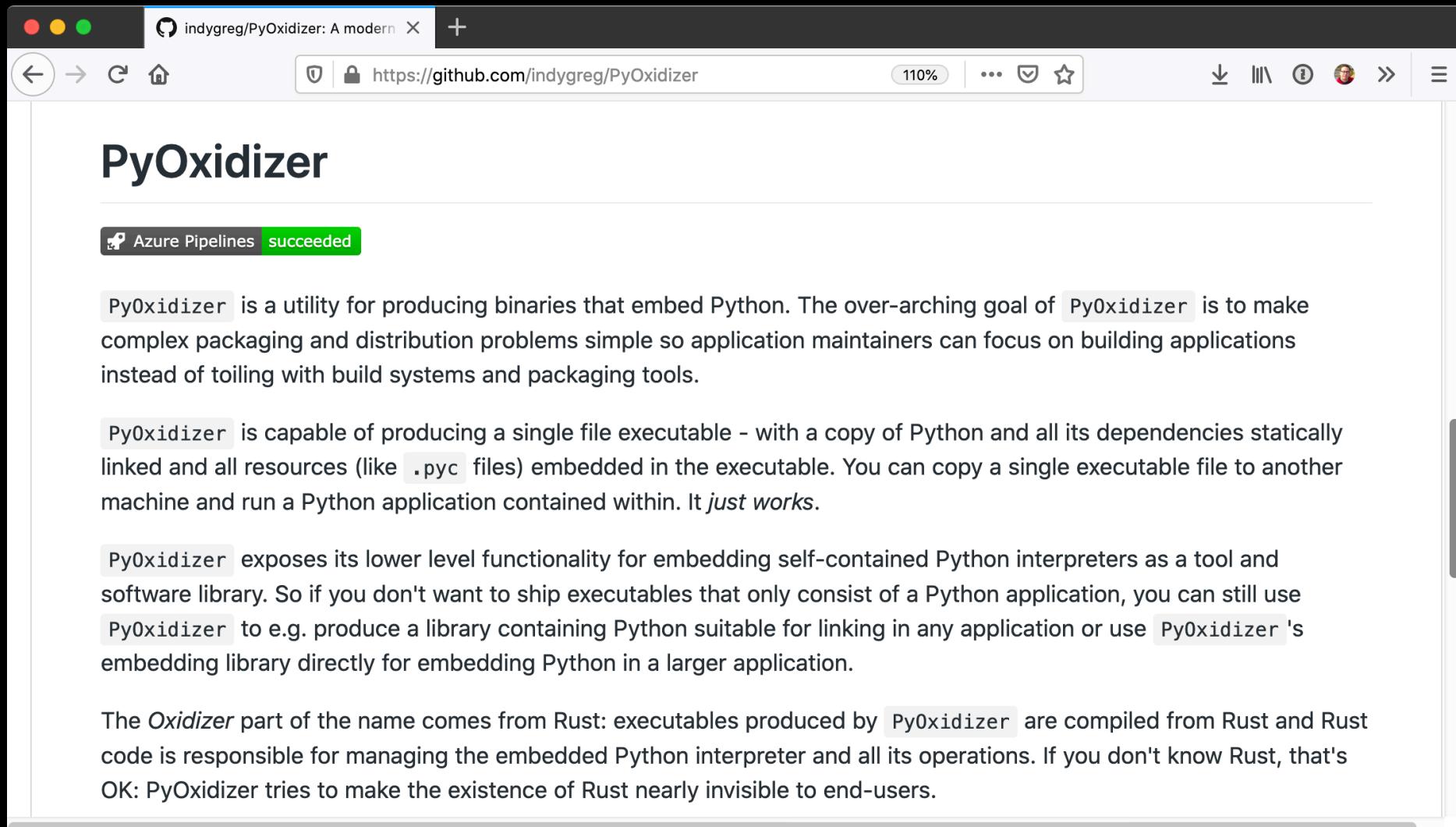
PyInstaller freezes (packages) Python applications into stand-alone executables, under Windows, GNU/Linux, Mac OS X, FreeBSD, Solaris and AIX.

PyInstaller's main advantages over similar tools are that PyInstaller works with Python 2.7 and 3.4–3.7, it builds smaller executables thanks to transparent compression, it is fully multi-platform, and use the OS support to load the dynamic libraries, thus ensuring full compatibility.

The main goal of PyInstaller is to be **compatible with 3rd-party packages out-of-the-box**. This means that, with PyInstaller, all the required tricks to make external packages work are already **integrated within PyInstaller itself** so that there is no user intervention required. You'll never be required to look for tricks in wikis and

If you have problems to get your application running, please have a look at [If Things Go Wrong](#) and [How to Report Bugs](#), which will help us a lot on solving the bug.

PyOxidizer



The screenshot shows a web browser window with the URL <https://github.com/indygreg/PyOxidizer>. The page title is "indygreg/PyOxidizer: A modern". The main content area features a large heading "PyOxidizer". Below it is a green button labeled "Azure Pipelines succeeded". The text explains what PyOxidizer is: "PyOxidizer is a utility for producing binaries that embed Python. The over-arching goal of PyOxidizer is to make complex packaging and distribution problems simple so application maintainers can focus on building applications instead of toiling with build systems and packaging tools." It also describes its capabilities: "PyOxidizer is capable of producing a single file executable - with a copy of Python and all its dependencies statically linked and all resources (like .pyc files) embedded in the executable. You can copy a single executable file to another machine and run a Python application contained within. It *just works*." Additionally, it mentions its lower-level functionality: "PyOxidizer exposes its lower level functionality for embedding self-contained Python interpreters as a tool and software library. So if you don't want to ship executables that only consist of a Python application, you can still use PyOxidizer to e.g. produce a library containing Python suitable for linking in any application or use PyOxidizer's embedding library directly for embedding Python in a larger application." Finally, it discusses its Rust-based nature: "The Oxidizer part of the name comes from Rust: executables produced by PyOxidizer are compiled from Rust and Rust code is responsible for managing the embedded Python interpreter and all its operations. If you don't know Rust, that's OK: PyOxidizer tries to make the existence of Rust nearly invisible to end-users."