Git and GitHub

Working Dir/Staging, Local/Remote, Clone, Push, Pull, Branch/Merge, Monorepo, GitHub Desktop

Source code is by far the most important asset any software company owns. It is more valuable than buildings, brand names, computer hardware, furniture or anything else a software company has. Source code needs to be valued and treated like the very important company asset that it is. Hence the need for a robust source code management system.

Git is the most popular source code management system; GitHub.com is the most popular Git cloud hosting solution. Either Git alone or Git and GitHub can be used to comprehensively manage and protect source.

Even if not using GitHub for their own source, app developers still need to get familiar with it as most of today's popular open source projects are using it and app developers will invariably need to use these.

This course covers both and helps developers gain hands-on experience in how to incorporate both into their development workflow. Many Git-related terms have entered the developer lexicon – push, pull request, cloning, forking, promoting, repo – and this course helps attendees understand each concept and mentally tie everything together to see how they work in unison.

Target Audience

Software engineers and architects wishing to correctly manage valuable source trees

Prerequisites

Programming knowledge and some previous hands-on experience of any source control system.

For the GitHub part of the course, each attendee will need their own (free) GitHub login to complete the lab work.

Contents of One-Day Training Course

Distributed Version Control

- Using what you might already know
- Adding distributed influence
- Organizing teams via Git
- Strategies for managing source trees
- Terminology - push / pull, clone, fork, fetch, branching and merging
- Lifecycle of a single line of code

Getting Started With Git

- Installing and configuration
- Simple usage
- We explore where source can be stored (local and remote)
- Promoting from working dir to staging and beyond
- What happens during a commit

Working Locally

- Init vs. clone
- File system layout
- The .gitignore file
- Creating, modifying and deleting locally
- Cancel changes (revert)
- Logging/history/status

Branching / Merging

- Creating and listing branches
- Merging a branch
- Change tracking / diff / Rebase
- Feature branches vs. trunk development

Remote

- Remote protocols
- Connecting to remote repo servers
- Push and pull commands
- Fetch command

Command Line Tooling

- Porcelain vs. plumbing
- Beyond the basics - more complete look at advanced command line tools for Git

Managing as part of Toolchain

- Git as part of toolchain
- Use with other tools
- Call via scripting (automated test runs, linting, Continuous Integration/Continuous Delivery - CI/CD)
- What to do with generated info

Monorepo

- Each project need not exist in separate repo
- Multiple projects can be placed in a single repository – known as a monorepo
- Practical ideas for using monorepos

GitHub

- Source repositories in the cloud
- Public (free hosting) & private (fee-paying)
- Organizations and teams
- Interacting with open source projects (issues, releases, changelog, pull requests)

GitHub Desktop

- Enhanced (Electron-based) GUI to comprehensively manage Git repositories
- Easy to set up and use

Git Internals

- The source code for Git itself is an interesting read: https://github.com/git/git
- What can we learn from exploring it?

Project

- Organizing a large source tree using Git
- Deciding on project and repo layout