Python Introduction, Resources and FAQs

Python is a high-level, general purpose, interpreted scripting language. The language's design has a strong emphasis on code readability, as well as flexibility and ease of use.

Like most similar languages (Ruby and Perl, for example), Python *can* be used for anything, for any type of application. There are a handful of web apps built using Python, and some excellent web development frameworks. With tools like Py2Exe, you can package Python scripts as stand-alone desktop applications.

But where Python really shines, and where it gets most of its use, is as a tool for data analysis, number crunching, scientific research, hacking, scripting, and task automation. Python is used by scientists, mathematicians, penetration testers, spies, cryptographers, musicians, stock brokers, and network administrators for a wide range of applications.

If you want to learn how to build nice web applications, you probably want to start with Ruby or (if you're into WordPress), PHP. But if you want to learn how to use computers to do interesting new things, Python is the language for you.

Online Python Resources

There is *a lot* of Python material on the web — some of it free, some of it paid. These are some of the best resources for learning Python.

Learning Python

Recommended Courses and Tutorials

- The official Python site
- Beginner's Guide to Python — The official place to start learning.
    - For non-programmers
    - For programmers new to Python
- Learn Python the Hard Way — An excellent Python course for people who want to really learn the language without taking any shortcuts.
- Google's Python Class — Google uses Python for a number of projects. Want to learn it the way they teach it? You can.
- Guide to the Standard Library
- Python Koans — An *amazing*, philosophical approach to Python programming and thinking like a developer, through the use of Test Driven Development.

Other Great Python Tutorials and Learning Resources

Here are some very good resources for learning Python that almost made it into the above "Recommended" list. Everyone has a different learning style, so maybe one of these will suit you better than the tutorials above.

- Python Videos — Huge list of videos about all sorts of Python Topics
- Python Programming for Beginners — Bare bones tutorial from Linux Journal.
- A Python Book — An in-depth online book that covers beginning Python, advanced topics, and mny exercises.
- Python Course — A free and extensive set of tutorials for Python.
- Building Skills in Python — 42 chapters of Python exercises designed
- Lark's Tongue Guide to Python — A set of simple project-based tutorials on Python concepts.
- Python 101 — An introduction to Python.
    - Python 201 — Intermediate Python.
- Learn Python in 10 Minutes — Very quick introduction to the language.
- Checkio — Learn to improve your code with a game. Playing a game, not writing one.
- Python Programming Wiki Book

Additional Python Tutorials

These are "Honorable Mention" tutorials on beginning Python. We didn't find them quite up to our (very high) standards, but they are all fairly popular — so maybe one of them will work well for you.

- Pythonspot
- Python Introduction
- Intro to Python
- Instant Python
- Python Tips
- Python Tutorial

Python Development Tools

Libraries, Plugins, and Add-ons

A big part of the strength of Python is the ecosystem of excellent tools for accomplishing a number of different types of tasks in the language. From graphics processing to mathematical analysis, there's probably a Python module for just about any domain you are working in.

- Shrapnel — Python Library for high-volume concurrency.
- MatPlotLib — Graphics and data visualization.
- Mako — Web templating engine.
- PIL — Python Imaging Library
    - Pillow — Fork of PIL. (PIL seems to have ceased active development. Pillow is picking up the slack on this popular project.)
- Pyx — Python Graphics package
- Beautiful Soup — Tools for screen scaping and then handling the parse-tree and content.
- Scrappy — Web scraping tools.
- Gooey — Tools for providing a GUI for command-line Python programs.
- Peewee — A minimalist SQL ORM for connecting a Python application to MySQL, PostgreSQL, or SQLite.

- SQL Alchemy — A more fully-featured SQL ORM.
- PyGame — Platform for building video games in Python.
- SciPy — Science and math tools for Python; very important to for scientific computing.
- Pandas — Data analysis tools.
- sh — Library for calling other programs from within Python scripts.

The Python wiki also maintains a list of some of the most useful and popular Python modules.

IDEs

An IDE is an Integrated Development Environment, a tool for managing the development of a large or complex application. Most Python users tend to work in a simple code editor, but a number of excellent Python-focused IDEs are available for those using Python for larger projects.

- PyDev — Python IDE for Eclipse.
- Komodo — Python IDE
- PyCharm — Python IDE
- Wing IDE — Commercial Python IDE.
- Spyder — Python IDE for Science and Mathematics.
- NetBeans
- PyScripter
- µ.dev — Microdev, a weird little IDE from Sakura studio.

Also see this list of Python editors.

Refactoring and Code Checking

Python developers have a culture that tends to prefer clean and efficient code. At the same time, they also value speed, and often plunge into coding quickly in order to solve immediate problems. A number of tools have developed to help Python programmers automate the task of checking code and making it more efficient.

- PyChecker — Code checker (linter).
- Rope
- Bicycle Repair

Build Tools

Python excels at task automation, so it should be no surprise that there are a number of tools for doing just that, and for speeding up build and deploy cycles.

Also included in this list are specialized development tools that are used for packaging and distributing Python apps.

- Invoke - Tasks execution and scripting tool.
- Microbuild — Lightweight build tool.

- Paver — Task scripting.
- Pynt — Build tool.
- VirtualEnv — Tool for building isolated Python environments.
- Bitten — Continuous integration tool for Python and Trac.
- iPython — Interactive Python shell and development library; too many cool features to list.
- Py2Exe — Compiles Python scripts into Windows executables.

Web Frameworks

If you want to use Python to build a web application, there are a number of low-level tasks you'll need to take care of first — or you could just start from step 10 and use a web application development framework.

- Django — By far, the most popular web application framework for Python. It is conceptually similar to Ruby on Rails.
- Python Paste — Not a framework, but a "framework for frameworks." Provides low-level tools for building Python web frameworks.
- CherryPy — Minimalist Python web framework.
- TurboGears
- Web2Py
- CubicWeb
- Giotto
- Reahl
- Wheezy

Applications Built in Python

Python is used by a lot of people, for a lot of different tasks and purposes, but is not hugely popular for building apps to be distributed as code to consumers and end users (the way, for example, PHP is). Still, it is sometimes used for this purpose. Here are some examples of some applications built in Python.

- SchoolTool — Student information system.

Content Management Systems

- Plone — The most popular Python-based CMS.
- Django CMS — Not as popular as Plone for CMS, but built on top of the most popular Python framework.
- MoinMoin — Python wiki-engine that powers the Python wiki. (For other Python-based Wiki CMSes, see this page.)
- CPS-CMS
- Silva
- ZMS

Online Python Reference

These are some of the most important single-source Python reference sites, which you should probably bookmark.

- Python Documentation
- Python Wiki
- Popular Python Recipes
- Python Package Index

Books

Beginning Python

If you prefer to learn using a printed book, there is no shortage of excellent ones available. Here are some of the best Python books for beginners. If you don't prefer books (or don't prefer paying for things) scroll down to the Online Resources section for a list of excellent online (and mostly free) tutorials.

- Learn Python the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code
- Python for Kids
- Python Crash Course: A Hands-On, Project-Based Introduction to Programming
- Python Programming for Beginners: An Introduction to the Python Computer Language and Computer Programming
- Python Programming: An Introduction to Computer Science
- Automate the Boring Stuff with Python: Practical Programming for Total Beginners
- Learning Python
- Hello! Python

Advanced

Many of the more advanced concepts in Python programming are not covered in online tutorials, and can only be found in printed books.

- Expert Python Programming
- Python High Performance Programming
- Python Parallel Programming Cookbook
- Python in Practice: Create Better Programs Using Concurrency, Libraries, and Patterns
- Advanced Python 3 Programming Techniques
- Mastering Object-oriented Python
- Effective Python: 59 Specific Ways to Write Better Python
- Python Machine Learning
- Think Python
- Python: Learn Python Regular Expressions FAST!

- Professional Python
- High Performance Python: Practical Performant Programming for Humans
- Web Scraping with Python: Collecting Data from the Modern Web
- Test-Driven Development with Python
- Fluent Python
- Python Playground: Geeky Projects for the Curious Programmer
- Foundations of Python Network Programming: The comprehensive guide to building network applications with Python
- Guide To: Learning Iteration and Generators in Python

Python for Math, Science, and Data

Python is widely used in math and science for a at least a couple of reasons: - There are great math and science tools for the language, such as the SciPy and NumPy. - The language lends itself well to quick programming tasks, so it is easy to use Python for *ad hoc* data analysis without building fully-featured apps.

As with general Advanced topics, if you are looking for information on specialized topics in advanced Python programming, you will find a lot more excellent books than free websites and online tutorials.

- Doing Math with Python
- Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython
- Python Data Science Essentials
- Data Science from Scratch: First Principles with Python
- Learning Data Mining with Python
- Advanced Python for Biologists
- Python for Finance: Analyze Big Financial Data
- Data Structure and Algorithmic Thinking with Python: Data Structure and Algorithmic Puzzles
- Python for Informatics: Exploring Information
- Derivatives Analytics with Python
- ArcPy and ArcGIS: Geospatial Analysis with Python
- Learning Pandas - Python Data Discovery and Analysis Made Easy
- Learning Geospatial Analysis with Python
- Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data

Python for Hacking

Because of its suitability for *ad hoc* programming, and for task automation, Python also gets used quite a bit by people who like to break into things, and also by the people who try to stop them. (We assume you are one of the good guys, of course.)

- Violent Python: A Cookbook for Hackers, Forensic Analysts, Penetration Testers and Security Engineers
- Black Hat Python: Python Programming for Hackers and Pentesters

- Python for Secret Agents

Reference

Here are a few great desk references for Python, for people who like that sort of thing. While some of the advanced topics mentioned above are book-only, most of the basic reference material here is easier to find online with a good search engine — but some people prefer to have hard copies.

- Python Essential Reference
- Python Cookbook
- Python Pocket Reference

Python FAQ

What does it mean that Python is a "scripting language"?

A scripting language is a language that is interpreted at run time, rather than compiled into a binary executable.

Some people use the phrase "scripting language" to indicate that the language is particularly good at writing short "scripts," or miniature *ad hoc* programs used to automate tasks.

Python fits both descriptions — it is an interpreted language, and it is also highly useful for writing short, *ad hoc* scripts.

Are scripting languages like Python good for writing full-scale applications?

There are some people who have a bias against the use of scripting/interpreted languages for entire applications. The wisdom of this bias is entirely up to individual context.

Scripting languages tend to run a little slower than compiled languages, and in some instances this difference in performance is a huge issue. In most contexts, though, it is a negligible concern.

Python is perfectly well suited for writing applications of all kinds. Using Django or another web framework allows you to build web-based applications. There is nothing deficient about Python in terms of the tools and capabilities needed to write full-scale applications. In fact, Python is arguably much better suited to such work than either PHP or JavaScript, both of which are frequently used for large, complex web applications.

Should I learn Python?

That depends on your goals.

If you are hoping to build typical web applications, you should probably learn PHP or Ruby (and Rails), along with with JavaScript, HTML, and CSS. There's no reason you could not use Python for this work, but it is not typical to do so. PHP and Ruby would give you access to a lot more existing web applications, frameworks, and web development tools.

If you are looking to use programming skills to directly accomplish tasks, such automation or analysis, Python is an excellent language for that sort of work, and is where it gets most of its use.

If you are building apps that need to manipulate data in a a specialized field or domain — such as math, science, finance, music, or cryptography — Python is an excellent language for these sorts of projects as well.