🐍

Python Developer Roadmap

follow @techie_programmer ! for more content

🔰 Stage 1: Beginner (0–3 months)

🎯 Goal:

Understand Python fundamentals, basic programming concepts, and start writing small scripts.

🧠 Key Topics:

- Python installation & setup (Windows/Linux/Mac)
- Variables and Data Types
- Basic Input/Output
- Operators
- Conditional Statements (if/else)
- Loops (for, while)
- Functions
- Error Handling (try/except)
- Basic Data Structures:
  - Lists
  - Tuples
  - Dictionaries
  - Sets
- String Manipulation
- Basic File I/O

🛠️ Tools & Technologies:

- Python 3.x
- Text Editor/IDE: VS Code, PyCharm (Community)
- Terminal/Command Prompt
- Git & GitHub (basic usage)

⏳ Suggested Duration:

8–12 weeks (8–10 hours/week)

📚 Recommended Resources:

- **Books:**
  - "Automate the Boring Stuff with Python" by Al Sweigart *(free online)*
- **Courses:**
  - Python for Everybody – Coursera (University of Michigan)
  - CS50's Introduction to Computer Science – edX
- **Practice:**
  - W3Schools Python
  - HackerRank Python Domain

⚙️ Stage 2: Intermediate (3–6 months)

🎯 Goal:

Work with APIs, object-oriented programming, and real-world problem solving. Start small projects.

🧠 Key Topics:

- Object-Oriented Programming (OOP)
- Modules & Packages
- Virtual Environments (venv, pip)
- Python Standard Library (e.g., os, datetime, collections)
- Working with APIs (REST, requests module)
- JSON & CSV Handling

- Advanced File I/O
- Intro to Testing (unittest, pytest)
- Debugging & Logging
- List Comprehensions
- Lambda Functions & Higher-Order Functions
- Basic Functional Programming Concepts

🛠️ Technologies & Tools:

- GitHub (version control & collaboration)
- Python Virtual Environments
- CLI tools (pip, poetry or pipenv optional)

Projects to Build:

- To-Do App (CLI)
- Weather App (API-based)
- File Organizer Script
- Expense Tracker

⏳ Suggested Duration:

12–16 weeks (10–12 hours/week)

📚 Recommended Resources:

- **Courses:**
  - Real Python Tutorials
  - Intermediate Python – Python Morsels
- **Books:**
  - “Fluent Python” by Luciano Ramalho *(for later in this stage)*
- **Practice:**
  - Exercism.io – Python Track
  - LeetCode Easy-Medium Problems
  - Codewars

🚀 Stage 3: Advanced (6–12 months)

🎯 Goal:

Master advanced topics, specialize in a domain (e.g., Web Dev, Data Science, Automation), contribute to open source.

🧠 Key Topics:

- Design Patterns
- Generators, Iterators, and Coroutines
- Decorators
- Context Managers
- Advanced OOP (Mixins, Abstract Base Classes)
- Multithreading & Multiprocessing
- Type Hinting & mypy
- Performance Profiling & Optimization
- Working with Databases (SQL & ORMs like SQLAlchemy)
- Testing (advanced pytest, test coverage, mocking)
- Packaging & Publishing Python Packages
- Asynchronous Programming (asyncio)
- Security Best Practices

🛠️ Technologies (by Domain):

🕸️ Web Development:

- Flask or Django
- Jinja2
- HTML/CSS/JS (basic)
- PostgreSQL or SQLite
- REST APIs
- Docker (basic containerization)

📊 Data Science & ML:

- Numpy, Pandas, Matplotlib, Seaborn

- Jupyter Notebooks
- Scikit-learn
- TensorFlow or PyTorch (optional)

🤖 Automation/Scripting:

- Selenium or Playwright
- OpenPyXL / Pandas for Excel/CSV
- cron jobs, schedulers

⏳ Suggested Duration:

4–6 months (10–15 hours/week)

📚 Recommended Resources:

- **Courses:**
    - Full Stack Python
    - Django for Beginners – William S. Vincent
    - MIT 6.0001 (Advanced Python)
- **Books:**
    - "Effective Python" by Brett Slatkin
    - "Python Cookbook" by David Beazley
- **Projects:**
    - Blog Web App
    - API-based Service
    - Web Scraper with Scheduler
    - Machine Learning Mini Project (if applicable)

🌱 Optional: Specialization Tracks (Post-Advanced)

Choose based on your interest:

Domain	Tools & Skills
Web Dev	Django, Flask, DRF, Docker, AWS/GCP, CI/CD
Data Science	Numpy, Pandas, Scikit-learn, SQL, Jupyter, ML Ops
DevOps/Scripting	Bash, Ansible, Terraform, Python Automation
Game Dev	Pygame, Panda3D
AI/ML	TensorFlow, PyTorch, NLP, Transformers

📌 Final Tips

- **Daily Practice**: Code every day, even just a bit.
- **Projects > Tutorials**: Apply what you learn.
- **Open Source**: Contribute to projects on GitHub.
- **Networking:** Join Python communities (Reddit, Discord, Stack Overflow).
- **Build a Portfolio:** Showcase your work with GitHub and a personal site.

Follow for more