

## Course duration

- 1 day

## Course Benefits

- Learn how to use Git for efficiently managing version control in software development.
- Learn the importance of version control.
- Learn the purpose of Git.
- Learn how to work with Git and manage workflows.
- Learn how to work with history in workflows.
- Learn advanced techniques.

## Course Outline

1. Introduction to Version Control
  1. What is Version Control
  2. "Undo" Capability
  3. Collaboration
  4. Communication and Sharing
  5. Auditing and Tracking
  6. Release Engineering, Maintenance, SDLC
  7. Diagnostics
  8. History of Version Control
  9. Distributed Version Control
  10. Summary
2. Introduction to Git
  1. What is Git
  2. Git's Design Goals
  3. Branching and Merging
  4. Centralized Version Control
  5. Distributed Version Control
  6. Git Basics
  7. Getting Git
  8. Git on the Server
  9. Git Repository Managers
  10. Git on Somebody Else's Server
  11. Summary
3. Basic Git Operations
  1. Using Git
  2. Definitions
  3. Commit

4. How to Think About Commits
5. Viewing History
6. Configuring Git
7. Configuration Scope
8. User Identification
9. GPG Signing
10. Gnu Privacy Guard
11. GPG Basics
12. GPG and Git
13. .gitignore
14. Other Useful Configurations
15. Summary
4. Branching, Merging and Remotes
  1. Branching
  2. Branches in Git
  3. Merge
  4. Fast Forward Merge
  5. --no-ff
  6. More Than One Repository
  7. Working with Remotes
  8. Fetch and Pull
  9. Push
  10. Pull Requests
  11. Tagging a Commit
  12. Lightweight Tags
  13. Annotated Tags
  14. Sharing Tags
  15. Checking Out a Tag
  16. Summary
5. Git Work Flows
  1. Work Flows
  2. Local Work Flow
  3. Feature Branches
  4. Centralized Workflow
  5. Integration Manager Work Flow
  6. Other Work Flows Are Possible
  7. Summary
6. Introduction to GitFlow
  1. What is GitFlow
  2. Benefits
  3. How GitFlow works?
  4. GitFlow Extension
  5. Initializing GitFlow
  6. Features
  7. Release
  8. Hotfixes
  9. Summary

- 7. Rewriting History
  - 1. Rewriting History
  - 2. Squashing Commits
  - 3. Rebase vs Merge
  - 4. Amending Commits
  - 5. Reset
  - 6. Summary
- 8. Examining History
  - 1. Looking at History
  - 2. Log
  - 3. Blame
  - 4. Bisect
  - 5. Summary
- 9. Submodules and Subtrees
  - 1. Submodules
  - 2. Dependency Management
  - 3. Git Submodules
  - 4. Adding a Submodule
  - 5. .gitmodules
  - 6. Cloning a Repository with Submodules
  - 7. Updating Submodules (Initial)
  - 8. Updating Submodules (Ongoing)
  - 9. Subtrees
  - 10. Subtrees – How They Work
  - 11. Subtrees
  - 12. Conclusion
- 10. Configuring Git
  - 1. Advanced Configuration
  - 2. Advanced Configuration Variables
  - 3. Environment Variables
  - 4. Aliases
  - 5. Git Hooks
  - 6. Summary
- 11. Lab Exercises
  - 1. Lab 1. Starting Out with Git
  - 2. Lab 2. Branching, Merging and Working with Remotes
  - 3. Lab 3. Experimenting with Workflows
  - 4. Lab 4. Using the GitFlow Workflow
  - 5. Lab 5. Rebasing and Rewriting History
  - 6. Lab 6. Git Submodules
  - 7. Lab 7. (Optional) GitFlow Workflow With the GitFlow Extensions

## Class Materials

Each student will receive a comprehensive set of materials, including course notes and all the class examples.

### Class Prerequisites

Experience in the following *is required* for this Git class:

- Basic computer (Windows or Mac or Linux) literacy.