Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

Git and GitHub for Professional Teams

Class Duration

14 hours of live training delivered over 2 days.

Student Prerequisites

- Software development experience (any language)
- Basic Git exposure: clone, commit, push, pull
- Comfort working in a terminal
- A GitHub account
- No advanced Git or GitHub administration experience required

Target Audience

This course is designed for professional developers who use Git every day but have never been taught it properly. Participants typically know enough Git to get by and enough to get into trouble. It is equally valuable for teams standardizing their branching and review practices, and for tech leads who configure repository policies and review the work of both human teammates and AI coding agents.

Description

Most developers learn Git as a handful of memorized commands and a prayer. This course replaces that with a clear mental model: once you understand that Git is a content-addressed database of objects with refs pointing into it, every command stops being magic and starts being obvious. From that foundation, the course builds out the professional daily workflow — staging deliberately, writing useful commits, keeping branches current — and compares the branching strategies real teams use, including trunk-based development, GitHub Flow, and release branching, with honest guidance on when each fits.

The second day is spent where professional Git skill actually shows: pull request workflows and code review etiquette, history surgery with interactive rebase, cherry-pick, and bisect, recovering “lost” work with the reflog, and resolving merge conflicts confidently at scale. The course covers monorepo versus polyrepo tradeoffs, Git hooks, commit signing, and GitHub’s team collaboration machinery — CODEOWNERS, branch protection, and rulesets. It

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

closes with a topic that is now part of every team's reality: working alongside AI coding agents, including using worktrees to parallelize agent work and reviewing agent-authored commits with appropriate scrutiny.

Learning Outcomes

- Explain the Git object model (blobs, trees, commits, tags) and how refs and HEAD point into it
- Run a deliberate daily workflow: stage hunks selectively, write meaningful commit messages, and keep feature branches current
- Compare trunk-based development, GitHub Flow, and release branching, and select a strategy that fits team size and release cadence
- Author pull requests that are easy to review and give code review feedback that is direct, kind, and actionable
- Rewrite local history safely with interactive rebase, fixup commits, and autosquash
- Apply cherry-pick for targeted backports and bisect to find the commit that introduced a bug
- Recover deleted branches and "lost" commits using the reflog
- Resolve complex merge conflicts confidently, including rebase-versus-merge tradeoffs and rerere
- Evaluate monorepo versus polyrepo structures and the Git features that make large repositories workable
- Automate checks with Git hooks and sign commits to establish provenance
- Configure CODEOWNERS, branch protection, and repository rulesets to enforce team policy on GitHub
- Use worktrees to run multiple AI coding agents in parallel and review agent-generated commits effectively

Training Materials

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

Software Requirements

- Git 2.40 or later installed locally
- A GitHub account with ability to create repositories
- Terminal access (macOS, Linux, or Windows with Git Bash or WSL)

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

- Text editor or IDE (VS Code recommended)
- GitHub CLI (`gh`) recommended

Training Topics

The Git Mental Model

- Git as a content-addressed object database
- Blobs, trees, commits, and annotated tags
- Refs, HEAD, and detached HEAD demystified
- The three areas: working tree, index, repository
- Why understanding the model makes commands predictable

The Professional Daily Workflow

- Staging deliberately with `git add -p`
- Writing commit messages that help future readers
- Amending, stashing, and switching contexts cleanly
- Fetch versus pull, and keeping branches current
- Useful configuration, aliases, and diff tooling

Branching Strategies

- Trunk-based development with short-lived branches
- GitHub Flow and continuous delivery
- Release branches and long-term support lines
- Feature flags versus long-lived feature branches
- Choosing a strategy for your team and cadence

Pull Requests and Code Review

- Shaping small, reviewable pull requests
- PR descriptions, draft PRs, and stacked changes
- Code review etiquette: comments that land well
- Responding to review feedback without friction
- Merge strategies: merge commit, squash, rebase

History Surgery

- Interactive rebase: reorder, squash, edit, drop
- Fixup commits and autosquash workflows
- Cherry-picking for backports and hotfixes
- Hunting regressions with `git bisect`
- Recovering lost work with the reflog

Cloud Contraptions
TRAINING & CONSULTING

To discuss this course and customizations:
Call: 434-509-5680 or Email: sales@cloudcontraptions.com

Merge Conflicts at Scale

- Why conflicts happen and how to read conflict markers
- Three-way merges and choosing merge tools
- Rebase versus merge: tradeoffs in shared history
- Reusing resolutions with rerere
- Reducing conflict frequency through team habits

Monorepo vs. Polyrepo

- Tradeoffs: atomic changes, ownership, tooling cost
- Sparse checkout and partial clone for large repos
- Submodules and subtrees: when and when not
- Dependency and release management in each model

Hooks, Signing, and Provenance

- Client-side hooks and pre-commit frameworks
- Server-side policy enforcement
- Signing commits and tags (SSH and GPG signing)
- Verified commits on GitHub

GitHub for Teams

- CODEOWNERS and required reviewers
- Branch protection rules and required status checks
- Repository rulesets: branch, tag, and push rules
- Merge queues for busy main branches
- Issues, projects, and linking work to commits

Working Alongside AI Coding Agents

- Git worktrees for parallel agent sessions
- Reviewing agent-authored commits and PRs critically
- Keeping agent changes small, scoped, and revertable
- Attribution, co-author trailers, and team policy
- Branch hygiene when agents generate many branches