

JBoss Clustering

This training focuses on high availability services of JBoss Enterprise Middleware System (JEMS). You will learn how JBoss Application Server leverages JGroups and JBoss Cache for replication and fail-over, how to configure, tune and implement JGroups protocol stacks, how to leverage JBoss Cache in your own middleware application implementation and how to use and configure mod_jk for HTTP load balancing. We will also cover details of JBoss Application Server high availability services such as HA-JNDI, HA-JMS, HA-singleton and datasource failover.

This course is targeted for people who are or wish to become experts on JEMS high availability technologies.

Course Duration: This is a 4 day course.

Course Format: The structure of the course is 60% theory and 40% hands-on lab exercises.

Course Prerequisites

Completion of the JBoss for Advanced J2EE Developers is strongly recommended. It is strongly recommended that the student has at minimum 18 month practical development experience using J2EE and other Java middleware technologies, and it is suggested that the student has some practical experience with JBoss Application Server. Solid Java programming experience (minimum 3 years) is required and understanding of basic TCP/IP topics is necessary.

The student must have the following skills:

- JTA, Transactions, Java concurrency
- EJB 2.1, JMS, reliable messaging technologies
- Previous experience with Apache httpd and some exposure to mod_jk and/or mod_proxy
- Familiar with JBoss AS microkernel and JMX.
- Familiarity with TCP/IP, UDP, Multicasting

Course Modules

- 1. JBoss State of the Union.** Introduction to Professional Open Source methodology and to JBoss Inc.'s role in leading the innovation of Open Source middleware development. Short introduction to JBoss Inc. service offerings, current product roadmaps, JBoss Enterprise Middleware System (JEMS) and the road ahead.
- 2. Overview to JEMS High Availability Services.** This module gives you an overview of the JBoss AS platform, how it integrates different JEMS products and where high availability features should be considered for mission-critical deployments. We give an architectural overview to which services benefit from replication, load balancing and fail-over and where we JBoss AS utilizes JEMS products such as JBoss Cache, JGroups and mod_jk.
- 3. Reliable Multicasting with JGroups.** Introduction to JGroups group communication protocol. JGroups is the underlying network level library utilized by most JEMS product to achieve high availability. This module gives you an overview into what JGroups is, and how to use the JGroups API. We introduce you to the concepts of JGroups channels, groups, views, events and messages.
- 4. JGroups Protocols.** This module gives a detailed description of different network protocols available in JGroups. We discuss different protocol implementations for reliable network transport, discovery, group membership, failure detection, message ordering, security and state transfer.
- 5. Protocol Stacks and Implementation.** In this module we discuss the JGroups protocol stack architecture, how different protocols may be assembled together, how they are implemented and finally, how to add your own protocol implementation into JGroups protocol stack.
- 6. JGroups Building Blocks and Troubleshooting.** JGroups building blocks are higher level implementations on top of JGroups protocol stack that help you implement common tasks using JGroups library. In this module we introduce you to the available default implementations, e.g. an RPC invocation implementation across replicated objects using JGroups. We will also look at common issues to troubleshoot when configuring and setting up your network for JGroups.

7. **JBoss Cache Overview and API.** This module introduces you to JBoss Cache – architecture overview, features and API. You will learn how to manage the tree structure of JBoss Cache, creating and removing nodes and modifying data in the cache. JBoss Cache builds on top of the JGroups library and is a key technology in implementing high availability services in JBoss AS.
8. **Cache Loaders and Eviction Policies.** Cache loading and eviction policies are critically important to understand how to manage the life cycle of cached data with JBoss Cache. In this section we cover cache loaders which mandate under what policies data is moved from persistent stores into memory cache. Eviction policies allow you to configure and manage the data while in the cache and decide under what conditions data should be evicted back to your datastore.
9. **Cache Replication, Transactions and Isolation Levels.** This module gets into details on different usage scenarios with JBoss Cache. It will help you to understand the different replication modes with JBoss Cache – asynchronous and synchronous replication – and the implication of replication mode to application performance. We will also learn how to use transactional access to the JBoss Cache, how to set different isolation levels for the cached data, and how the transaction and isolation levels affect the locking performed by JBoss Cache.
10. **JBoss Cache AOP Overview and API.** In this section we look at an advanced version of JBoss Cache that utilizes the JBoss AOP framework for efficient implementation of data replication. We will see how the AOP version of JBoss Cache differs in architecture, the changes in the cache API and how JBoss Cache AOP manages Java objects inserted into cache differently from JBoss Cache.
11. **JBoss Cache AOP Implementation.** This module goes into deeper detail on JBoss Cache implementation. We see how JBoss AOP is leveraged to dynamically introduce interceptors to cache objects, how we can transparently keep track of the exact data changes in AOP Cache, how complex Java objects are automatically mapped into AOP cache, the use of collections in cached data sets and more.
12. **Web Tier Load Balancing and Failover.** In this module we move up one abstraction level and start looking at how high availability features are implemented at JBoss Middleware and J2EE component level. We start with HTTP session replication and see how JBoss Cache is used to implement it. We continue by looking at mod_jk features and how to implement load balancing, hot

stand-by and domain clustering when combining Tomcat servlet container with native web servers.

- 13. EJB Load Balancing and Failover.** This module looks into EJB components, recap on the proxy architecture and how we can leverage services on top of JGroups to implement load balancing and failover. We will also learn how to customize and implement your own load balancing policies with EJBs. We will also cover in more detail the naming implementation used with EJBs and how it implements high-availability.
- 14. JEMS Clustered Services.** We look at how to easily replicate service implementations (singletons) in a clustered JBoss AS environment. We discuss master-slave scenarios and how fail-over works with clustered MBeans. We will also study how to enable datasource failover with JBoss AS and discover the implementation details behind HA-JNDI and HA-RMI.
- 15. JMS Failover and Load Balanced MDB.** In this final section, we build upon the HA singleton implementation and see how the current JMS implementation in JBoss AS leverages the HA singleton framework to build fail-over capabilities for JBoss MQ. We will also have a brief introduction to the upcoming JBoss Messaging core and how it uses JGroups to build a peer network of messaging servers.