ORIGIN ACADEMY · COURSE SYLLABUS

Golang Developer

Build fast, reliable backend services in Go: concurrency, APIs, databases, real-world open-source libraries, testing and production deployment.

18 weeks · about 12 hours a week · live cohort of up to 10
Curriculum last updated 19 November 2026

Availability

This course opens soon. Join the waitlist on the course page to hear first about dates and prices.

Before you start

- Some programming experience in any language
- Comfort with the command line
- English good enough to follow technical documentation and live sessions: about a credit (C6 or better) in WASSCE or NECO English Language, or an equivalent such as KCSE or IGCSE English (roughly CEFR level B1). You don't need the certificate: Phase 0 is free, so you can check for yourself
- A laptop running Linux, macOS, or Windows with WSL

How the course works

- A live cohort of up to 10 students, with a mentor.
- At least half your time is hands-on: every module ends with a lab.
- Live sessions are recorded, so you can rewatch them or catch up.
- Questions between sessions are answered within 24 hours on weekdays.
- A mentor reviews every phase project, like a pull request at work.

Phase 0 is free for everyone, before you enroll.

Curriculum

PHASE 0 · WEEKS 1–2 · FREE

Go Fundamentals

The go command and modules, syntax, types, slices, maps, zero values and packages.

- Why Go, and setting up
- The go command: run, build, test, fmt, vet
- Variables, types and zero values
- Slices, maps and strings
- Functions, closures and defer
- Packages and modules

You build: Command-line to-do app

PHASE 1 · WEEKS 3–4

Types, Interfaces & Generics

Structs, methods, pointers, interfaces, embedding and generics.

- Structs and methods
- Pointers and value vs pointer receivers
- Interfaces and type switches
- Embedding and composition
- Generics and type constraints

You build: Pluggable storage layer

PHASE 2 · WEEKS 5–6

Errors & Testing

Idiomatic errors, table-driven tests, mocks, httptest, benchmarks, fuzzing and coverage.

- Errors as values: wrapping and sentinel errors
- panic and recover
- Table-driven tests
- Mocks, stubs and httptest
- Benchmarks, fuzzing and coverage

You build: Fully tested parsing library

PHASE 3 · WEEKS 7–8

Concurrency

Goroutines, channels, select, the sync package, context, and patterns like worker pools and pipelines.

- Goroutines
- Channels: buffered and unbuffered
- select
- sync: Mutex, WaitGroup and Once
- context: cancellation and deadlines
- Patterns: worker pools, fan-in, fan-out and pipelines
- The race detector

You build: Concurrent web crawler

PHASE 4 · WEEK 9

Tools, CLIs & the Standard Library

Building command-line tools, I/O and files, JSON, structured logging, code generation, linting and vulnerability checks.

- Building CLIs with flag and Cobra
- I/O, files and bufio
- encoding/json and struct tags
- Structured logging with slog
- go generate and go:embed
- golangci-lint and govulncheck

You build: Polished CLI tool released as a binary

PHASE 5 · WEEKS 10–11

Web APIs

net/http, routing, JSON APIs, middleware, authentication and an introduction to gRPC.

- net/http and routing
- JSON APIs
- Middleware
- Authentication and sessions
- gRPC and Protocol Buffers

You build: REST API with auth

PHASE 6 · WEEKS 12–13

Data & Persistence

PostgreSQL, SQL, pgx and sqlc, migrations and background jobs.

- SQL and PostgreSQL essentials
- pgx
- Type-safe queries with sqlc
- Migrations
- Background jobs

You build: API with a database and job queue

PHASE 7 · WEEKS 14–15

Building on Open-Source Libraries: the Firecracker Go SDK

How to choose, read and depend on someone else's library, using AWS's Firecracker Go SDK as the case study: boot and manage microVMs from Go.

- Choosing a dependency: licences and compliance, maintenance, security
- Modules in practice: go get, versions, replace and vendoring
- Reading a library: godoc, examples and source
- Case study: Firecracker and microVMs
- The SDK's design: config structs, functional options, builders and handlers
- Booting a microVM from Go: kernel, drives and networking
- Wrapping a dependency behind your own interface for testing
- Contributing upstream: issues and pull requests

You build: MicroVM launcher service with an HTTP API

PHASE 8 · WEEK 16

Production Go

Docker, observability, profiling with pprof and trace, garbage collection and escape analysis, and cloud deployment.

- Building small images with Docker
- Logs, metrics and traces
- Profiling with pprof and trace
- Memory: the GC and escape analysis
- Deploying to the cloud

You build: Deployed, monitored service

PHASE 9 · WEEKS 17–18

Capstone

Build a production backend in a team, to an enterprise brief.

- Design doc
- Build in sprints
- Code review and CI/CD
- Demo

You build: Payments-style service with CI/CD

Your certificate

You earn it by doing two things:

- Attend every live session. If you miss one, make it up by watching the recording and submitting that week's lab within 7 days. You can do this up to 2 times per course.
- Score more than 50% on the final evaluation at the end of the course.

Your Enterprise Readiness Rating is separate: it scores your projects, capstone and assessments so employers can see how job-ready you are.

Questions

Email academy@originclouds.io, or see the course online: https://academy.originclouds.io/courses/golang-developer