Course duration

- 3 days

Course Benefits

- Learn what DevOps is.
- Learn to implement continuous integration.
- Use version control and integrate it with continuous integration tools
- Learn configuration management and infrastructure-as-code.
- Learn automation with shell scripting and other scripting languages.
- Learn to implement continuous monitoring.
- Learn to implement continuous quality.
- Learn about containerization.

Course Outline

1. What is DevOps
    1. Dev and Ops Views
    2. Leading By Example ...
    3. What is DevOps?
    4. More DevOps Definitions
    5. DevOps and Software Delivery Life Cycle
    6. Main DevOps' Objectives
    7. The Term "DevOps" is Evolving!
    8. Infrastructure as Code
    9. Agile IT in the Cloud
    10. DevOps on the Cloud
    11. Prerequisites for DevOps Success
    12. Alignment with the Business Needs
    13. Collaborative Development
    14. Continuous Testing and Integration
    15. Continuous Release and Deployment
    16. Continuous Application Monitoring
    17. Benefits of DevOps
    18. What is Involved in DevOps
    19. Summary
2. Configuration Management
    1. What is Chef?
    2. Benefits of Infrastructure-as-Code
    3. Chef - Sample Usages
    4. Deployment / License

5. Who uses Chef
6. Chef Architecture
7. Chef Components
8. Workstation
9. Recipe
10. Cookbook
11. Ruby
12. Knife
13. Node
14. Chef-client
15. Chef Server
16. Chef Analytics
17. Chef Supermarket
18. Salient Features of Chef
19. Supported Platforms
20. Chef Components
21. Chef Server prerequisites
22. Install Configuration Scenarios
23. Standalone Installation
24. Installing Optional Chef Server Components
25. Workstation
26. Chef DK
27. Chef DK Prerequisites
28. Chef Repository
29. Installing Chef DK
30. Ohai
31. Ohai Attributes
32. Cookbooks
33. Components of a Cookbook
34. Metadata
35. Recipes
36. Resources
37. Directory Resource
38. Package Resource
39. Service Resource
40. File Resource
41. Script Resource
42. User Resource
43. Additional Chef Advanced Features
44. Summary

3. Distributed Version Control
    1. What is Version Control
    2. History of Version Control
    3. "Undo" Capability
    4. Collaboration
    5. Communication and Sharing
    6. Auditing and Tracking

7. Release Engineering, Maintenance, SDLC
8. Diagnostics
9. Distributed Version Control
10. Integrating Version Control into Jenkins
11. What is Git
12. Git's Design Goals
13. Branching and Merging
14. Centralized Version Control
15. Distributed Version Control
16. Git Basics
17. Getting Git
18. Git on the Server
19. Git Repository Managers
20. Git on Somebody Else's Server
21. Using Git
22. Definitions
23. Commit
24. Commit (continued)
25. How to Think About Commits
26. Viewing History
27. Configuring Git
28. Configuration Scope
29. User Identification
30. GPG Signing
31. Gnu Privacy Guard
32. GPG Basics
33. GPG and Git
34. .gitignore
35. Other Useful Configurations
36. Summary

4. Enterprise Version Control
    1. SVN
    2. SVN vs CVS
    3. SVN Installation
    4. SVN Life Cycle
    5. Some Useful Commands
    6. Some Useful Commands (Contd.)
    7. Perforce
    8. Important Perforce Terms
    9. Perforce Clients
    10. Mercurial
    11. Installation
    12. Some Useful Commands
    13. Some Useful Commands (Contd.)
    14. Team Foundation Version Control
    15. TFVC Workspaces
    16. TFVC Capablities

17. Atomic Check-In
18. Check-In Policies
19. Shelving
20. Team Visibility
21. Locks
22. Labeling
23. Branching
24. Branch Visualization and Tracking
25. Cross-Platform Support
26. Disconnected Work
27. Summary

5. Continuous Integration and Delivery Tools, Technology and Process
    1. What is Continuous Integration
    2. Integration Tools
    3. Typical Setup for Continuous Integration
    4. Jenkins Continuous Integration
    5. Jenkins Features
    6. Running Jenkins
    7. Jenkins Integration with various Version Control Solutions
    8. Jenkins Job
    9. Apache Maven
    10. Goals of Maven
    11. What is Apache Maven?
    12. Why Use Apache Maven?
    13. The Maven EcoSystem
    14. Consistent Easy-to-Understand Project Layout
    15. Convention Over Configuration
    16. Maven is Different
    17. Maven Projects have a Standardized Build
    18. Effect of Convention Over Configuration
    19. Importance of Plugins
    20. A Key Point on Maven!
    21. Summary
6. Continuous Code Quality
    1. Continuous Code Quality
    2. What is SonarQube
    3. SonarQube - Benefits
    4. SonarQube (Multilingual)
    5. Seven Axes of Quality
    6. Potential Bugs
    7. Tests
    8. Comments and Duplication
    9. Architecture and Design
    10. Complexity
    11. SonarQube Installation
    12. SonarQube Components
    13. Code Quality (LOC, Code Smells)

14. Code Quality (Project Files)
15. Code Quality (Code)
16. Summary

7. Automation - Scripting
    1. Why Automate
    2. When to Automate
    3. Goals for Scripting
    4. Error Handling
    5. Logging
    6. Automating Versioned Builds
    7. Automating Deployment
    8. Automating Continuous Integration Tests
    9. Automated Cleanup
    10. Introduction to Shell Scripts
    11. Basic Shell Script
    12. Return Status
    13. Variables
    14. Special Variables
    15. Arrays
    16. Operators
    17. Conditional Statements
    18. Conditional Statements (contd.)
    19. Loops
    20. Loops - while
    21. Loops - for
    22. Loops - until
    23. Loops - select
    24. Summary
8. Monitoring
    1. What is Continuous Monitoring
    2. Monitoring Tools
    3. Dynatrace Application Monitoring
    4. Dynatrace Application Monitoring (contd.)
    5. Dynatrace Application Monitoring
    6. Splunk
    7. Splunk Functionalities
    8. Splunk Searching
    9. Splunk Functions
    10. Nagios
    11. Nagios (contd.)
    12. Nagios - Installation
    13. Nagios - Hosts
    14. Nagios - Web User Interface (Hosts)
    15. Nagios - Monitoring Services
    16. Monitoring HTTP
    17. Monitoring FTP
    18. Monitoring SSH

19. Monitoring SMTP
20. Monitoring POP3
21. Monitoring IMAP
22. Summary

9. Containerization
    1. Containerization (Virtualization)
    2. Hypervisors
    3. Hypervisor Types
    4. Type 1 hypervisors
    5. Type 2 hypervisors
    6. Type 1 vs Type 2 Processing
    7. Paravirtualization
    8. Virtualization Qualities (1/2)
    9. Virtualization Qualities (2/2)
    10. Disadvantages of Virtualization
    11. Containerization
    12. Virtualization vs Containerization
    13. Where to Use Virtualization and Containerization
    14. Popular Containerization Systems
    15. What are Linux Containers
    16. Docker
    17. OpenVZ
    18. Solaris Zones (Containers)
    19. What is Docker
    20. Where Can I Ran Docker?
    21. Docker and Containerization on Linux
    22. Linux Kernel Features: cgroups and namespaces
    23. The Docker-Linux Kernel Interfaces
    24. Docker Containers vs Traditional Virtualization
    25. Docker as Platform-as-a-Service
    26. Docker Integration
    27. Docker Services
    28. Docker Application Container Public Repository
    29. Competing Systems
    30. Docker Command-line
    31. Starting, Inspecting, and Stopping Docker Containers
    32. Docker Benefits
    33. Summary
10. Collaboration
    1. What is JIRA?
    2. License
    3. JIRA Technical Specifications
    4. Issues
    5. Who uses JIRA
    6. JIRA Products
    7. JIRA Core
    8. JIRA Software

9. JIRA Service Desk
10. What a typical project involves?
11. JIRA Integration
12. Integrating JIRA into Jenkins
13. Summary

11. DevOps - The Journey
    1. Agile Development
    2. Typical Setup for Continuous Integration
    3. DevOps in the Enterprise
    4. Scaling DevOps
    5. Scaling DevOps (Organization Structure)
    6. Scaling DevOps (Locality)
    7. Scaling DevOps (Team Flexiblity)
    8. Scaling DevOps (Teams: Hiring as Scaling)
    9. Scaling DevOps (Teams: Employee Retention)
    10. DevOps Myths
    11. DevOps Anti-Patterns (Blame Culture)
    12. DevOps Anti-Patterns (Silos)
    13. DevOps Anti-Patterns (Root Cause Analysis)
    14. DevOps Anti-Patterns (Human Error)
    15. DevOps Patterns For Success
    16. DevOps Patterns For Success (Cloud)
    17. DevOps Patterns For Success (Automation)
    18. DevOps Patterns For Success (Culture)
    19. Summary

Class Materials

Each student will receive a comprehensive set of materials, including course notes and all the class examples.

Class Prerequisites

Experience in the following *is required* for this DevOps class:

- Foundational knowledge of the software delivery problem domain.

Experience in the following *would be useful* for this DevOps class:

- Some knowledge of executing Linux shell commands.