Introduction to GitHub for Developers (TTDV7551) Training

Duration: **2 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

Overview

Learn Git and GitHub workflows, version control, branching strategies, pull requests, and collaborative software development practices.

The **Introduction to GitHub for Developers** course is a fast-paced, hands-on training program designed to help developers understand the fundamentals of **Git** and **GitHub**.

This practical course introduces participants to essential concepts including**version control**, **source code management (SCM)**, **Git workflows**, **branching strategies**, **repository management**, and modern **GitHub collaboration** practices.

While GitHub is widely used for software development, it also supports documentation management, bug tracking, feature requests, project collaboration, task management, and wiki systems for teams and organizations.

Throughout the course, learners gain practical experience with:

- **Git branching**
- **Pull requests**
- **Commit management**
- **Merge workflows**
- **Repository creation**
- **GitHub Issues**
- **Markdown**
- **Git reset and revert**
- Advanced Git commands and troubleshooting

This course provides an excellent foundation for developers new to GitHub while helping teams improve collaboration and modern development workflows.

Prerequisites

No prior Git experience is required for this course. However, participants should have:

- Basic coding or programming knowledge
- Familiarity with software development concepts

Introduction to GitHub for Developers (TTDV7551) Training

Duration: **2 Day**

BİLGİNÇ IT Academy
info@bilginc.com
bilginc.com

Who Should Attend

This course is ideal for:

- **New software developers**
- **New GitHub users**
- **Programming students**
- Technical professionals joining development teams
- Anyone interested in learning Git and version control systems

What You Will Learn

Participants will learn how to:

- Understand **Git and GitHub fundamentals**
- Manage repositories effectively
- Use **Git branching workflows**
- Work with local and remote repositories
- Create and merge pull requests
- Fix commit mistakes and manage history
- Optimize Git workflows
- Use GitHub collaboration tools
- Apply version control best practices

Outline

1. Getting Started with Collaboration

- What is GitHub?
- The GitHub ecosystem
- What is Git?
- Exploring GitHub repositories
- Using GitHub Issues
- Markdown basics
- Installing Git and setting up GitHub
- Key terminology:
  - Repository
  - Commit
  - Branch
  - Merge
  - Pull Request
- Understanding GitHub methodology

2. Local Git Configuration

- Checking Git version
- Git configuration levels

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Configuring username and email
- Configuring autocrlf
- Setting default editor and merge tools
- Global vs local configuration
- Using .gitconfig

3. Working Locally with Git

- Creating local repositories
- Using git status
- Working with branches locally
- Switching branches
- Creating files and commits
- The two-stage commit process
- Repository initialization
- Staging and committing changes
- Using .gitignore and .gitattributes

4. Searching for Events in Your Code

- Using git bisect
- Finding bugs in projects
- Using git grep and git log
- Tracking changes to files
- Identifying bug introductions

5. Reverting Commits

- Understanding commits
- Safe Git operations
- Using git revert
- Comparing git revert vs git reset
- Undoing commits safely
- Reverting merges

6. Helpful Git Commands

- Moving and renaming files
- Managing staged changes
- Essential daily Git commands
- Advanced Git commands
- Troubleshooting with:
  - git fsck
  - git reflog
  - git stash

7. Viewing Local Changes

- Comparing repository changes
- Using git diff

BİLGİNÇ IT Academy

info@bilginc.com

bilginc.com

- Using git blame
- Tracking staged and unstaged changes
- Visual Git tools

8. Creating a New Local Repository

- Using git init
- Creating local repositories
- Connecting to remote repositories
- Using git remote add

9. Fixing Commit Mistakes

- Revising commits
- Amending commits
- Changing commit messages

10. Rewriting History with Git Reset

- Understanding Git reset
- Reset modes:
  - Soft Reset
  - Mixed Reset
  - Hard Reset
- Recovering lost commits
- Using git reflog
- Best practices and cautions

11. Collaborating on Your Code

- Pushing code to GitHub
- Creating pull requests
- Performing code reviews
- Using GitHub Discussions
- Release tagging
- Managing milestones