AI / PYTHON / SOFTWARE DEVELOPMENT

Python Vibe Coding Tools

This tutorial provides you with valuable information on a set of tools to use for vibe coding in Python.

Aug 3rd, 2025 9:00am by Jessica Wachtel

Ready for a crazy stat? In this Wall Street Journal piece on vibe coding, it says Gartner projects that within three years, 40% of new business software will be developed using AI-assisted techniques. It's amazing to see how things have changed since I went to coding school. I imagine my experience would have been quite different with AI's help.

That said, now that we're in this new world of vibe coding, what actually is vibe coding and how can we all make sure not to get left behind? Vibe coding is a term coined by Andrej Karpathy in February 2025. He said, "Where you fully give in to the vibes, embrace exponentials, and forget that the code even exists … I just see stuff, say stuff, run stuff, and copy/paste stuff, and it mostly works."

What that means to Karpathy is that vibe coding is a way of programming where you talk to an AI instead of writing all the code yourself. You describe what you want in plain language, and the AI writes the code for you. You then give more instructions, test the results, and make changes by talking to the AI, not by typing every line. In short, this translates to less writing code, more describing ideas and functionality. It won't make developers obsolete per this post in Business Insider, but it will definitely be helpful to master the skill.

AI tools are everywhere, and it seems like five new ones pop up every day. Let's take a look at some of the more helpful tools to help you vibe code in Python.

Python Vibe Coding Tools

Most of these tools can generate code in any language, but it's a great idea to stick with languages you're familiar with; in this tutorial, we'll use Python.

GitHub Copilot

GitHub Copilot is an AI assistant that lives inside your code editor. It reads your comments or code and suggests complete functions, snippets or tests. GitHub Copilot is especially useful when it comes to:

- Writing repetitive or boilerplate code.
- Getting suggestions for functions or snippets based on simple descriptions.
- Learning how to use new libraries or APIs.
- Prototyping ideas without starting from scratch.

How to use it

TRENDING STORIES

1. Python: Introduction to Timestamps and Time Strings
2. Python for Beginners: Data Types
3. Decode Any Python Code With This 5-Step Method
4. Python Pandas Ditches NumPy for Speedier PyArrow
5. NVIDIA Finally Adds Native Python Support to CUDA

After installing it in your IDE, you can type a comment like `# Download an image from a URL and convert it to grayscale`. Copilot will respond with the full code section:

```
# Download an image from a URL and convert it to grayscale

import requests
from PIL import Image
from io import BytesIO

url = "https://example.com/image.jpg"
response = requests.get(url)
image = Image.open(BytesIO(response.content))

grayscale_image = image.convert("L")
grayscale_image.save("grayscale_image.jpg")
```

gistfile1.txt hosted with ❤️ by **GitHub** **view raw**

Cursor

Cursor is a version of VS Code that comes with a built-in AI assistant. It can edit, refactor or explain your code whenever you ask. Since Cursor is a code editor, you can install it directly on your machine. Cursor is especially helpful for:

- Improving and cleaning up existing code.
- Adding features like async functions or error handling.
- Debugging tricky parts of larger projects.
- Keeping your code organized and easy to maintain.

Cursor allows you to highlight a piece of code and type a request like, "Refactor this to use async."

```
# Original code
def fetch_data():
    data = get_data_from_api()
    process(data)
```

gistfile1.txt hosted with ❤️ by **GitHub** **view raw**

It will respond automatically with updated code:

```
# Refactored code with async
async def fetch_data():
    data = await get_data_from_api()
    process(data)
```

gistfile1.txt hosted with ❤️ by GitHub view raw

Open Interpreter

Open Interpreter is a local, open source AI assistant that acts like a terminal-savvy coding buddy. It can run commands, generate scripts and help automate tasks directly from your machine. Open Interpreter is especially helpful for:

- Writing and running small scripts quickly.
- Automating repetitive tasks.
- Working in a local or offline environment.
- Interacting with your terminal through natural language.

After installing it locally, launch the assistant in your terminal:

```
pip install open-interpreter
interpreter
```

gistfile1.txt hosted with ❤️ by GitHub view raw

Once you're inside the Open Interpreter interface, you just type a plain English request like, "Create a Python script that sends a desktop notification every hour."

Open Interpreter will respond by generating and running (when necessary) something like this:

```
# notification_script.py

import time
from plyer import notification

```

```
while True:
    notification.notify(
        title='Reminder',
        message='Time to take a break!',
        timeout=10
    )
    time.sleep(3600)
```

gistfile1.txt hosted with ❤️ by **GitHub** **view raw**

If you say something like, "Save that script as reminder.py and run it," it will follow those instructions, too.

AI Notebook Tools

AI notebook tools (like AI Notebook Copilot) are add-ons for Jupyter that turn natural language prompts into code cells. These tools are especially useful for:

- Quickly generating data analysis or visualization code.
- Exploring datasets interactively.
- Teaching or learning with instant code examples.
- Streamlining notebook-based development.

You will need to install the Jupyter and GitHub Copilot extensions in VS Code, then open your Jupyter notebook to get started.

In a Jupyter Notebook code cell, type a prompt as a comment:

```
# %% Prompt
# Create a pandas DataFrame with 5 rows of random integers between 1 and 100
```

gistfile1.txt hosted with ❤️ by **GitHub** **view raw**

The AI will fill in the code in the next cell:

```
import pandas as pd
import numpy as np

data = np.random.randint(1, 101, size=(5, 3))  # 5 rows, 3 columns of random ir
df = pd.DataFrame(data, columns=['A', 'B', 'C'])
print(df)
```

gistfile1.txt hosted with ❤️ by GitHub view raw

Tips for better vibe coding

- **Be specific with your prompt.** Rather than saying "make a graph," instructions like "plot a bar chart showing three temperature readings in Fahrenheit" will yield better results.
- **Similar to actual coding, break big steps into smaller parts.** Giving the AI step-by-step guidance produces cleaner, more accurate code.
- **Don't assume everything is perfect — AI makes mistakes.** Test the code for things like correct logic and security.
- **Practice!** Just because it's AI doesn't mean there isn't still a learning curve. Your first project will always be your worst project, but you can only go up from here!

Conclusion

Vibe coding helps you quickly turn ideas into working code without getting lost in syntax or boilerplate. It's about collaborating with AI to speed up development and keep your focus on what matters most: building cool stuff.

TNS

Jessica Wachtel is a developer marketing writer at InfluxData where she creates content that helps make the world of time series data more understandable and accessible. Jessica has a background in software development and technical journalism.

Read more from Jessica Wachtel →

HAPROXY

HAProxy Enterprise WAF protects against Microsoft SharePoint CVE-2025-53770 / CVE-2025-53771

22 July 2025

HAProxyConf 2025 Recap

9 July 2025

Announcing HAProxy 3.2

28 May 2025

embrace

Exploring User Journeys: a walkthrough of our latest user-focused observability feature

22 July 2025

Introducing User Journeys: Now user engagement is a reliability signal

22 July 2025

Embrace welcomes Jason Robinson as Chief Revenue Officer

21 July 2025

Signadot

Why We Shift Testing Left: A Software Dev Cycle That Doesn't Scale

20 May 2024

Improve Developer Velocity by Decentralizing Testing

23 March 2024

We Need a New Approach to Testing Microservices

16 March 2024

TNS RSS Feeds

THE NEW STACK

About / Contact
Sponsors
Advertise With Us
Contributions

roadmap.sh

Community created roadmaps, articles, resources and journeys for developers to help you choose your path and grow in your career.

Frontend Developer Roadmap
Backend Developer Roadmap
Devops Roadmap

FOLLOW TNS