



Introduction to Git- 24 hours

General

Git is the de-facto standard source control system for the tech industry and is one of the most flexible software tools to be found. Any developer or devops person probably needs at least a basic understanding of Git to get going and this course is intended for that purpose.

This course covers all of the fundamental operations an experienced coder would use on a daily basis. The course begins with an introduction to Git and a comparison of Git to other version control systems. It then transitions into the nuts-and-bolts of working with Git, including everything from setting up a repository to advanced topics like branching and merging.

Because the industry sometimes misuses git this course is also focuses on clearing up muddled understanding of git: the staging area, merge vs rebase, history management, branching and more

Intended Audience

- Any software developer or devops who needs to work with Git or understand Git better.
- Any software developer who has worked with Git but wants a deeper understanding of it.
- System administrators who are moving to devops in general or Git specifically.
- Any manager who needs to understand what is possible and how to manage git using projects.

Prerequisites

- Tech affinity.

At the end of the course participants could

- Setup and use git
- Understand and use git's branching features correctly and effectively
- Decide on which workflows to use when using Git

Outline

- Introduction to Git
 - History of Git
 - Who is using Git
 - Adopting Git
- Core Git Concepts
 - Always on a branch
 - Everything is sha1 (files, changes, tags, branches)
 - Everything has a parent except first change.
 - Never store anything twice
 - sha includes all history
 - sha is unique in the world
- Git basics
 - Setting up a local repository
 - Setting up a client to a repository
 - local
 - remote
 - The staging area
 - Making your first change
 - Committing
 - Seeing history
 - Renaming, moving and removing files
- Configuring git
 - local and global config files
 - configuring git commands
 - configuring signing
 - adding aliases
 - ignoring files
- Undoing things
 - Staging area undoings
 - Undoing latest commit
 - Undoing last n commits
 - Cherry picking from latest commits
- Remote repositories
 - Working with a remote repository
 - Setting up / publishing a repository
 - Understanding the repository structure
 - Working with Multiple remotes
 - Working with GitHub

- Branches
 - Creating local branches
 - Working on local branches
 - Committing on local branches
 - Moving between local branches
 - Pruning branches
- Merging changes
 - git fetch
 - git pull
 - git rebase
 - what is fast forwarding?
 - cherry picking
 - handling conflicts
 - basics
 - using merge tools
- Merge vs Rebase
 - Which should you choose? (Rebase)
 - Why?
- Workflows
 - git does not force a workflow
 - feature branches
 - dev vs production
 - back porting changes
 - Examples of workflows
 - working on your own workflow
 - Jenkins
 - pull requests
- Getting git data
 - Git log and it's many options
 - Visual tools
 - Using programming (example is python)
- Under the hood
 - The git object store and how it works
 - What happens when you "git add"
 - What happens when you commit?
 - What happens when you create an annotated tag?
 - What happens when you branch?
- Worktrees
 - Why are they needed?
 - Creating a worktree
 - Working with worktrees
 - Pruning worktrees
- Tagging
 - Why tag?
 - difference between anotates and non annotated tags
 - pushing and pulling tags
 - Using tags in oter git commands
- Rewriting History
 - Why you should never do it
 - How to do it anyway
- Stashing
 - Why would you want stashing?
 - Creating and naming stashes
 - Apply a specific stash
 - Delete stashes

- Git hooks
 - How to set up hooks?
 - What guarantees do you get?
- External git tools
 - GitHub and BitBucket
 - Gitlab
 - Git and IDEs
 - Git and Jenkins, Bamboo etc