Oracle Coherence 12c: Share and Manage Data in Clusters Ed 1.1

JAVA

DURATION	MODULES	COURSE CODE
5 Days	**2 Lectures**	**—**

Course Overview

This Oracle Coherence 12c: Share and Manage Data in Clusters training will teach you about Coherence and Coherence development. Oracle Coherence is an in-memory data caching and event engine often referred to as an in-memory data grid solution, designed to seamlessly improve performance, reliability and fault tolerance of Java, .NET and C++ applications.

What You Will Learn

Course Introduction

- Audience
- Class Introductions
- Goal
- Course Objectives
- Prerequisites
- Course Environment
- Course Conventions
- Course Schedule
- How Can I Learn More?

Introduction to Coherence

- Objectives
- Agenda
- System Performance Degradation
- When Performance Problems Are Not Performance Problems
- When Performance Problems Are Capacity Problems
- Quantifying Capacity
- Scalability
- Scaling: Example
- Origins of Performance Problems
- Cache: Definition

- Scaling Example Revisited
- Caching to Scale
- Notes on Caching
- Scalability Strategy Comparison
- Scaling Example Revisited: Caching and Scaling Horizontally
- Agenda
- Welcome to Coherence
- Coherence: Definition
- Coherence Principles
- Coherence Features
- Coherence Cluster Node
- Coherence Cluster
- Coherence Data Grids and Fault Tolerance
- Coherence Cache Topology: Examples
- Elastic Data
- Event and Parallel Processing
- Live Event Model
- Queries and Filters
- Query Optimization: Explain Plans
- Coherence Transactions
- WebLogic Managed Coherence Servers
- Coherence Application Grid Archives
- Eclipse and OEPE
- Coherence Security
- Coherence Management
- Coherence*Extend
- Coherence*Web
- Representational State Transfer
- Coherence Data Grid Solution Set
- Coherence Standard Edition
- Coherence Enterprise Edition
- Coherence Grid Edition
- Oracle Fusion Middleware
- Summary
- Practice 2-1 Overview: Reviewing the Retail Domain Object Model

Getting Started with Oracle Coherence

- Objectives
- Agenda
- Installing Oracle Coherence Server
- Universal Installer
- Coherence Directory Structure
- Other Directories
- Coherence Script Basics
- Starting Coherence

- Understanding Startup
- Coherence Configuration
- Default Configuration
- Command-Line Properties
- Coherence Port Use
- Coherence Services
- Service Types
- Coherence Start Sequence
- Example: tangosol-coherence-override.xml
- Quiz
- Agenda
- Coherence Programming Model
- Named Caches
- Configuring Caches
- Minimal Cache: Example
- Coherence Console
- Console Commands
- Agenda
- Clustering Concerns
- Single-Server Cluster: Configuration
- Single-Computer Cluster: Command Line
- Coherence and TCMP
- Coherence and Logging
- Stopping Members
- Shutdown Behavior
- Practice 3-1 Overview: Installing, Configuring, and Starting a
- Coherence Cluster
- Practice 3-2 Overview: Using the Coherence Console
- Agenda
- Coherence Application Basics
- Coherence Application: Overview
- CacheFactory Useful Methods
- NamedCache Interface Useful Methods
- NamedCache Extending Various Interfaces
- Data Loading Efficiencies
- Data Loading and putAll()
- Reading Entries in a Cache
- Arguments for Running DefaultCacheServer
- Eclipse and OEPE
- Coherence Facets
- Creating Coherence Projects
- Running External Applications in Eclipse
- Coherence Project Anatomy
- Coherence File Editors
- Launching Cache Servers from Eclipse
- Eclipse Environment Variables

- Eclipse Libraries
- Defining Libraries
- Eclipse Run Configurations
- Run Configurations and Coherence
- Maven Support
- OSGi and Coherence
- Summary
- Quiz
- Practice 3-3 Overview: Coherence “Hello World”

Working with Objects

- Objectives
- Agenda
- Java Objects and Coherence
- AirPort Object: Example
- Objects and Identity
- Identity Types
- Implementing Entities
- Identity Generation
- Using Sequence Generators
- Sequence Generators and Entities
- Quiz
- Aggregate Objects
- Value Objects
- Properties of Relationships
- Modeling Relationships in Java
- Cardinality: Examples
- Modeling Relationships in Coherence
- Solving the getXXX problem
- Customer with Repository: Example
- Getting Objects via Repository
- AbstractRepository: Example
- CreditCardRepository: Example
- CoherenceCreditCardRepository: Example
- Quiz
- Practice 4-1 Overview: Developing with Complex Objects
- Agenda
- Serialization Concepts
- Serialization and Performance
- Serialization Options with Coherence
- Serialization Comparison
- Implementing Java Serialization
- Implementing ExternalizableLite
- ExternalizableLite: Example
- Portable Object Concepts

- PortableObject Requirements
- PortableObject: Example
- POF Indices: Requirements
- Registering Portable Objects
- Selected PofWriter Methods
- Selected PofReader Methods
- Quiz
- External Serialization
- PofSerializer Basics
- Registering POF Serializers
- Evolvable Objects
- Implementing Evolvable
- Evolvable and POF
- Serialized Versus Unserialized Caches
- Serialization Testing Support
- Quiz
- Serialization Testing: Example
- Practice 4-2 Overview: Serialization Using ExternalizableLite
- Practice 4-3 Overview: Serialization Using Portable Object Format
- Agenda
- Annotations
- Portable Object Format: Annotations
- Specifying Portable Object Annotations
- Understanding codec
- POF Annotations: Example
- POF Automatic Indexing
- POF Serializer Configuration
- POF Configuration Generation
- Quiz
- Summary

Configuring Coherence Caches

- Objectives
- Agenda
- What Happens to Cached Data?
- Local and Replicated Cache Semantics
- Partitioned and Near Cache Semantics
- Partitioned Cache Semantics
- Near Cache Semantics
- Access to Java Objects: Semantics
- Pass by Reference and Maps
- Pass by Copy and Maps
- Local Storage
- Agenda
- Revisiting coherence-cache-config.xml

- Anatomy of a Cache Configuration
- What Is a Scheme?
- Scheme Composition
- Declaring Cache Mappings
- Using System Properties to Override XML Elements
- Using Macro Parameters to Override XML Elements
- Backing Maps
- Agenda
- What Is a Local Cache?
- When to Use a Local Cache
- When to Not Use a Local Cache
- Local Cache get Diagram
- Local Cache put Diagram
- Configuring a Local Cache
- Defining Local Cache Parameters
- Quiz
- Practice 5-1 Overview: Configuring a Local Cache Scheme
- Agenda
- What Is a Replicated Cache?
- When to Use a Replicated Cache
- When to Not Use a Replicated Cache
- Visualizing Data in a Replicated Cache
- Replicated Cache get Diagram
- Replicated Cache put Diagram
- Configuring a Replicated Cache
- Defining Replicated Cache Parameters
- Quiz
- Practice 5-2 Overview: Configuring a Replicated Cache Scheme
- Agenda
- What Is a Partitioned Cache?
- When to Use a Partitioned Cache
- When to Not Use a Partitioned Cache
- Visualizing Data in a Partitioned Cache
- Partitioned Cache get Diagram
- Partitioned Cache put Diagram
- Partitioned Cache Fault Tolerance
- Storage-Disabled Partitioned Cache
- Configuring a Partitioned Cache
- Defining Partitioned Cache Parameters
- Quiz
- Practice 5-3 Overview: Configuring a Partitioned Cache Scheme
- Agenda
- What Is a Near Cache?
- When to Use a Near Cache
- When to Not Use a Near Cache
- Visualizing Data in a Near Cache

- Near Cache get Diagram
- Near Cache Concurrency Options
- Near Cache put Diagram
- Near Cache put Diagram with Local Storage Disabled
- Near Cache Invalidation Options
- Configuring a Near Cache
- Defining Near Cache Parameters
- Quiz
- Practice 5-4 Overview: Configuring a Near Cache Scheme
- Agenda
- Other Cache Types
- Choosing the Right Cache
- Advanced Configurations
- Agenda
- Elastic Data: Overview
- Benefits of Elastic Data
- Elastic Data and Journaling
- Flash Versus RAM Journals
- Configuring a Journaling Backing Map
- Controlling Journaling Behavior
- Considerations for Elastic Data
- Quiz
- Summary

Observing Data Grid Events

- Objectives
- Agenda
- Event Concepts
- Application Versus Server Side Events
- Event Handler Mechanisms
- Live Event Model: Overview
- Map Trigger: Overview
- Map Listener: Overview
- Agenda
- Map Listener: Review
- MapListener Behavior
- Post-events and MapListeners
- Contract for writing a MapListener
- Selected MapEvent Methods
- MapListener Event
- Synthetic Events
- Determining if an Event is Synthetic
- Behind the scenes: ObservableMap
- Observable Map Workings
- Quiz

- Registering MapListeners
- Map Listeners and Exceptions
- MapListener Registration Variations
- MapListener Lite
- MapListener Implementations
- Anonymous AbstractMapListener: Example
- Making Maps Observable
- MapListeners and Filters
- Provided Filters
- Transforming Events
- Transformer: Example
- Registering a Transform Programmatically
- Quiz
- Practice 6-1 Overview: Working with Map Listeners
- Agenda
- Map Triggers: Review
- MapTrigger Basics
- Registering MapTriggers Programmatically
- MapTrigger: Example
- Registering a MapTrigger Declaratively
- MapTriggers and Exceptions
- Quiz
- Practice 6-2 Overview: Working with Map Triggers
- Agenda
- Live Event Interceptors
- Live Event Model Events
- Partitioned Cache Events
- Partitioned Cache Event Entry Event Behavior
- Partitioned Cache Event Entry Processor Event Behavior
- Partitioned Service Events
- Partitioned Service Event Behavior
- Lifecycle Events
- Developing and Registering Event Handlers
- Contract for Developing Event Interceptors
- Accessing Entry Data
- Base Event and EntryEvent Classes
- Select BinaryEntry Interface methods
- Developing Entry Processor Event Interceptors
- Developing Transfer Event Interceptors
- Developing Transaction Event Interceptors
- Developing Lifecycle Event Interceptors
- Narrowing Interceptor Events
- Narrowing Events: Example
- Interceptors and Event.nextInterceptor
- Registering Event Interceptor Declaratively
- Registering Interceptors for Specific Caches

- Interceptor Chaining
- Quiz
- Practice 6-3 Overview: Working with Event Interceptors
- Agenda
- Continuous Query Cache
- Continuous Query Cache Basics
- Summary

Querying and Aggregating Data in the Cache

- Objectives
- Agenda
- Filters and Caches
- Filter Execution
- Developing Filters
- Accessing Object Properties
- Out-of-the-box Filters
- Example: Filtering with EqualsFilter
- Sorting
- Example: Sorting with a Comparator
- Paging and LimitFilters
- ValueExtractors
- Provided ValueExtractors
- Value Extractors and Dot Notation
- Practice.07.01 Overview: Filtering and Sorting Data
- Aggregating Results
- Understanding Aggregation Execution
- Developing Aggregators
- Main Versus Parallel Execution
- Working with isFinal
- Example Aggregator
- Using Aggregators
- Out-Of-the-Box Aggregators
- Practice 7-1 Overview: Filtering and Sorting Data
- Practice 7-2 Overview: Developing and using Aggregators
- Filters and Indexes
- Index: Example with Ordering
- Index: Example without Ordering
- Quiz
- Data Affinity
- Specifying Data Affinity
- Implementing KeyAssociations
- Booking.Id: Example
- Implementing a KeyAssociator
- Registering KeyAssociators
- Quiz

- Agenda
- What is CohQL?
- CohQL Statements
- Working with Queries
- General SELECT Syntax
- Path Expressions
- Filtering with WHERE
- QueryFilter: Example
- key() and value() Pseudo functions
- Aggregating in SELECT
- Parameters
- Quiz
- Managing Cache Life Cycle
- UPDATE Syntax
- INSERT Syntax
- DELETE Syntax
- Index Management
- Starting QueryPlus
- QueryPlus Arguments
- Query Optimization: Explain Plans and Trace
- Determining Query Performance
- Understanding EXPLAIN PLAN Output
- EXPLAIN PLAN Output: Example
- TRACE Output
- TRACE Output: Example
- Using QueryPlus with TRACE and EXPLAIN PLAN
- Query Optimization: Monitoring
- Summary

Performing In-place Processing of Data with Entry Processors

- Objectives
- Agenda
- Managing Data Consistency and Transactions: Databases
- Managing Data Consistency and Transactions: Coherence
- ConcurrentMap
- ConcurrentMap: Example
- Issues with Locking
- Updating Data in Coherence
- What Is an EntryProcessor?
- Entry Processor Execution Diagram
- Out-of-the-Box EntryProcessors
- EntryProcessor Requirements
- InvocableMap Interface
- InvocableMap.EntryProcessor Interface
- InvocableMap.Entry Interface

- EntryProcessor Semantics
- EntryProcessor Behavior
- Contract for Writing EntryProcessors
- Example: Creating a Custom EntryProcessor
- Concurrent Processing and Entry Processors
- Partition-Level Transactions
- Common Partition-Level Transaction Use Cases
- Example: Pre Partition-Level Transaction
- Partition-Level Transaction Contract
- Accessing Other Entries
- Example: With Partition-Level Transactions
- Accessing data outside a transaction
- Data Affinity: Review
- Specifying Data Affinity
- Implementing KeyAssociations
- KeyAssociation: Example
- Quiz
- Practice 8-1 Overview: Implement, run, and review an EntryProcessor
- Agenda
- What Is an Invocation Service?
- Implementing the Invocable Interface
- Implementing the AbstractInvocable Interface
- AbstractInvocable Implementation: Example
- Examining the InvocationService Interface
- Executing an Invocation Agent Synchronously
- Executing an Invocation Agent Asynchronously
- Registering an Invocation Service
- Obtaining Member Sets
- Quiz
- Practice 8-2 Overview: Implement, run, and review an InvocationService
- Summary

Integrating a Data Source with Coherence

- Objectives
- Agenda
- Persisting Data to Storage
- O/RM Integration
- Data Integration Patterns
- Cache-Aside Pattern
- Read-Through Pattern
- Read-Through: Example
- Write-Through Pattern
- Write-Through: Example
- Write-Behind Pattern
- Write-Behind: Example

- Refresh-Ahead Pattern
- Agenda
- CacheLoader Interface
- CacheLoader and CacheStore Life Cycle
- CacheLoaders and Initialization
- CacheLoader: Example
- Configuring CacheLoaders
- CacheStore Interface
- CacheStore.Store Example
- CacheStore.Load Example
- CacheStore.*All Example
- CacheStore Architecture
- Refresh-Ahead Caching
- Refresh-Ahead Configuration
- Write-Through
- Write-Behind
- Write-Behind Configuration
- What Happens If a Write-Behind Transaction Fails?
- Write-Behind Caveats
- Write Behind and Thread Count
- Idempotency
- Setting Write-Behind Factor Dynamically
- Practice 9-1 Overview: Developing and Registering CacheStores
- Bulk Cache Loading via Put
- Bulk Loading with Queries
- Distributed Invocable Loading
- Agenda
- Java Persistence Architecture
- JPA and Persistence Requirements
- JPA Approach
- What Are JPA Entities?
- Entity Class Requirements
- JPA and CacheStores
- Integrating JPA and Coherence
- Obtaining a JPA Implementation
- Mapping Entities
- Configuring JPA
- Coherence JPA Configuration
- Cachestore-scheme
- Cache Mapping
- Agenda
- Oracle GoldenGate
- Coherence GoldenGate Adapter
- Object Mapping and Operations
- Configuration: Overview
- Summary

Typical Caching Architectures

- Objectives
- Agenda
- Single-Application Instances
- Multiple-Application Instances
- Local Caching Pattern
- Distributed Coherent Caching Pattern
- Introducing the Caching Infrastructure Layer
- Cache-Aside Patterns
- Read-Through Pattern
- Write-Aside Pattern
- Write-Through Pattern
- Write-Behind Pattern
- External Update Anti-Pattern
- External Update Pattern Using Messaging
- External Update with Direct Access
- Near-Caching Pattern
- Client-Side Event Processing Pattern
- Server-Side Event Processing Pattern
- Server-Side Processing Pattern: Queries, Map Reduction, and Computation
- Combining Features for a Scalable Platform
- Quiz
- Summary
- Practice 10-1 Overview: Examining Sample Topologies

Representational State Transfer

- Objectives
- Agenda
- What Is REST?
- Coherence REST
- Coherence REST Requirements
- Supported Representations
- Agenda
- Annotations
- Class Requirements (JAXB)
- XmlAccessType.PUBLIC_MEMBER: Example
- XmlAccessType.PROPERTY: Example
- XmlAccessType.FIELD: Example
- XmlAttribute and XmlTransient
- Class Requirements (JSON)
- JSON Mapping: Example
- JSonProperty and JSonIgnore
- Key Converters
- KeyConverter: Example

- Quiz
- Agenda
- REST Configuration
- Registering Objects
- REST resources
- REST and POF
- Serving Coherence REST Requests
- Configuring REST Proxies
- Configuration: Example
- Starting a REST Proxy
- REST Configuration Best Practices
- Custom REST Server
- Contract for AbtractHttpServer
- Start Method
- Stop Method
- Registering a Custom REST Server
- WebLogic Server Deployment
- REST Web Application Structure
- web.xml Deployment Descriptor
- weblogic.xml Deployment Descriptor
- Deploying the Web Application
- Agenda
- Single-Object Operations
- Multiple-Object Operations
- Partial Results
- Queries
- Named Queries
- Named Query: Example
- Queries with Parameters
- Entry Processors and REST
- Custom Entry Processors
- Quiz
- Agenda
- Coherence REST Security
- Configuring HTTP Basic Security
- HTTP Basic Authentication
- Login Module
- Specifying a Login Module
- SSL Authentication
- Well Known Addresses and SSL
- HTTP Basic Authentication and SSL
- Agenda
- Jersey and JAX-RS
- Understanding JAX-RS Clients
- Working with Clients
- Working with WebResources

- Select WebResource Methods
- Working withs ClientResponses
- Select ClientResponse Methods
- Working with Builders
- Example
- Summary
- Practice 11-1 Overview: Exposing Objects using REST

Managed Coherence Servers for WLS

- Objectives
- Agenda
- Coherence and WebLogic Server
- WebLogic Managed Coherence Servers Operations
- What Is a Grid Archive (GAR)?
- Coherence Application Deployment
- Coherence Container: Benefits
- Agenda
- Installing WebLogic with Coherence
- Terminology: Review
- WebLogic Server Domain: Review
- Administration Server: Review
- Managed Servers: Review
- Machines and WebLogic Clusters: Review
- Coherence Cluster
- Managed Coherence Server
- Working with Coherence Clusters
- Coherence Cluster Configuration
- Default Coherence Cluster
- Coherence Cluster Settings
- Additional Coherence Cluster Settings
- WebLogic Cluster Coherence Settings
- Debugging Configuration
- Creating Managed Coherence Servers
- Configuring Managed Coherence Servers
- Additional Coherence Server Settings
- Coherence Deployment Tiers and Topologies
- Best Practice: Single Data with Web Cluster Topology
- Coherence Deployment Topology
- Agenda
- Coherence Application Grid Archives
- GAR Structure
- Deployment Descriptor coherence-application.xml
- GAR Deployment Modes
- Coherence Application Isolation
- Packaging GARs Inside EARs

- Application Isolation and EARs
- Deploying Coherence Applications
- DefaultCacheServer and GARs
- WebLogic Scripting Tool (WLST)
- Running WLST Scripts
- WLST: Creating Coherence Clusters
- Modifying Coherence Cluster Parameters
- Associating a Coherence Cluster with a WLS Cluster
- WLST: Creating Managed Coherence Servers
- Practice 12-1 Overview: Coherence Applications and WebLogic Server
- Agenda
- Application Lifecycle Listeners
- Developing and Registering LifecycleListeners
- Accessing Context Within a LifecycleListener
- Lifecycle Listener Example
- Overriding Configuration by Using JNDI
- Overriding Cache Configuration: Example
- Coherence Applications and Scope
- WebLogic and Deployment
- Agenda
- Coherence*Web: Overview
- In-Memory Session Replication: Review
- Coherence*Web and WebLogic Clusters
- Coherence*Web Architecture
- Coherence*Web Configuration
- Managed Coherence Server Coherence*Web Instance
- Quiz
- Summary
- Practice 12-2 Overview: Lifecycle Listeners

Coherence*Extend

- Objectives
- Agenda
- What Is Coherence*Extend?
- Coherence*Extend Clients
- Coherence*Extend Architecture
- Coherence*Extend Capabilities
- Coherence*Extend Advantages
- Coherence*Extend Disadvantages
- When to Use Coherence*Extend?
- Data Replication
- Agenda
- Configuring Coherence*Extend
- Cluster-Side Cache Configuration Descriptor
- Client-Side Cache Configuration Descriptor

- Launching a Coherence*Extend-Enabled DefaultCacheServer Process
- and Java Client Application
- Quiz
- Agenda
- Supported Environments
- Installing the C++ Distribution
- C++ Distribution Structure
- Installing the .NET Distribution
- .NET Distribution Structure
- C++ Extend Clients
- C++ Application: Overview
- Coherence C++ Namespaces
- Interacting with Caches
- CacheFactory Useful Methods
- Select NamedCache Methods
- Agenda
- Simplifying Coherence*Extend Configuration
- Address Provider References
- Configuring Address Provider References
- Step 1: Override File
- Step 2: Cluster Configuration
- Step 3: Client Configuration
- Configuring a Naming Service
- Step 1: Cluster Configuration
- Step 2: Client Configuration
- Dynamic Proxy Thread Pooling
- Configuring Thread Counts
- F5 BIG-IP Local Traffic Manager
- Dynamic Load Balancing for Coherence*Extend
- Proxy-Based Load Balancing
- Connection Load Balancing
- Client-Based Load Balancing: Per Proxy
- Client-Based Load Balancing: Systemwide
- Proxy-Based Load Balancing Algorithm
- Custom Load Balancing
- ProxyServiceLoadBalancer Interface
- AbstractProxyServerLoadBalancer Class
- ProxyServiceLoad Interface
- Practice 13-01 Overview: Configuring and Running Coherence*Extend
- Practice 13-02 Overview: Writing a Coherence*Extend Java Client
- Summary

Understanding Coherence Security

- Objectives
- Agenda

- Cluster Connectivity (TCMP)
- Member Identity
- Authorized Hosts
- Agenda
- Access Control
- Configuring Custom Access Controllers
- Access Controller security framework
- ClusterPermission
- Registering an Access Controller
- Agenda
- Extend Pluggable Identity: Architecture
- Extend Pluggable Identity: Client-Side
- Extend Pluggable Identity: Client-Side Code Example
- Extend Pluggable Identity: Identity Transformer
- Extend Pluggable Identity: Identity Transformer Code
- Extend Pluggable Identity: Proxy-Side Identity Asserter
- Extend Pluggable Identity: Identity Asserter Code
- Extend Pluggable Identity: Identity Asserter Semantics
- Extend Pluggable Identity: Identity Asserter Subject Scoping
- Extend Access Control: Authorization Wrappers
- Extend Access Control: Configuring Authorization Wrappers
- Extend Access Control: EntitledCacheService
- Extend Access Control: EntitledNamedCache
- Extend Access Control: EntitledInvocationService
- Extend Access Control: Authorization Example Implementation
- Agenda
- Transport Layer Security: SSL
- Transport Layer Security: SSL Recommendations
- Transport Layer Security: Setting up SSL for the Cluster
- Transport Layer Security: Setting up SSL for *Extend
- Security Examples
- Quiz
- Summary

Coherence JMX and Reporter

- Objectives
- Agenda
- Overview
- Management Architecture
- Configuring Coherence JMX
- Accessing the Coherence MBean Using HTTP and JMX RI
- Accessing the Coherence MBean Using Java Bundled JMX Implementation
- Accessing the Coherence MBean Using the Coherence MBeanConnector
- System MBeans to Watch
- Quiz

- Agenda
- What Is the Reporter?
- Configuring Basic Settings
- Managing Reporter MBean Attributes
- Managing Reporter MBean Operations
- Finding Reporter Log Data
- Viewing Reporter Data
- Creating Custom Reports
- Running Reporter in a Distributed Environment
- Quiz
- Summary
- Practice 15-1 Overview: Configuring and Running the Reporter

Oracle Cloud

- Agenda
- What is Cloud?
- What is Cloud Computing?
- History – Cloud Evolution
- Components of Cloud Computing
- Characteristics of Cloud
- Cloud Deployment Models
- Cloud Service Models
- Industry Shifting from On-Premises to the Cloud
- Oracle IaaS Overview
- Oracle PaaS Overview
- Oracle SaaS Overview
- Summary

Oracle Java Cloud Service Overview

- Objectives

Introducing Java Cloud Service Your platform for running business applications in

- the cloud
- Java Cloud Service: Three Options
- Java Cloud Service Main Use Cases
- Java Cloud Service Feature: Provisioning
- Java Cloud Service Feature: Patching
- Java Cloud Service Feature: Backup / Restore
- Java Cloud Service Feature: Scaling
- Oracle Coherence Option: Data Caching & Scaling
- Oracle Coherence Option: Your Cloud Data Grid Scalable, fault-tolerant cloud
- infrastructure
- How You Interact with Java Cloud Service
- Speaking of Dev Environments… Developer Cloud Service
- Java Cloud Service On-Premises!
- Summary

- Configuration Management Reduces time to diagnose and repair issues B-17
- Monitor Complete Infrastructure as a Single System Single console
- drastically reduces total cost of ownership B-18
- Integrated Fusion Middleware Management B-19
- End-to-End Management Manage performance and change across all tiers B-20
- Third-Party Management Tools B-21
- Quiz B-22
- Agenda B-23
- Production versus Development Modes B-24
- Hardware Recommendations B-26
- JVM Recommendations B-29
- JVM Deployment Concerns B-30
- JVM Heap Sizing B-32
- Tuning Garbage Collection B-34
- Sizing a Coherence System B-36
- General Sizing Guidelines B-37
- Size-Limiting Storage JVMs B-39
- Binary Calculator: Example B-40
- Network Configuration B-41
- Operating System Tuning B-45
- Setting Buffer Sizes B-46
- Testing Multicast B-48
- Datagram Test B-51
- Service Configuration B-54
- Setting Thread Count B-55
- Partition Count B-56
- Tuning the OS for Coherence*Extend B-57
- Tuning the Coherence*Extend Client Side B-58
- Tuning the Coherence*Extend Cluster Side B-59
- Cluster Quorum B-60
- Partitioned Cache Quorum B-61
- Extend Proxy Quorum B-63
- Coherence Logs B-64
- Performance Testing: General Advice B-65
- Quiz B-67
- Summary B-69
- C New Features Overview
- Objectives C-2
- JDK 8 Language Features C-3
- Asynchronous Named Cache and Entry Processors C-4
- Coherence Cluster Management C-5
- Security Enhancements C-6
- Elastic Data Improvements C-7
- Persistent Caching C-8
- Federated Caching C-9
- Coherence and WebLogic MT C-10