Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Spec-Driven Agentic Coding with Kiro: Specs, Steering, Hooks, and the CLI

Class Duration

14 hours of live training delivered over 2-3 days to accommodate your scheduling needs.

Student Prerequisites

- Professional software development experience in at least one language
- Working knowledge of Git and GitHub or GitLab
- Comfort with a VS Code-style editor and the command line
- A Kiro account on an individual plan or an organization subscription
- Experience with Amazon Q Developer is helpful for the migration topics
- Familiarity with spec-driven development is helpful; see *Spec-Driven Development with AI Coding Agents*

Target Audience

Software engineers, tech leads, and platform teams adopting Kiro as their agentic coding environment. AWS teams moving off Amazon Q Developer, and administrators who will roll Kiro out across an organization, will also benefit.

Description

Kiro is an agentic development environment from AWS, with one agent harness behind its IDE, CLI, web, and mobile apps. This course teaches Kiro's spec workflow: EARS requirements, a design document, and a task list that agents carry out. Students add steering, hooks, powers, and skills, then run Kiro CLI in the terminal and in CI. The course also covers permissions, credits, and governance. It ends with a migration plan for Amazon Q Developer, whose IDE plugins and paid subscriptions reach end of support on April 30, 2027.

Learning Outcomes

- Set up Kiro IDE and Kiro CLI with individual or organization sign-in.

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- Choose a session type for the scope and risk of the work.
- Write feature specs with EARS requirements, designs, and task lists.
- Fix defects with bugfix specs that preserve unchanged behavior.
- Check implementations against requirements with property-based tests.
- Encode project conventions with steering files and `AGENTS.md`.
- Automate quality checks with hooks.
- Extend Kiro with powers, MCP servers, skills, and custom agents.
- Run Kiro CLI headless in CI and in cloud sessions.
- Control agent autonomy with permission modes and the sandbox.
- Manage credits, models, and enterprise governance.
- Plan a team migration from Amazon Q Developer.

Training Materials

Comprehensive courseware is distributed online at the start of class. All students receive a downloadable MP4 recording of the training.

Software Requirements

Kiro IDE and Kiro CLI installed, a Kiro account (AWS Builder ID, GitHub, Google, or an organization sign-in) on a paid plan or organization subscription for the headless and cloud session topics, Git, and a GitHub or GitLab account with a repository.

Training Topics

Kiro Foundations and Setup

- Kiro IDE, CLI, Web, and Mobile
- One shared agent harness
- Kiro IDE on Code OSS
- Importing VS Code settings
- Open VSX extensions
- Signing in with `kiro-cli login`
- Builder ID and IAM Identity Center
- External identity providers
- Project and user configuration
- Configuration Sync and workspace trust

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

Default, Plan, and Spec Sessions

- The session type picker
- Default, Spec, and Plan sessions
- Bug Fix and Quick Spec sessions
- Matching the session to the work
- Plan sessions with `/plan`
- Context providers such as `#codebase`
- Checkpoints and Revert
- `/rewind` and `/tangent` in the CLI
- Compaction for long sessions

Writing and Refining Feature Specs

- Spec files in `.kiro/specs/`
- `requirements.md`, `design.md`, and `tasks.md`
- Requirements-first and design-first workflows
- Quick Spec for well-understood features
- EARS acceptance criteria
- Finding gaps with Analyze Requirements
- Architecture and error handling in designs
- Iterating before any code

Executing Tasks and Fixing Bugs with Specs

- Running tasks one at a time
- Parallel task waves
- Tracking task status
- Keeping specs in sync
- Bugfix specs and `bugfix.md`
- Current, expected, and unchanged behavior
- Root cause analysis
- Property-based tests from requirements
- Specs in the terminal with `/spec`

Steering for Project Context and Conventions

- Workspace and global steering
- `product.md`, `tech.md`, and `structure.md`
- Inclusion modes
- `fileMatch` patterns

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

- `AGENTS.md` support
- File references in steering
- Excluding files with `.kiroignore`
- Sharing team steering

Hooks for Event-Driven Automation

- JSON hook files in `.kiro/hooks/`
- Global hooks for Kiro CLI
- Creating hooks in natural language
- Prompt, agent, and session triggers
- Tool use and file triggers
- Task and compaction triggers
- `command` and `agent` hook actions
- Blocking risky tool calls
- Lint and test on save

Powers, MCP, Skills, and Custom Agents

- MCP servers in `mcp.json`
- Remote servers and OAuth
- Powers that load on demand
- The kiro.dev/powers catalog
- Agent Skills in `.kiro/skills/`
- Custom agents in `.kiro/agents/`
- Agent prompts, tools, and permissions
- Sub-agents for parallel work

Memory and Workflows

- Global and repository memories
- Memory access levels
- Reviewing and deleting memories
- Multi-agent workflows in `.kiro/workflows/`
- Built-in `investigate` and `publish-pr` workflows
- Running workflows with `/workflow run`
- Fresh context per step

Kiro CLI in the Terminal and in CI

- The V3 and V2 agent harnesses
- Upgrading agents with `/upgrade-agent`

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved

- Core slash commands
- Spawning sessions with `/spawn`
- Headless mode with `--no-interactive`
- `KIRO_API_KEY` and `--trust-tools`
- JSON Lines output
- Exit codes and `--require-mcp-startup`
- Pull request review in GitHub Actions
- Kiro in JetBrains and Zed over ACP

Supervision, Sandboxing, and Cloud Sessions

- Supervised and Accept file changes modes
- Approve for me mode
- Capability rules in `permissions.yaml`
- Workspace trust
- When to use `--trust-all-tools`
- The local command sandbox
- Sandboxing stdio MCP servers
- Cloud sessions in Agent Focus Mode
- Cloud sessions with `kiro-cli chat --cloud`
- Kiro Web and Kiro Mobile sessions
- PR feedback with `/kiro fix`

Credits, Models, and Enterprise Administration

- How credits are consumed
- Per-model credit multipliers
- Claude, GPT, and open-weight models
- Tracking usage with `/usage`
- Kiro profiles and the console
- Organization identity and credit pooling
- Model and MCP governance controls
- Enterprise Management policies
- Usage dashboards and prompt logs
- AWS CloudTrail and OpenTelemetry

Migrating Teams from Amazon Q Developer

- What reaches end of support
- Moving VS Code users to Kiro IDE

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved



To discuss this course and customizations:
Call: +1 434-509-6890 or Email: sales@cloudcontraptions.com

- JetBrains users and Kiro over ACP
- Visual Studio and Eclipse users
- Q Developer CLI to Kiro CLI
- Existing `.amazonq/` rules
- Converting rules to steering
- AWS Transform for Java and .NET
- AI-DLC workflows in Kiro
- Rollout with governance first

Copyright 2026 Cloud Contraptions LLC – All Rights Reserved