Python Pros and Cons

Jason Olson
SOFTWARE ENGINEER

@jolson88 www.jolson88.com

Pros and Cons

Pro: Comprehensive Standard Library

Pro: Community-driven

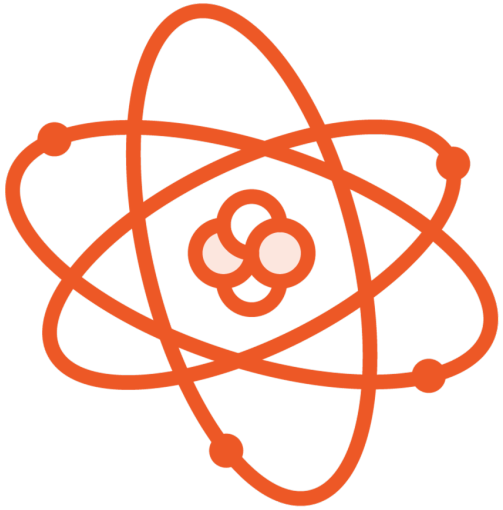
Pro: 3rd Party Libraries

Pro: 3rd Party Tools

Python Cons

Pro: Comprehensive Standard Library

Standard Library Philosophies



Minimal Standard Library

Pay only for what you use

Comprehensive Standard Library

You can do it all

A Comprehensive Standard Library

Collections	**File I/O**	**Dates and times**
Compression	**User interfaces**	**Much more...**

docs.python.org/3/library/index.html

Previous topic
10. Full Grammar specification

Next topic
Introduction

This Page
Report a Bug
Show Source

The Python Standard Library

While The Python Language Reference describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python's standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming. Some of these modules are explicitly designed to encourage and enhance the portability of Python programs by abstracting away platform-specifics into platform-neutral APIs.

The Python installers for the Windows platform usually include the entire standard library and often also include many additional components. For Unix-like operating systems Python is normally provided as a collection of packages, so it may be necessary to use the packaging tools provided with the operating system to obtain some or all of the optional components.

In addition to the standard library, there is a growing collection of several thousand components (from individual programs and modules to packages and entire application development frameworks), available from the Python Package Index.

Pro: Community-driven

PEP 0 -- Index of Python Enhancement Proposals (PEPs)

PEP:	0
Title:	Index of Python Enhancement Proposals (PEPs)
Last-Modified:	2021-02-26
Author:	python-dev <python-dev at python.org>
Status:	Active
Type:	Informational
Created:	13-Jul-2000

Contents

www.python.org/dev/peps/pep-0020/

Abstract

Long time Pythoneer Tim Peters succinctly channels the BDFL's guiding principles for Python's design into 20 aphorisms, only 19 of which have been written down.

The Zen of Python

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

docs.python.org

Python 3.9.2 documentation

Welcome! This is the documentation for Python 3.9.2.

Parts of the documentation:

- **What's new in Python 3.9?**
  or all "What's new" documents since 2.0
- **Tutorial**
  start here
- **Library Reference**
  keep this under your pillow
- **Language Reference**
  describes syntax and language elements
- **Python Setup and Usage**
  how to use Python on different platforms
- **Python HOWTOs**
  in-depth documents on specific topics
- **Installing Python Modules**
  installing from the Python Package Index & other sources
- **Distributing Python Modules**
  publishing modules for installation by others
- **Extending and Embedding**
  tutorial for C/C++ programmers
- **Python/C API**
  reference for C/C++ programmers
- **FAQs**
  frequently asked questions (with answers!)

Indices and tables:

- **Global Module Index**
  quick access to all modules
- **General Index**
  all functions, classes, terms
- **Glossary**
  the most important terms explained
- **Search page**
  search this documentation
- **Complete Table of Contents**
  lists all sections and subsections

Download

Download these documents

Docs by version

Python 3.10 (in development)
Python 3.9 (stable)
Python 3.8 (stable)
Python 3.7 (security-fixes)
Python 3.6 (security-fixes)
Python 3.5 (EOL)
Python 2.7 (EOL)
All versions

Other resources

PEP Index
Beginner's Guide
Book List
Audio/Visual Talks
Python Developer's Guide

Python Release Notes

What's New In Python 3.9

Release:	3.9.2
Date:	February 25, 2021
Editor:	Łukasz Langa

This article explains the new features in Python 3.9, compared to 3.8. Python 3.9 was released on October 5th, 2020.

For full details, see the changelog.

See also: **PEP 596** - Python 3.9 Release Schedule

Summary – Release highlights ¶

New syntax features:

- **PEP 584**, union operators added to `dict`;
- **PEP 585**, type hinting generics in standard collections;
- **PEP 614**, relaxed grammar restrictions on decorators.

New built-in features:

- **PEP 616**, string methods to remove prefixes and suffixes.

New features in the standard library:

- **PEP 593**, flexible function and variable annotations;
- `os.pidfd_open()` added that allows process management without races and signals.

Interpreter improvements:

- **PEP 573**, fast access to module state from methods of C extension types;
- **PEP 617**, CPython now uses a new parser based on PEG;

Type:	Standards Track
Created:	01-Mar-2019
Python-Version:	3.9
Post-History:	01-Mar-2019, 16-Oct-2019, 02-Dec-2019, 04-Feb-2020, 17-Feb-2020
Resolution:	https://mail.python.org/archives/list/python-dev@python.org/thread/6KT2KIOTYXMDCD2CCAOLOI7LUGTN6MBS

Contents

The Python Community

Great software is supported by great people. Our user base is enthusiastic, dedicated to encouraging use of the language, and committed to being diverse and friendly.

Tweets b

Python
@ThePy

Check out the P
2020 Results jb
#pythondevsur

Python Dev
Official Pytho
jetbrains.com

Python
@ThePy

PSF Basic Mer
subscribe to ou
Mailing list deta
mail.python.org

community welcome and encourage
utual respect, tolerance, and encouragement,
rinciples.

nore varied, expressed in our diversity statement;
welcome you.

the Python Software Foundation)

ython Software Foundation here

www.python.org/psf-landing/

python SOFTWARE FOUNDATION

Donate | Search | GO | Socialize | Sign In

About | Sponsorship | Membership | Donations | Grants | Invoices & Reimbursements | Legal | Media | PSF Blog

The Python Software Foundation is an organization devoted to advancing open source technology related to the Python programming language.

The official Python Developers Survey 2020 results are here! Python Developers Survey 2020

We Support The Python Community through...

Grants

In 2019 we awarded $326,000 USD for over 200 grants to recipients in 60 different countries.

Infrastructure

We support and maintain python.org, The Python Package Index, Python Documentation, and many other services the Python Community relies on.

PyCon US

We produce and underwrite the PyCon US Conference, the largest annual gathering for the Python community. Our sponsors' support enabled us to award $138,162 USD in financial aid to 144

Pro: 3rd Party Libraries

pypi.python.org

Help Sponsor Log in Register

Find, install and publish Python packages with the Python Package Index

Search projects

Or browse projects

294,822 projects 2,419,631 releases 3,951,917 files 489,037 users

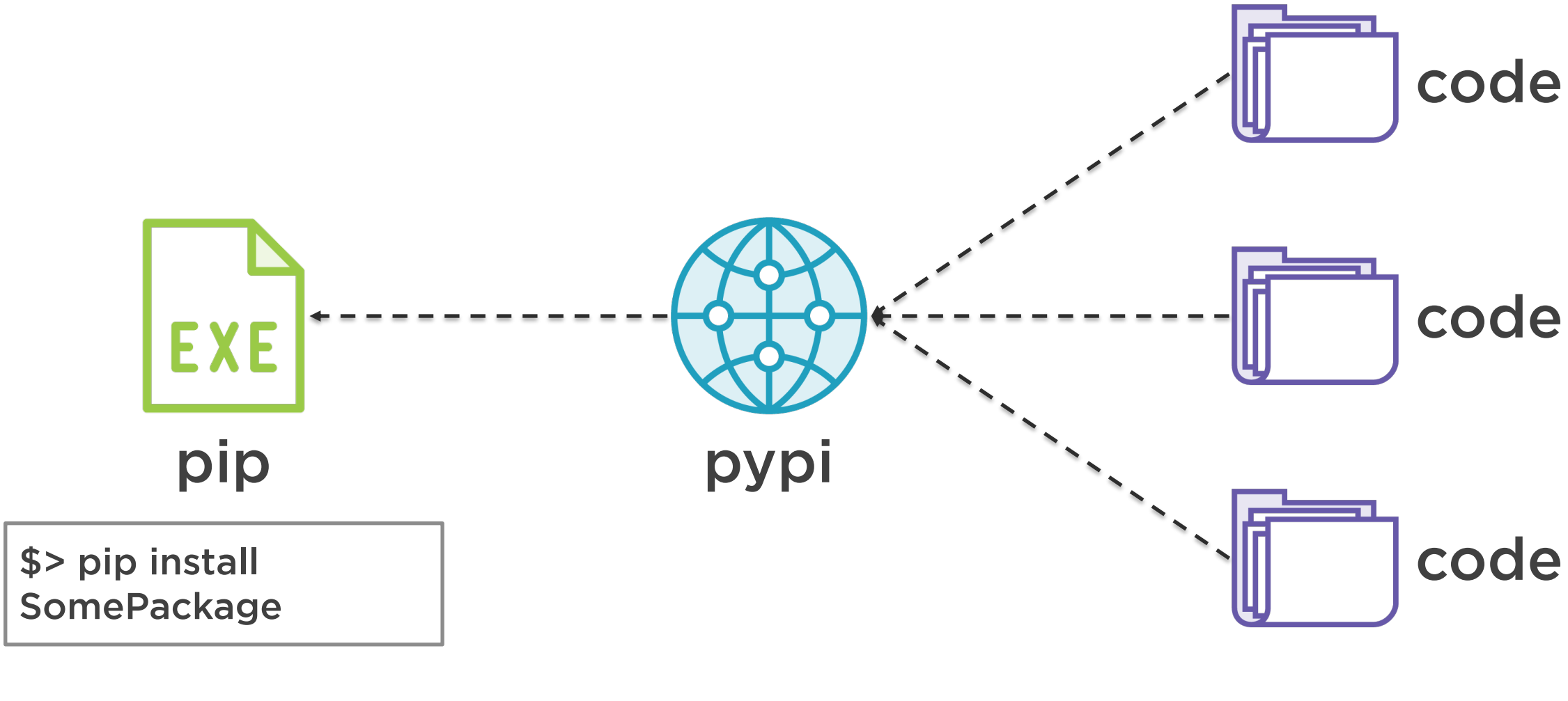
python™ Package Index

The Python Package Index (PyPI) is a repository of software for the Python programming language.

PyPI helps you find and install software developed and shared by the Python community. Learn about installing packages.

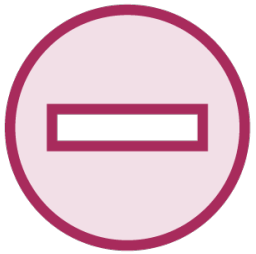
Package authors use PyPI to distribute their software. Learn how to package your Python code for PyPI.

Working with the Code of Others



pip

Install



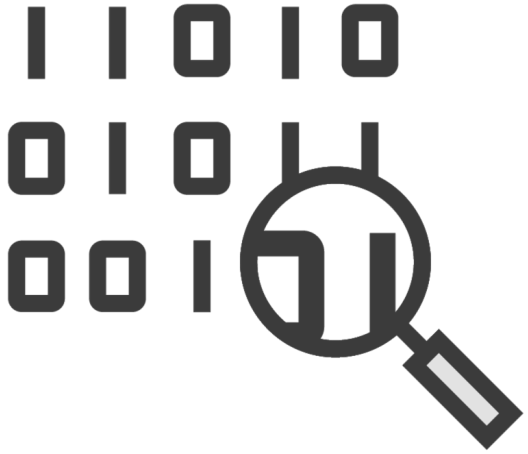
Uninstall

Dependencies

Package Groups

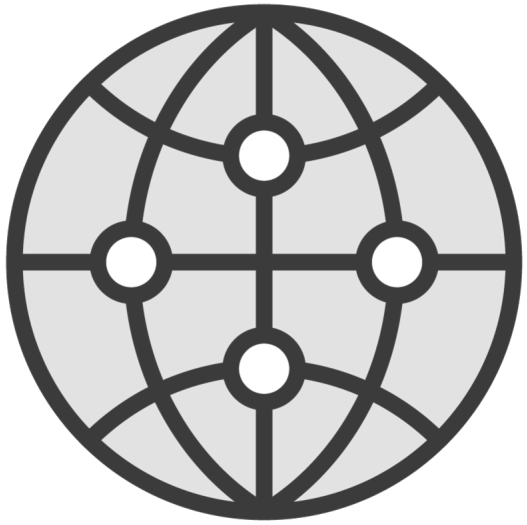
Versions

Popular 3rd Party Libraries



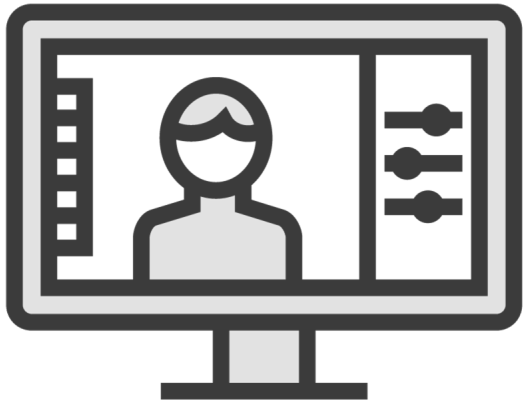
Data Science

NumPy, Pandas, Matplotlib, Tensorflow, Keras, Scikit Learn



Web Development

Flask, Requests, Django



Images / Computer Vision

Pillow, Pygal, OpenCV, Mahotas

Applications

wxPython (GUI), PyGTK (GUI), Fire (CLI), Kivy (Mobile/Multi-touch), Pygame

If you need it, it probably exists!

Pro: 3rd Party Tools

Python IDEs and Editors

Pydev

Pycharm

Visual Studio Code

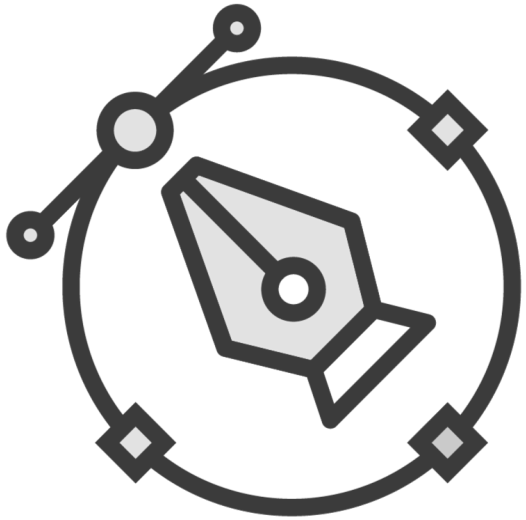
Spyder

Sublime

Vim

GNU/Emacs

Python Code Tools



Style Guide Enforcement

flake8

Code Analysis

PyLint

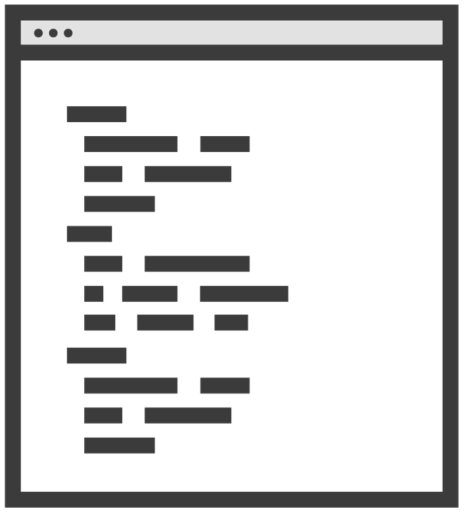
Code Formatter

black

Performance Analyzers

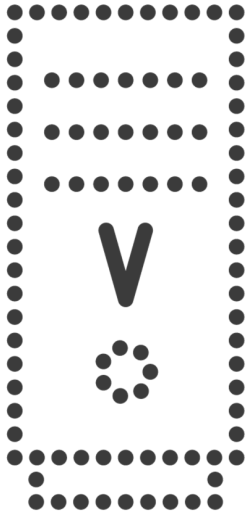
Python Cons

Python Drawbacks



Interpreted

Slow



Not Native

High memory usage, lack of native security sandbox

Dynamic

Runtime errors

Do the pros outweigh the cons?

The Big Picture Summary

Why Python?

Simple to learn

Simple to use

Great community

Widely used

High demand

What Is Python?

Dynamically-typed

Syntax

Garbage-collected

General-purpose

Interpreted

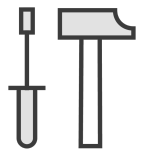
Multi-paradigm

Python Pros and Cons

Pro: Comprehensive Standard Library

Pro: Community-driven

Pro: 3rd Party Libraries



Pro: 3rd Party Tools

Python Drawbacks (Interpreted, Not Native, and Dynamic)

Thank you so much!

Jason Olson
Software Engineer
@jolson88 www.jolson88.com