AMBIENT-IT

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

Updated on 11/08/2025

Sign up

Bazel training: Build, Test and CI/CD at scale

3 days (21 hours)

Presentation

Bazel is an open-source build system for enterprises. This industrialization platform for multi-language, multi-platform applications provides tools for incremental, reproducible compiling</strong testing and packaging.

Our Bazel: Build, Test and CI/CD at scale course will enable you to structure your projects, manage dependencies and accelerate your pipelines by taking advantage of caching and remote execution.

You'll be able to get more information on build performance, configure a complete Bazel environment and define CI/CD and security best practices.

Bazel makes it easy to orchestrate tests and correlate results with pipeline metrics. What's more, Bazel adapts to both single-repository and distributed architectures, from developer workstations to CI agents.

Following this course, you'll be able to install, configure and operate Bazel, as well as understand its architecture and optimization mechanisms. Like all our training courses, this one will introduce you to the latest stable release and its new features.

Like all our training courses, this one uses the latest stable version v8.3.1 of Bazel.

Objectives

- Understand Bazel's architecture and key concepts
- Industrialize a multilingual build environment
- Reduce build times via cache and remote execution
- Automate tests and integrate Bazel into CI/CD
- Guarantee artifact portability, security and traceability

Target audience

- Software developers
- DevOps engineers
- Application architects

Prerequisites

- Command line and Git skills
- Notions of development (Java, Python, C++ or Go)
- Basic knowledge of continuous integration

Our Bazel training program: Build, Test and CI/CD at scale

Introduction to Bazel and fundamental concepts

- Understanding Bazel's architecture and positioning
- Compare with Maven, Gradle, Make and other tools
- Installation and initial configuration on your workstation
- Role of WORKSPACE and BUILD files
- Managing external dependencies and Bzlmod
- Workshop: Installation and initial configuration of a Bazel project

Project structure and dependency management

- Organization into packages, targets and polyglot monorepo
- Native rules: cc_binary, java_library, py_test...
- Introduction to Starlark for creating custom rules
- Declaration/consumption of dependencies and platform constraints
- Modularization strategies and macro reuse
- Workshop: Structuring a multi-language project with Bazel

Optimizing builds

- Incremental build principle and dependency graphs
- Local vs. remote caching: when and how
- Remote execution and large-scale parallelization
- Performance tuning (sandboxing, spawn strategies, RAM/CPU)
- Measurement and diagnostics: profiles, build event protocol
- Workshop: Setting up a remote cache and measuring gains

Test management with Bazel

- Unit, integration and end-to-end testing with test rules

- Hermeticity, reproducibility and flaky testing
- Key parameters : --test_output, --test_filter, --runs_per_test
- Integration with JUnit, pytest, Go test...
- Reports, execution times and sharding
- Workshop: Writing and executing a multi-language test suite

Bazel and CI/CD

- Integrating Bazel into Jenkins, GitHub Actions, GitLab CI
- Pipeline strategies: incremental builds, branch builds, releases
- Artifact management, shared cache and job isolation
- Cycle time and execution cost optimization
- Good reliability practices (timeouts, retry, cache keys)
- Workshop: Creating a complete CI/CD pipeline with Bazel

Cross-platform development

- Builds for Linux, macOS, Windows
- Cross?compilation and toolchains (C/C++, Java, Go, Python...)
- Native dependencies and dynamic/static links
- Docker integration and reproducible images
- Portability, compatibility and multi-agent CI
- Workshop: Building and testing a project for multiple platforms

Extension and customization

- Rules_* ecosystem and extension selection
- Writing rules in Starlark: providers, actions, attrs
- Macros and build code factorization
- Internal publication of shared rules
- Managing dependencies, versions and compatibility
- Workshop: Creating a customized Bazel rule

Security and governance

- Access policies, auditing and build traceability
- Artifact validation and signature (SBOM, attestations)
- Open-source dependency management and licensing
- Attack surface analysis and hardening
- Compliance and regulatory requirements
- Workshop: Implementing security controls in the chain

Maintenance and best practices

- Structuring a monorepo for maintainability
- Naming convention, review of BUILD files

- Performance monitoring and long-term metrics
- Maintenance automation (linters, buildifier)
- Migration from other tools (Maven/Gradle/Make)
- Workshop: Optimizing an existing Bazel project

Closure and implementation plan

- Assessment of achievements and key gas pedals
- Bazel CI/CD industrialization checklist
- .bazelrc model and reusable patterns
- Adoption roadmap and change strategy
- Resources and monitoring (releases, LTS, roadmap)
- Workshop: Drawing up your Bazel adoption plan

Companies concerned

This course is aimed at both individuals and companies, large or small, wishing to train their teams in a new advanced computer technology, or to acquire specific business knowledge or modern methods.

Positioning on entry to training

Positioning at the start of training complies with Qualiopi quality criteria. As soon as registration is finalized, the learner receives a self-assessment questionnaire which enables us to assess his or her estimated level of proficiency in different types of technology, as well as his or her expectations and personal objectives for the forthcoming training course, within the limits imposed by the selected format. This questionnaire also enables us to anticipate any connection or security difficulties within the company (intra-company or virtual classroom) which could be problematic for the follow-up and smooth running of the training session.

Teaching methods

Practical training: 60% hands-on, 40% theory. Training material distributed in digital format to all participants.

Organization

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

Validation

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

Certification

A certificate will be awarded to each trainee who completes the training course.