+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated on 01/08/2026

Register

Jetbrains GoLand Training

3 days (21 hours)

Overview

JetBrains GoLand is a comprehensive IDE for developing in Go with maximum productivity: integrated navigation, refactoring, testing, and debugging. The training shows you how to use it on a daily basis to speed up your development cycles, make your code more reliable, and standardize your team practices.

In this training, you will learn how to configure GoLand for your projects (modules, GOPATH, SDK), master smart editing (auto-import, inspections, intentions), and use the integrated tools: execution, testing, coverage, profiling, and Git integration.

The approach is practical: guided workshops, demos on a common project, refactoring and debugging exercises. Deliverables include a reproducible configuration (key settings), workflow checklists (testing, review, CI), and productivity snippets (templates, live templates, run configurations).

Objectives

- Configure the Go environment (SDK, modules, variables) in GoLand.
- Navigate efficiently through a code base (symbols, usages, hierarchy).
- Refactor safely with inspections and quick fixes.
- Debug and analyze execution (breakpoints, goroutines, profiling).
- Industrialize the workflow (tests, coverage, Git, run configurations).

Target audience

- Beginner to intermediate Go developers.
- Backend developers who want to standardize their tools.
- Tech leads/lead devs in charge of quality practices.

Prerequisites

- Basic knowledge of Go programming (types, functions, packages).
- Basic understanding of (unit) testing and the CLI.
- Familiarity with Git (commit, branch, merge).
- Knowledge of Go concurrency (goroutines, channels) is appreciated.

Technical requirements

- GoLand installed (recent version) and installation rights.
- Go (toolchain) installed, access to a terminal.
- OS: Windows 10/11, macOS, or Linux.
- RAM: 8 GB minimum, 16 GB recommended.
- Internet access for plugins and Go module dependencies.

Jetbrains GoLand training program

[Day 1 - Morning]

Getting started with GoLand and setting up a Go environment

- Install GoLand and configure the Go SDK (GOROOT, GOPATH, Go Modules)
- Exploring the interface: Project, Editor, Terminal, Tool Windows, Search Everywhere
- Create/import a Go project, recommended structure, and module management
- Configuring the editor: formatting, inspections, templates, essential shortcuts
- Hands-on workshop: Create a modular Go project and validate the configuration (build/run).

[Day 1 - Afternoon]

Navigation, refactoring, and everyday productivity

- Advanced navigation: symbols, files, usages, call hierarchy
- Safe refactorings: rename, extract method/variable, move, change signature
- Code generation: implementations, stubs, tags, struct literals
- Code quality: inspections, quick fixes, gofmt/goimports integration
- Hands-on workshop: Refactor an existing package by improving readability and Go conventions.

[Day 2 - Morning]

Debugging and execution analysis in GoLand

- Launching in debug mode: breakpoints, conditions, logpoints, watch

- Inspecting variables, call stacks, goroutines, and threads
- Debugging tests: targeted execution, subtests, table-driven tests
- Diagnostics: panic handling, error analysis, and stack traces
- Hands-on workshop: Fixing a concurrency bug (goroutines) using the debugger.

[Day 2 - Afternoon]

Tests, coverage, and Go tools (lint, static analysis)

- Writing and running tests: go test, run configurations, filters
- Coverage: generation, reading reports, and uncovered areas
- Benchmarks and profiling: running benchmarks and interpreting results
- Quality: configuring golangci-lint, vet, staticcheck via GoLand
- Hands-on workshop: Implement a testing strategy + coverage and correct lint alerts.

[Day 3 - Morning]

Git integration and development workflows in GoLand

- Git operations: commit, amend, stash, rebase, cherry-pick from the IDE
- Branch management and conflict resolution with the merge tool
- Local code review: diff, annotations, history, blame
- Hooks and conventions: messages, automatic formatting before commit
- Hands-on workshop: Perform a complete feature branch workflow with rebase and conflict resolution.

[Day 3 - Afternoon]

Run configurations, Docker, and delivery flow optimization

- Run configurations: profiles (dev/test), environment variables, parameters
- Dependency management: module updates, vendor, version checking
- Docker in GoLand: running, debugging, and testing a containerized app
- Automation: tasks, external tools, CI preparation (tests, lint, build)
- Hands-on workshop: Containerize a Go API, run and debug it from GoLand, then set up a build pipeline.

Target companies

This training is intended for both individuals and companies, large or small, wishing to train their teams in a new advanced IT technology or to acquire specific business knowledge or modern methods.

Placement at the start of training

The positioning at the start of the training complies with Qualiopi quality criteria. Upon final registration, the learner receives a self-assessment questionnaire that allows us to assess their estimated level of proficiency in different types of technologies, as well as their expectations and personal objectives for the upcoming training, within the limits imposed by the selected format. This questionnaire also allows us to anticipate certain connection or internal security issues within the company (intra-company or virtual classroom) that could be problematic for the monitoring and smooth running of the training session.

Teaching methods

Practical training: 60% practical, 40% theory. Training materials distributed in digital format to all participants.

Organization

The course alternates between theoretical input from the trainer, supported by examples and discussion sessions, and group work.

Assessment

At the end of the session, a multiple-choice questionnaire is used to verify that the skills have been correctly acquired.

Certification

A certificate will be issued to each trainee who has completed the entire training course.