GitHub Copilot in the CLI

Copilot in Practice | Intermediate

Session 04 of 18 | Duration: 3 hours (1hr content + 2hr lab)

Agenda & Time Plan

Time	Activity	Duration
0:00	Why a Dedicated CLI Session?	5 min
0:05	Installation & Setup	5 min
0:10	Core Interactive Modes	10 min
0:20	Programmatic Mode: Scriptable AI	10 min
0:30	Built-in Agents & Slash Commands	10 min
0:40	CLI-Exclusive Features	10 min
0:50	CLI vs. IDE Decision Framework	7 min
0:57	Security Considerations	3 min
1:00	Wrap-Up & Lab Preview	3 min
1:03	☕ Break	10 min
1:13	Lab Start	—
3:00	Wrap-up	—

Breaks: 10-minute break between trainer content and lab. Additional 5-minute breaks at trainer's discretion.

Why a Dedicated CLI Session

The CLI is a separate Copilot surface. It supports terminal-first workflows.

1. It can read and write files, run commands, manage Git, and coordinate subagents.
2. It supports programmatic use and other terminal-oriented features.
3. It works in terminals, including SSH, CI/CD pipelines, and headless containers.

> This session is especially useful for people who work mainly in a terminal.

Installation & Setup

Method	Command
npm (recommended)	`npm install -g @github/copilot`
Homebrew	`brew install --cask github-copilot-cli`
WinGet	`winget install GitHub.Copilot`
Install script	`curl -fsSL https://gh.io/copilot-install | bash`

Authentication:

```
copilot            # Launch interactive mode
/login             # Triggers OAuth device flow
```

For CI/CD: set `COPILOT_GITHUB_TOKEN` or `GH_TOKEN`.

Three Interactive Modes

Cycle between modes with `Shift+Tab`:

Mode	Behavior	When to Use
Standard	Ask/execute — asks permission before acting	Default for most work
Plan	Analyzes, asks questions, builds structured plan BEFORE coding	Complex tasks; review approach first
Autopilot	Fully autonomous — no approval prompts	Trusted tasks, batch operations

Plan mode is what a lot of users wish IDE agent mode had. It asks questions and shows the plan before touching anything.

Key Slash Commands

Session Management:

Command	Purpose
`/clear` / `/new`	Start fresh conversation
`/resume`	Resume a previous session
`/compact`	Compress context to free tokens

Agentic Commands:

Command	Purpose
`/agent`	Browse and select specialized agents
`/delegate`	Send work to the cloud agent
`/fleet`	Enable parallel subagent execution
`/plan`	Create implementation plan
`/research`	Deep research via GitHub + web

Programmatic Mode: Scriptable AI

Run a prompt from the command line with this CLI feature.

```
copilot -p "PROMPT" [OPTIONS]
```

Embed Copilot in scripts, CI/CD, and cron jobs:

```
copilot -p "Generate a changelog from the last 10 commits" \
  --silent > changelog.md

copilot -p "Fix all ESLint errors in src/" --autopilot

cat error.log | copilot -p "Summarize the top 5 error patterns"
```

Programmatic mode supports terminal automation.

Built-in Specialized Agents

The CLI includes verified built-in agents:

Agent	Purpose
Explore	Fast codebase exploration and Q&A
Task	Runs commands such as tests, builds, lints
Code Review	High-signal review of diffs and changes
Plan	Plans implementation before code changes
Rubber Duck	Challenges assumptions and strengthens ideas

```
/agent    # Browse and select a specialized agent
/plan     # Create an implementation plan
```

CLI-Exclusive Features

Feature	CLI	IDE
Autopilot mode	✅	❌
Programmatic mode (`-p`)	✅	❌
Session persistence (`/resume`)	✅	❌
Custom model providers (Ollama, Azure)	✅	❌
Subagent orchestration (`/fleet`)	✅	❌
Pipe output as input (`cat log | copilot`)	✅	❌
40+ slash commands	✅	Limited
Works over SSH / headless	✅	❌

Copilot App — Orientation

Copilot App is a dedicated surface for agent sessions and customizations.

Key capabilities:

- **Agent sessions** for app-managed work
- **Customize** for plugins, skills, MCP servers, and canvases
- **Canvases** for shared plans, boards, checklists, and other work artifacts
- **Approved access** based on plan, policy, and data boundary

When to use:

- Work that needs a shared artifact or direct steering
- Discovering approved customizations

> Session 08 covers the Copilot App, plugins, and canvas extensions.

Cross-IDE Comparison

Copilot works across multiple IDEs. Which features are available depends on the IDE:

IDE	Completions	Chat	Agent Mode	Best For
VS Code	✅	✅	✅	Primary platform; most features first
Visual Studio	✅	✅	✅	.NET developers
JetBrains	✅	✅	✅	Multi-language (IntelliJ, PyCharm, WebStorm)
Eclipse	✅	✅	⚠️ Preview	Agent mode in preview
Xcode	✅	✅	⚠️ Preview	Apple dev; agent mode in preview
Vim/Neovim	✅	❌	❌	Minimal overhead

For this curriculum: VS Code is the teaching baseline. Verify the currently supported experience for any other IDE in the official Copilot feature matrix and customer policy.

IDEs vary in feature completeness. VS Code is the reference. Check the Copilot feature matrix for the latest.

Four Surfaces, Four Workflows

Surface	Best For	Where It Lives
IDE (Chat & Agent)	Writing code, visual control	VS Code, Visual Studio, JetBrains, etc.
CLI	Automation, scripting, headless	Terminal (all platforms)
Copilot App	Shared artifacts and app-managed work	Copilot App
Inline completions	Flow-state coding	All IDEs (IDE-only feature)

You'll use all four in a normal week. Each solves a different problem.

CLI vs. IDE: Decision Framework

Choose CLI When...	Choose IDE When...
Working over SSH / remote server	You need visual file diffs
Automating tasks in scripts / CI	Working on a large multi-file refactor
Analyzing logs or command output	You want inline Chat for surgical edits
Running batch operations (Autopilot)	You want Copilot Spaces for context
No IDE available (containers, minimal env)	You want NES and inline completions
You want Plan mode for complex tasks	Visual project navigation matters

Choose the surface that fits the task. Many workflows use both.

Security Considerations

- CLI sends **file contents and terminal output** to the Copilot service
- Autopilot mode can execute commands **without approval** — use carefully
- In CI/CD, scope the `GH_TOKEN` to minimum required permissions
- Tool use is permissioned: approve once, approve similar for the session, or deny
- Pre-approve or limit tools with `--allow-tool`, `--deny-tool`, and `--available-tools`
- Never pipe secrets or credentials to Copilot

Same privacy story as IDE Copilot — encrypted in transit, never used for training.

Lab Time

Hands-on Exercises (2 hours)

Lab Overview

Exercise	Topic	Duration
1	CLI Setup & First Commands	25 min
2	CLI Agent Mode	30 min
3	CLI-Exclusive Features	35 min
4	CLI vs IDE Comparison & Workflow Patterns	30 min

Deliverable: A CLI workflow cheat sheet + a fixed Node.js project + a completed pipeline script.

What to remember

1. The CLI is a **standalone tool** for terminal-first agent workflows.
2. **Three modes**: Standard (safe), Plan (review first), Autopilot (fully autonomous)
3. **Programmatic mode** (`-p`) supports CI/CD and automation
4. **Built-in agents** (Explore, Task, Code Review, Plan, Rubber Duck) provide specialized capabilities
5. CLI works where IDEs don't — **SSH, containers, headless environments**
6. Use the CLI and IDE for the work each surface fits

Questions?

Take a moment to discuss with your trainer

Next Up

Session 05: Agent Mode in the IDE

Thank you!