

Lesson 05 — Programming Today

What writing code actually looks like in 2026 — languages, IDEs, AI assistants, git, GitHub.

IT Landscape

Module 2

What is a “programming language”?

A formal language a human writes so a computer can do something predictable with it.

- **Hundreds exist** — new ones invented every year.
 - **Most paying jobs** use one of about ten.
 - Languages are **tools, not religions**.
-

The 2026 language landscape

Language	Where you'll find it
Python	Data, AI/ML, scripting, backends
JavaScript / TypeScript	Everywhere on the web — browsers + Node.js backends
Java	Enterprise systems + Android
C#	Windows, Unity games, Microsoft-shop enterprise
Go	Cloud infrastructure, CLIs (Docker, Kubernetes)
Rust	Systems, performance-critical, replacing C/C++ in places
Swift	iOS apps
Kotlin	Android + JVM backends

Plus **C / C++** — still huge in operating systems, embedded, games.

Note: as of 2026 — see TIOBE / GitHub Octoverse for current rankings.

An IDE in 2026

An **IDE** (integrated development environment) is a text editor with a lot of brains. It:

- Edits code with **syntax coloring** and autocomplete
- **Runs** the code and **debugs** it
- Shows **git changes** inline
- Runs **tests**
- Talks to the **AI assistant** in a side panel

Names you'll hear (as of 2026):

- **VS Code** — Microsoft, free, dominant.
 - **JetBrains IDEs** — PyCharm, IntelliJ, GoLand. Paid, very polished.
 - **Cursor / Windsurf** — VS Code forks with AI built in.
 - **Neovim** — modal, terminal-based, plugin culture.
-

AI coding assistants

The biggest shift in how code gets written since the internet.

- **GitHub Copilot** — inline suggestions, oldest at scale.
- **Cursor** — chat + edits + agent inside the editor.
- **Claude Code** — terminal-first, agentic.
- **Windsurf, Cody, Continue.dev** — same space, different flavors.

Most engineers in 2026 use at least one daily. AI writes a meaningful fraction of new code; the engineer's job has shifted toward **direction, review, and ownership**.

Note: as of 2026 — vendors and capabilities shift quarterly.

Git in one slide

Git remembers every change you make to your code.

- Every project has a `.git` folder.
- `git add` — choose what to save.
- `git commit` — snapshot it with a message.
- `git push` — send it to a shared server.
- **Branching** = work on a side line. **Merging** = bring it back.

That's 80% of daily git.

GitHub in one slide

GitHub is the most popular **place** to keep your git repositories. It also runs:

- **Issues** — bug and feature tracking.
- **Pull requests** — proposed changes that teammates review.
- **Actions** — automated checks that run on every change.
- The home of most **open source** software.

Alternatives: **GitLab**, **Bitbucket**, **Codeberg**, **Gitea**.

A real Python snippet

```
def greet(name):  
    return f"Hello, {name}!"  
  
people = ["Ada", "Linus", "Grace"]  
greetings = [greet(p) for p in people]  
  
if __name__ == "__main__":  
    for line in greetings:  
        print(line)
```

You don't need to run this — just notice the indentation and the readable shape.

A real TypeScript snippet

```
interface User {  
  id: number;  
  name: string;  
}  
  
function greet(user: User): string {  
  return `Hello, ${user.name}!`;  
}  
  
const users: User[] = [  
  { id: 1, name: "Ada" },  
  { id: 2, name: "Linus" },  
];  
  
users.forEach((u) => console.log(greet(u))):
```

Same idea, different language. Notice the explicit types — that's the "TS" in TypeScript.

The open source ecosystem

Most software is built on free, open code.

- **npm** (JS / TS) — ~3M packages.
- **PyPI** (Python) — ~500k packages.
- **Maven** (Java), **crates.io** (Rust), **Go modules**.

You install dependencies with one command:

- `npm install`
- `pip install`
- `go get`

Real apps have **hundreds** of indirect dependencies. Security and licensing matter.

Reading other people's code

Most of a working programmer's day.

Engineers read more code than they write.

Skills:

- Skim a repo's **README**.
 - Follow **imports**.
 - Navigate by **symbol search** (Cmd-T in VS Code).
 - Use AI to **summarize** unfamiliar files.
-

Career paths from “writes code”

Build the product

- Backend engineer
- Frontend engineer
- Full-stack engineer
- Mobile engineer

Adjacent roles

- Data engineer
- DevOps / SRE
- ML engineer
- Engineering manager

Lesson 14 returns to this map.

Recap

- The **language pool** is small once you know which jobs use which.
- The **editor** is just a text editor with a brain bolted on.
- **AI assistants** are the biggest shift in how code gets written since the internet.

Coming up: Lesson 06 — how software is shaped at the bigger-picture level.

