

**Git Fundamentals for Engineers**

**Course Number:** GIT-110
**Duration:** 2 days

**Overview**

The Git Fundamentals for Engineers training course teaches attendees how to use Git at a fundamental level and includes plenty of hands-on exercises managing repositories, creating and merging branches, properly backing out of mistakes, and getting comfortable resolving conflicts. This course gives students a solid grasp of how Git works, helping reduce the typical “fear of blowing something up” in a Git project. Students consistently complete the class with a very high level of confidence with Git.

This Git course can use the repository hosting platform of your choice, such as GitHub, GitLab, BitBucket, or Azure DevOps. Students will learn how to leverage such a platform to share changes, synchronize their branches, collaborate through pull/merge requests, review code, and other collaboration best practices.

**Prerequisites**

* No prior experience with Git is presumed.
* Prior experience with other version control systems is helpful but not required.
* Experience with the command-line or DOS command prompt is preferred.

**Materials**

All Git training attendees receive comprehensive course materials.

**Software Needed on Each Student PC**

* Git 2.x or later
* Internet access for all attendees and the instructor

**Objectives**

All students will:

* Understand Git and Git fundamentals
* Review and edit the commit history
* Improve your daily workflow
* Branch, merge, and use remote repositories
* Understand collaboration best practices as a team
* Be able to apply their Git knowledge to a pre-selected Git platform (GitHub, etc)

**Outline**

* Introduction to Source Code Management
	+ The Core Principles of Change Management
	+ The Power to Undo Changes
	+ Audit Trails and Investigations
	+ Reproducible Software
* Git Introduction and Basics
	+ Introduction to Git
	+ Trees and Commits
	+ Configuring Git
	+ Adding, Renaming, and Removing Files
* Reviewing and Editing the Commit History
	+ Reviewing the Commit History
	+ Revision Shortcuts
	+ Fixing Mistakes
* Improving Your Daily Workflow
	+ Simplifying Common Commands with Aliases
	+ Ignoring Build Artifacts
	+ Saving Changes for Later Use (Stashing)
* Branching
	+ Branching Basics
	+ Listing Differences Between Branches
	+ Visualizing Branches
	+ Deleting Branches
	+ Tagging
* Merging
	+ Merging Basics
	+ Merge Conflicts
	+ Merging Remote Branches
* Remote Repositories
	+ Remote Repositories
	+ Synchronizing Objects with Remotes
	+ Tracking Branches
	+ Remote branch management
	+ Forking and working with upstreams
	+ Rolling back changes
* Collaboration Through a Platform
	+ Introduction to GitHub/GitLab/BitBucket/Azure DevOps (choose one)
	+ Hosting Git Repositories
	+ Daily Workflow, ex: GitLab Flow or GitHub Flow
	+ Using pull/merge requests and code reviews
	+ Leveraging Forks
	+ Managing Releases
* History Management
	+ Rebasing
	+ Searching the log
	+ Cherry picking
	+ Squashing
	+ Bisecting
	+ Advanced workflows
* Workflows/Best Practices
	+ Branch strategies
	+ Remote strategies
	+ Tagging strategies
* Repairing Branches Mastering History
	+ Reset
	+ Reflog
* Conclusion