Developing with Git and GitHub

Duration: 3 Days Course Code: GK840250 Delivery Method: Virtual Learning

Overview:

Gain expertise in Git and GitHub to streamline your development workflow and enhance team collaboration.

Developing with Git and GitHub is designed to provide software developers, DevOps engineers, cybersecurity specialists, technical project managers, and data scientists with a comprehensive understanding of Git and GitHub. This intermediate-level course spans three days and offers a blend of theoretical knowledge and hands-on practice, ensuring participants gain practical skills in version control and collaborative development.
Throughout the class, participants will learn to define Git's architecture, execute foundational Git operations, and implement effective remote repository operations. You will also explore advanced GitHub features such as project management tools, security best practices, and CI/CD pipelines using GitHub Actions. By the end of the course, attendees will be proficient in leveraging GitHub's capabilities to enhance team collaboration, manage code changes, and streamline development workflows.
This course is ideal for professionals looking to deepen their understanding of version control systems and improve their collaborative development skills. Participants will leave with the ability to set up and utilize GitHub's project management tools, design effective CI/CD pipelines, and develop custom development environments with GitHub Codespaces. Whether you're aiming to enhance your team's productivity or advance your career, this course provides the essential knowledge and skills needed to succeed in today's fast-paced development environment.

Virtual Learning

This interactive training can be taken from any location, your office or home and is delivered by a trainer. This training does not have any delegates in the class with the instructor, since all delegates are virtually connected. Virtual delegates do not travel to this course, Global Knowledge will send you all the information needed before the start of the course and you can test the logins.

Target Audience:

- Software developers
- DevOps engineers
- Cybersecurity specialists
- Technical project managers
- Data scientists

Objectives:

- **After completing this course you should be able to:**
- Define Git's architecture including Working Directory, Staging Area, and Repository
- Explain how Git's backtracking and recovery mechanisms work for code safety
- Describe GitHub's role in enabling team collaboration through remote repositories
- Execute foundational Git operations including staging, committing, and managing branches
- Implement effective remote repository operations including fetch, pull, and push
- Set up and utilize GitHub's project management tools including Issues and Project Boards
- Examine repository histories to track and understand code changes over time
- Investigate and resolve merge conflicts in collaborative environments
- Compare different security approaches including SSH keys and two-factor authentication
- Design effective CI/CD pipelines using GitHub Actions
- Develop custom development environments with GitHub Codespaces
- Construct efficient coding workflows leveraging GitHub Copilot's AI capabilities

Prerequisites:

Course Level: INTERMEDIATE

6-12 months of software development experience

Basic proficiency with command line operations, including:

- Navigating directories (cd, ls/dir)
- Creating/editing files and folders
- Running basic commands
- Understanding of file paths

Familiarity with basic collaborative development concepts:

- Code versioning
- Team-based development workflows
- Code review processes
- GK840203 - Introduction to Programming

Testing and Certification

Recommended as preapartion for the following exams:

- This course is not aligned to any specific exam.

Content:

Git Fundamentals

- Introduction to Version Control and Git
- What is Git?
- Git vs Other Version Control Systems
- Git Architecture Overview
- Working Directory, Staging Area, and Repository
- Basic Git Workflow
- Git Configuration and Setup (git config)
- Git Operations
- Repository Initialization (git init)
- Staging Files (git add, git status)
- Creating Meaningful Commits (git commit)
- Viewing and Understanding History (git log)
- Understanding HEAD
- Best Practices for Commits
- .gitignore Files
- Backtracking and Recovery
- Git Reset Types (git reset)
- Git Checkout (git checkout)
- Reverting Changes
- Managing the Staging Area (git diff)
- Temporary Storage with Stash (git stash)
- Recovery Strategies
- Git Reflog
- Branching and Merging
- Branch Concept and Purpose
- Creating and Managing Branches (git branch)
- Branch Naming Conventions
- Merging Fundamentals (git merge)
- Handling Merge Conflicts

GitHub Fundamentals and Collaboration

- Introduction to GitHub
- What is GitHub?
- Creating and Setting Up Account
- GitHub vs Git
- Repository Creation and Settings
- GitHub Interface Overview
- Repository Templates
- Remote Operations
- Connecting Local to Remote (git remote)
- Remote Repository Management
- Cloning Repositories (git clone)
- Fetch vs Pull (git fetch, git pull)
- Push Operations (git push)
- Tracking Branches
- File Management Commands (git rm, git mv)
- Collaborative Workflows
- Understanding Fork vs Clone
- Pull Requests
- Code Review Process
- Branch Protection Rules
- Contributing Guidelines
- Merge Strategies
- Resolving Conflicts in Pull Requests
- GitHub Project Management
- Issues and Milestones
- Project Boards
- Markdown Documentation
- Wiki Pages
- README Best Practices

Advanced GitHub Features

- GitHub Security (1.5 hours)
- Personal Access Tokens
- SSH Keys Setup
- Two-Factor Authentication
- Repository Security Settings
- Access Management
- Security Best Practices
- GitHub Actions
- CI/CD Concepts
- Understanding Workflows
- Creating Custom Actions
- Workflow Triggers
- Environment Variables and Secrets
- Common Use Cases
- Testing and Deployment
- Advanced History Commands (git rebase, git show)
- GitHub Codespaces
- Development Environments
- Customizing Codespaces
- github.dev Overview
- Performance Considerations
- Cost Management
- Best Practices
- GitHub Copilot
- AI-Assisted Development
- Setting Up Copilot
- Effective Prompting
- Code Suggestions and Completions
- Best Practices and Limitations
- Security Considerations

Further Information:

For More information, or to book your course, please call us on 0800/84.009

info@globalknowledge.be

www.globalknowledge.com/en-be/