

**DevOps Foundations for Java with Git, Jenkins, and Maven**

**Course Number:** JEN-114WA
**Duration:** 3 days

**Overview**

This in-person or onsite DevOps Foundations for Java with Git, Jenkins, and Maven training course teaches attendees fundamental DevOps concepts, how version control with Git fits into broader DevOps practices, and how to build Java projects with Maven. Participants learn Continuous Integration (CI) techniques, including integrating Git with Jenkins and using SonarQube for continuous code quality measurement and enforcement.

**Prerequisites**

Basic Java knowledge is beneficial.

**Materials**

All DevOps training students receive comprehensive courseware.

**Software Needed on Each Student PC**

Attendees will not need to install any software on their computer for this class. The class will be conducted in a remote environment that Accelebrate will provide; students will only need a local computer with a web browser and a stable Internet connection. Any recent version of Microsoft Edge, Mozilla Firefox, or Google Chrome will be fine.

**Objectives**

* Understand what DevOps is
* Implement version control with Git
* Build Java projects with Maven
* Implement continuous integration with Jenkins
* Integrate Git with Jenkins
* Implement continuous code quality with SonarQube

**Outline**

* DevOps Fundamentals
	+ Why DevOps
	+ What is DevOps?
	+ Collaborative, Matrixed, and Cross-Functional Teams
	+ Key Components of Successful DevOps Teams
	+ DevOps-ification
	+ DevOps Vocabulary
	+ DevOps Goals
	+ Not DevOps - Crush Buzzwords
	+ Driving Business Outcomes with DevOps
	+ Technology-Enabled Business
	+ DevOps Key Enabler for Digital Transformation
	+ Core Values and Mission
	+ Communication
	+ Collaboration
	+ Value Stream Mapping
	+ Behavioral Patterns for Success
	+ DevOps Org Structures
	+ DevOps Team - Separate
	+ DevOps Merged Organization
	+ DevOps Overlapped Organization
	+ Organizational Structure Leadership
	+ What Does Continuous Delivery Mean?
	+ Deployment Pipelines
	+ Your Organization is Doing CD if …
	+ Pipelining for CD
	+ Continuous Integration
	+ CI Pipeline
	+ CD & CI Methodologies
	+ Key Tool Categories for CI/CD
* Introduction to Git
	+ What is Git
	+ Git's Design Goals
	+ Branching and Merging
	+ Centralized Version Control
	+ Distributed Version Control
	+ Git Basics
	+ Getting Git
	+ Git on the Server
	+ Git Repository Managers
	+ Git on Somebody Else's Server
* Basic Git Operations
	+ Using Git
	+ Definitions
	+ Commit
	+ How to Think About Commits
	+ Viewing History
	+ Configuring Git
	+ Configuration Scope
	+ User Identification
	+ GPG Signing
	+ Gnu Privacy Guard
	+ GPG Basics
	+ GPG and Git
	+ .gitignore
	+ Other Useful Configurations
* Branching, Merging, and Remotes
	+ Branching
	+ Branches in Git
	+ Merge
	+ Fast Forward Merge
	+ --no-ff
	+ More Than One Repository
	+ Working with Remotes
	+ Fetch and Pull
	+ Push
	+ Pull Requests
	+ Tagging a Commit
	+ Lightweight Tags
	+ Annotated Tags
	+ Sharing Tags
	+ Checking Out a Tag
* Introduction to Git Flow
	+ Why Use an SCM Workflow?
	+ What is Git Flow
	+ Benefits
	+ How does Git Flow work?
	+ Git Flow Extension
	+ Initializing Git Flow
	+ Features
	+ Release
	+ Hotfixes
	+ Git Flow and Continuous Integration
	+ Git Flow Alternatives
	+ Trunk-based Development
	+ GitHub Flow
* Continuous Code Quality
	+ Continuous Code Quality
	+ What is SonarQube
	+ SonarQube - Benefits
	+ SonarQube (Multilingual)
	+ Seven Axes of Quality
	+ Potential Bugs
	+ Tests
	+ Comments and Duplication
	+ Architecture and Design
	+ Complexity
	+ SonarQube Installation
	+ SonarQube Components
	+ Code Quality
* Introduction to Continuous Integration, Continuous Delivery, and Jenkins-CI
	+ Foundation of Agile AppDev
	+ XP Flow
	+ Extreme Programming
	+ Agile Development
	+ What is Continuous Integration
	+ Typical Setup for Continuous Integration
	+ Setup Notes for Continuous Integration
	+ CI with Artifact Management
	+ What is Continuous Delivery?
	+ Why Continuous Delivery?
	+ DevOps and Continuous Delivery
	+ Continuous Delivery Challenges
	+ Continuous Delivery vs. Continuous Deployment
	+ Jenkins Continuous Integration
	+ Jenkins Features
	+ Running Jenkins
* Introduction to Apache Maven
	+ Build Tools for Java
	+ History of Build Tools
	+ Traditional Scripting
	+ 'make'
	+ Problems with Make
	+ Manual Build with JavaC
	+ ANT
	+ Pros and Cons of Ant
	+ Apache Maven
	+ Goals of Maven
	+ What is Apache Maven?
	+ Why Use Apache Maven?
	+ The Maven EcoSystem
	+ Consistent Easy-to-Understand Project Layout
	+ Convention Over Configuration
	+ Maven is Different
	+ Maven Projects have a Standardized Build
	+ Effect of Convention Over Configuration
	+ Importance of Plugins
	+ A Key Point on Maven!
	+ Key Features of Maven
* Installing and Running Apache Maven
	+ Downloading Maven
	+ Installing Maven
	+ Run From Command Line
	+ Running Inside an IDE
	+ Settings.xml
	+ Local Repository
* Installing and Running Jenkins
	+ Downloading and Installing Jenkins
	+ Running Jenkins as a Stand-Alone Application
	+ Running Jenkins on an Application Server
	+ The Jenkins Home Folder
	+ Installing Jenkins as a Windows Service
	+ Initial Configuration
	+ Configuration Wizard
	+ Configuring Tools
	+ Configuring Tools - Best Practices
	+ Logging in Jenkins
	+ Custom Log Recorders
* Job Types in Jenkins
	+ Different types of Jenkins Items
	+ Configuring Source Code Management(SCM)
	+ Working with Subversion
	+ Working with Git
	+ Storing Credentials
	+ Service Accounts
	+ Build Triggers
	+ Schedule Build Jobs
	+ Polling the SCM
	+ Polling vs. Triggers
	+ Maven Build Steps
* Getting Started With Maven
	+ Terminology and Basic Concepts
	+ Artifacts
	+ Lifecycle
	+ Default Lifecycle
	+ Plugins
	+ Running Maven - the Story So Far
	+ Running Maven from an IDE
	+ Common Goals
	+ pom.xml
	+ Artifact Coordinates
	+ Standard Layout for Sources
* A Web Application in Maven
	+ A More Complex Project
	+ Putting it Together With Maven
	+ Packaging the Target Artifact
	+ The Source Tree
	+ Dependencies
	+ Transitive Dependencies
	+ Dependency Scope
	+ Working With Servers
	+ Declaring and Configuring Plugins
	+ Running the Plugin
	+ Binding a Plugin Goal to the Lifecycle
	+ Archetypes
* Commonly Used Plugins
	+ Maven Plugins
	+ Declaring and Configuring Plugins
	+ Running the Plugin
	+ Binding a Plugin Goal to the Lifecycle
	+ Maven Surefire Test Plugin
	+ Failsafe Plugin
	+ Site Plugin
	+ JavaDoc Plugin
	+ PMD Plugin
	+ Code Coverage – Cobertura
* Creating Archetypes
	+ Introduction to Maven Archetypes
	+ Using Interactive Mode to generate Goal
	+ Common Maven Archetypes
* Repository Management
	+ Maven's Approach to Artifacts
	+ Publishing Artifacts
	+ Summary of Maven's Artifact Handling
	+ Repository
	+ Repository Manager
	+ Proxy Remote Repositories
	+ Types of Artifacts
	+ Release Artifacts
	+ Snapshot Artifacts
	+ Reasons to Use a Repository Manager
	+ Repository Coordinates
	+ Addressing Resources in a Repository
* Release Management
	+ What is release Management?
	+ Release Management with Nexus
	+ Release Management with Maven
* Jenkins Plugins
	+ SCM
	+ Build and Test
	+ Analyzers
	+ Jenkins for Teams
	+ Installing Jenkins Plugins
* Securing Jenkins
	+ Jenkins Security
	+ Authentication
	+ Authorization
	+ Confidentiality
	+ Activating Security
	+ Configure Authentication
	+ Using Jenkins's Internal User Database
	+ Creating Users
	+ Authorization
	+ Matrix-Based Security
	+ Note – Create the Administrative User
	+ Project-based Matrix Authorization
	+ Project-Based Authentication
	+ Role-Based Access Control
* Distributed Builds with Jenkins
	+ Distributed Builds - Overview
	+ Distributed Builds – How?
	+ Agent Machines
	+ Configure Jenkins Master
	+ Configure Projects
* Continuous Delivery and the Jenkins Pipeline
	+ Continuous Delivery
	+ DevOps and Continuous Delivery
	+ Continuous Delivery Challenges
	+ Continuous Delivery with Jenkins
	+ The Pipeline Plugin
	+ Defining a Pipeline
	+ A Pipeline Example
	+ Parallel Execution
	+ Creating a Pipeline
	+ Invoking the Pipeline
	+ Interacting with the Pipeline
	+ Pipeline vs. Traditional Jobs
* Best Practices for Jenkins
	+ Secure Jenkins
	+ Users
	+ Backups
	+ Reproducible Builds
	+ Testing and Reports
	+ Large Systems
	+ Distributed Jenkins
* Conclusion