

## **Oracle® Fusion Middleware**

Tutorial for Oracle Coherence

12c (12.1.3)

**E47889-01**

May 2014

Documentation for developers and architects that provides step-by-step examples of creating, configuring, and deploying Oracle Coherence-based Java applications.

Oracle Fusion Middleware Tutorial for Oracle Coherence, 12c (12.1.3)

E47889-01

Copyright © 2014, Oracle and/or its affiliates. All rights reserved.

Primary Author: Thomas Pfaeffle

Contributing Author: Noah Arliss, Mark Falco, Alex Gleyzer, Gene Gleyzer, David Guy, Charlie Helin, Tim Middleton, Brian Oliver, Cameron Purdy

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

---

---

# Contents

<b>Preface</b> .....	xvii
Audience .....	xvii
Documentation Accessibility .....	xvii
Related Documents .....	xvii
Conventions .....	xviii
<b>What's New in This Guide</b> .....	xix
New and Changed Features for 12c (12.1.3) .....	xix
Other Significant Changes in this Document for 12c (12.1.3) .....	xix
<b>1 Installing and Configuring Coherence</b>	
<b>Prerequisites</b> .....	1-1
<b>Installing Coherence</b> .....	1-1
<b>Testing a Coherence Installation</b> .....	1-2
To Test a Coherence Installation—Main Steps .....	1-3
Configure and Run the Sample Cache Server Application .....	1-3
Configure and Run the Sample Cache Client Application .....	1-7
Exercise the Sample Cache Client Application .....	1-9
Troubleshooting Cache Server Clustering .....	1-14
Restricting Coherence to Your Own Host .....	1-15
Advanced Steps to Restrict Coherence to Your Own Host .....	1-15
<b>Introducing the Coherence examples.zip File</b> .....	1-15
<b>2 Installing and Configuring Eclipse and OEPE with Coherence</b>	
<b>Installing Eclipse and OEPE</b> .....	2-1
<b>Configuring Eclipse and OEPE for Coherence</b> .....	2-1
<b>Creating a New Project in the Eclipse IDE</b> .....	2-3
<b>Launching a Cache Server in the Eclipse IDE</b> .....	2-6
<b>Learning Eclipse IDE Basics</b> .....	2-11
Creating a Java Class .....	2-11
Creating a Runtime Configuration .....	2-12
Stopping Cache Servers .....	2-14

<b>3</b>	<b>Accessing the Data Grid from Java</b>	
	Introduction.....	3-1
	<b>Creating Your First Coherence-Based Java Program</b> .....	3-2
	Create a Program to Put Values in the Cache .....	3-2
	Create a Program to Get Values from the Cache.....	3-5
	<b>Creating Your First Coherence-Based Java Application</b> .....	3-9
	Create the Console Application .....	3-10
	Run the Console Application .....	3-11
<b>4</b>	<b>Working with Complex Objects</b>	
	Introduction.....	4-1
	<b>Creating and Caching Complex Objects</b> .....	4-2
	Create the Data Objects .....	4-2
	Create the Complex Object .....	4-13
	Create the Driver Class.....	4-17
	Create the POF and Cache Configuration Files .....	4-18
	Run the Sample Project .....	4-20
<b>5</b>	<b>Loading Data Into a Cache</b>	
	Introduction.....	5-1
	<b>Populating a Cache with Domain Objects</b> .....	5-1
	Create a Class with the Key for the Domain Objects .....	5-2
	Edit the POF Configuration File .....	5-5
	Create the Data Generator .....	5-6
	Create a Console Application to Load the Cache.....	5-10
	Run the Cache Loading Example.....	5-12
	<b>Querying and Aggregating Data in the Cache</b> .....	5-15
	Create the Class to Query Cache Data .....	5-17
	Run the Query Example.....	5-19
	Edit the Query Example to Perform Aggregations.....	5-21
	Run the Query and Aggregation Example.....	5-24
<b>6</b>	<b>Simplifying Cache Calls and Aggregations</b>	
	Introduction.....	6-1
	<b>Simplifying the Query Example</b> .....	6-2
	<b>Rerunning the Query Example</b> .....	6-6
<b>7</b>	<b>Listening for Changes and Modifying Data</b>	
	Introduction.....	7-1
	<b>Creating a Cache Listener</b> .....	7-2
	Create a Class to Listen for Changes in the Cache .....	7-2
	Run the Cache Listener Example .....	7-4
	<b>Responding to Changes in the Cache</b> .....	7-5
	Create a Class to Update Entries in the Cache.....	7-6
	Edit the POF Configuration File .....	7-8

Run the Cache Update Example .....	7-9
<b>8 Using JPA with Coherence</b>	
<b>Introduction</b> .....	8-1
<b>Mapping Relational Data to Java Objects with JPA</b> .....	8-2
Unlock the Oracle Database.....	8-2
Configure the Project for JPA .....	8-3
Create the JPA Persistence Unit and Entities .....	8-10
Edit the persistence.xml File.....	8-13
Create the Cache Configuration File for JPA .....	8-17
Create a Cache Server Start-Up Configuration.....	8-18
Create a Class to Interact with the Data Object .....	8-19
Create a Run Configuration for RunEmployeeExample.....	8-20
Run the JPA Example .....	8-21
<b>9 Interacting with the Cache and the Database</b>	
<b>Introduction</b> .....	9-1
<b>Creating a Cache Application</b> .....	9-2
Create an Application that Constructs a Cache .....	9-2
Create a Cache Configuration File.....	9-4
Create a Run Configuration for the Application.....	9-5
Create the Cache Server Start-Up Configuration .....	9-5
Run the Cache Creation Application.....	9-6
<b>Creating a Database Cache</b> .....	9-7
Create an Oracle Database Cache .....	9-7
Create a Class to Define a Custom Cache Store.....	9-8
Modify the Cache Configuration File.....	9-11
Create a Class to Construct the Database Cache .....	9-13
Run the Database Cache Application.....	9-16
<b>10 Working with Security</b>	
<b>Introduction</b> .....	10-1
<b>Enabling Token-Based Security</b> .....	10-1
Use a Security Helper File.....	10-2
Create an Identity Transformer .....	10-5
Create an Identity Asserter.....	10-7
Create the Password File.....	10-9
Enable the Identity Transformer and Asserter .....	10-10
Create a Cache Configuration File for the Extend Client.....	10-11
Create a Cache Configuration File for the Extend Proxy .....	10-12
Create a Start-Up Configuration for a Cache Server .....	10-14
Create a Start-Up Configuration for a Cache Server with a Proxy Service .....	10-15
Run the Password Example.....	10-16
<b>Including Role-Based Access Control to the Cluster</b> .....	10-18
Define Which User Roles Are Entitled to Access Cache Methods.....	10-18
Apply the Entitlements to the Cache Service.....	10-27

Create the Access Control Example Program .....	10-29
Edit the Cluster-Side Cache Configuration File.....	10-31
Run the Access Control Example.....	10-32
<b>Including Role-Based Access Control to an Invocable Object.....</b>	<b>10-36</b>
Create an Invocable Object .....	10-36
Create an Entitled Invocation Service .....	10-38
Create the Access Invocation Service Example Program .....	10-40
Edit the Cluster-Side Cache Configuration File.....	10-42
Create a POF Configuration File.....	10-42
Edit the Run Configurations for the Servers .....	10-43
Run the Access Invocation Service Example.....	10-44

## 11 Working with Live Events

<b>Introduction.....</b>	<b>11-1</b>
About Event Interceptors.....	11-1
About Cache Events.....	11-2
About Partitioned Service Events .....	11-2
About Event Interceptor Registration .....	11-2
<b>Creating, Registering, and Executing an Event Interceptor .....</b>	<b>11-3</b>
Create a Event Interceptor to Measure the Time Between a Pre- and a Post-commit Event .....	11-4
Create a Class to Delay the Processing of Events.....	11-9
Register the Timed Events Event Interceptor .....	11-11
Create a POF Configuration File for the Lazy Processor Class .....	11-13
Create a Class to Exercise the Timed Events Event Interceptor.....	11-13
Create a Driver File for Timed Events Example .....	11-16
Create a Cache Server Startup Configuration.....	11-17
Create a Startup Configuration for the Timed Events Driver .....	11-18
Run the Timed Events Example.....	11-19
<b>Vetoing Pre- and Post-commit Events Using an Event Interceptor .....</b>	<b>11-21</b>
Create an Event Interceptor to Detect and Veto Events .....	11-21
Register the Veto Events Event Interceptor.....	11-23
Create a Class to Exercise the Veto Events Event Interceptor .....	11-25
Edit the Driver File for the Veto Events Example .....	11-27
Run the Veto Events Example .....	11-27
<b>Logging Partition Activity Using an Event Interceptor .....</b>	<b>11-29</b>
Create a Class to Terminate a JVM and to Enable and Disable Logging.....	11-30
Create an Event Interceptor to Log Partition Activity .....	11-33
Create a Class to Exercise the Log Partition Activity Example .....	11-34
Register the Log Partition Activity Event Interceptor .....	11-35
Edit the POF Configuration File .....	11-37
Edit the Driver File for the Log Partition Activity Example.....	11-38
Run the Log Partition Activity Example.....	11-39

## 12 Working with JCache

<b>Introduction.....</b>	<b>12-1</b>
<b>Creating a JCache-Based Java Project.....</b>	<b>12-2</b>
<b>Creating a JCache-Based Application to Put Values in the Cache.....</b>	<b>12-3</b>

<b>Running a JCache Application in a Cluster</b> .....	12-5
Configure an Example Cluster .....	12-6
Store the Object in a Pass-Through Cache.....	12-6
Modify the Sample JCache Application .....	12-7
Define the Sample Cache for JCache.....	12-7
Start the Example Cache Server .....	12-8
Run the Application.....	12-9
Verify the Cache .....	12-11
Create a JAR file for the Application.....	12-11
Verify the Cache Contents Using a Cache Factory Instance.....	12-12

## 13 Caching Sessions with Managed Coherence Servers

<b>Introduction</b> .....	13-1
<b>Caching Session Information for Web Application Instances</b> .....	13-2
Configure and Start WebLogic Server .....	13-3
Create a Machine.....	13-3
Create the WebLogic Servers.....	13-5
Create a Coherence Cluster .....	13-7
Enable a Server for Coherence*Web Local Storage.....	13-9
Create the Counter Web Application.....	13-10
Start the WebLogic Servers.....	13-12
To Start the WebLogic Servers from the Command Line .....	13-12
To Start the WebLogic Servers from the WebLogic Server Administration Console ..	13-13
Deploy the Application .....	13-13
Verify the Example .....	13-17
<b>Working with Custom Session Cache Configuration Files</b> .....	13-18

## A Coherence Examples in the examples.zip File

<b>Examples Provided in the examples.zip File</b> .....	A-1
<b>Obtaining the examples.zip File</b> .....	A-3
<b>How to Build the Examples</b> .....	A-3
How to Build the Java Examples .....	A-3
Prerequisites for Java.....	A-4
Directory Structure for Java.....	A-4
Build Instructions for Java .....	A-4
How to Build the .NET Examples.....	A-4
Prerequisites for .NET .....	A-5
Directory Structure for .NET.....	A-5
Build Instructions for .NET .....	A-5
How to Build the C++ Examples .....	A-5
Prerequisites for C++.....	A-6
Directory Structure for C++ .....	A-6
Build Instructions for C++.....	A-7
<b>How to Run the Examples</b> .....	A-8
How to Run the Java Examples.....	A-8
Prerequisites for Java.....	A-8

Directory Structure for Java.....	A-8
Instructions for Java.....	A-9
How to Run the .NET Examples.....	A-9
Prerequisites for .NET.....	A-10
Directory Structure for .NET.....	A-10
Instructions for .NET.....	A-10
How to Run the C++ Examples.....	A-10
Prerequisites for C++.....	A-11
Directory Structure for C++.....	A-11
Instructions for C++.....	A-11
<b>Coherence Basic Features Examples.....</b>	<b>A-12</b>
Running the Example Set.....	A-13
Understanding the Features Driver File.....	A-13
Basic Data Access Example.....	A-14
Example Output.....	A-14
Loader Example.....	A-15
Example Output.....	A-15
Query Example.....	A-15
Example Output.....	A-17
Observer Example.....	A-17
Processor Example.....	A-18
Example Output.....	A-19
Query Language.....	A-20
Example Output.....	A-21
Data Generator.....	A-22
<b>Coherence Security Examples.....</b>	<b>A-22</b>
This Example Set.....	A-22
Running the Security Example Set.....	A-22
Understanding the Security Driver File.....	A-22
Password Example.....	A-23
Access Control Example.....	A-24
Example Output.....	A-25
Password Identity Transformer.....	A-25
Password Identity Asserter.....	A-26
Entitled Cache Service.....	A-27
Entitled Invocation Service.....	A-27
Entitled Named Cache.....	A-27
Security Example Helper.....	A-28
<b>Coherence Live Events Examples.....</b>	<b>A-28</b>
This Example Set.....	A-29
Running the Live Events Example Set.....	A-29
Understanding the Live Events Driver File.....	A-29
EventsExamples.....	A-29
EventsTimingExample.....	A-30
VetodEventsExample.....	A-30
RedistributionEventsExample.....	A-30
TimedTraceInterceptor.....	A-31

CantankerousInterceptor ..... A-31  
RedistributionInterceptor ..... A-31  
RedistributionInvocable ..... A-31  
LazyProcessor ..... A-32



## List of Examples

1-1	cache-server.cmd File with an Edited COHERENCE_HOME .....	1-4
1-2	Output from Starting a Coherence Cache Server .....	1-5
1-3	query.cmd File with an Edited COHERENCE_HOME.....	1-7
1-4	Output from Starting the Coherence Cache Client.....	1-8
1-5	Output from Starting a Coherence Cache Client.....	1-9
1-6	Exercising Coherence Commands .....	1-12
2-1	Cache Server Output in the Eclipse Console Window .....	2-9
3-1	Creating a NamedCache Object: Inserting and Verifying Values .....	3-3
3-2	Output of MyFirstSample Program .....	3-3
3-3	Getting a Value from the Cache.....	3-5
3-4	Output of the MyFirstSampleReader Program .....	3-5
3-5	Output of MyFirstSampleReader Program with a Running Cache Server .....	3-7
3-6	Output of the MyFirstSample Class with Cache Storage Disabled .....	3-9
3-7	A Coherence-Based Java Application .....	3-10
3-8	Output for the QueryPlus Cache Client .....	3-12
3-9	Output of the YourFirstCoherenceApplication Class .....	3-12
3-10	Output of the YourFirstCoherenceApplication Class with a New Key Value .....	3-13
4-1	Implementation of an Address Class.....	4-7
4-2	Implementation of a PhoneNumber Class .....	4-10
4-3	Sample Contact Class .....	4-14
4-4	Sample ContactDriver Class.....	4-17
4-5	POF Configuration File .....	4-18
4-6	Cache Configuration File .....	4-19
4-7	Output from the Contacts Cache Server.....	4-21
4-8	Output of the Contacts Example in the Eclipse IDE.....	4-23
4-9	Contacts Cache Server Displaying the Arrival and Departure of the Contacts Client..	4-24
5-1	Simple Contact ID Class.....	5-3
5-2	POF Configuration File with the ContactId Entry .....	5-5
5-3	Sample Data Generation Class.....	5-6
5-4	Contents of the contacts.csv File.....	5-10
5-5	Sample Cache Loading Program .....	5-10
5-6	Output from the Sample Cache Loading Program .....	5-14
5-7	Sample QueryExample Class .....	5-17
5-8	Results of the QueryExample Program .....	5-20
5-9	Methods to Aggregate Over Keys or by Specifying Filters .....	5-21
5-10	QueryExample with Aggregation .....	5-22
5-11	Output from the Aggregators .....	5-24
6-1	Edited QueryExample File .....	6-4
6-2	Output of the MA Residents Filter .....	6-6
6-3	Output of the MA Residents, Work Elsewhere Filter .....	6-7
6-4	Output of the City Begins with S Filter.....	6-7
6-5	Output of the Age Greater than 42 Filter.....	6-8
6-6	Output of the State and Age Aggregators.....	6-8
7-1	Listener Methods on a NamedCache.....	7-1
7-2	Code Pattern for Registering an Event .....	7-2
7-3	Sample Listener Class.....	7-2
7-4	Listener Program Waiting for Events .....	7-4
7-5	Sample Program to Update an Object in the Cache.....	7-7
7-6	Output from the ObserverExample and ProcessorExample Classes .....	7-9
8-1	Generated persistence.xml File .....	8-16
8-2	Cache Configuration for JPA.....	8-17
8-3	Sample Employee Class File.....	8-20
8-4	Output from the RunEmployee Executable.....	8-21
8-5	Cache Server Response to Logging In to the Database .....	8-22

9-1	Implementation of a Coherence Cache .....	9-3
9-2	Cache Configuration File .....	9-4
9-3	Output of the Coherence Cache Application.....	9-7
9-4	SQL Script for Creating a Database Table .....	9-8
9-5	Running the SQL Script for Creating a Database Table .....	9-8
9-6	Database Cache Store Implementation.....	9-8
9-7	Database Cache Configuration File.....	9-12
9-8	Implementation for the DatabaseCache Class File .....	9-13
9-9	Output of the DatabaseCache Program.....	9-17
9-10	Cache Server Response to the DatabaseCache Program.....	9-17
9-11	Output from the SELECT Statement.....	9-18
10-1	A Security Helper File .....	10-3
10-2	Sample Identity Transformer Implementation.....	10-6
10-3	Sample Identity Asserter Implementation .....	10-7
10-4	Sample Implementation to Run the Password Example.....	10-9
10-5	Specifying an Identity Transformer and an Asserter .....	10-11
10-6	Sample Extend Client Cache Configuration File.....	10-11
10-7	Sample Cache Configuration File for the Proxy Server.....	10-13
10-8	Password Example Output in the Eclipse Console.....	10-17
10-9	Response from the Cache Server Running the Proxy Service Shell .....	10-17
10-10	Entitled Named Cache .....	10-19
10-11	Entitled Cache Service.....	10-28
10-12	Sample Program to Run the Access Control Example .....	10-29
10-13	Cache Service Proxy Configuration for a Cluster-Side Cache Configuration.....	10-32
10-14	Access Control Example Output in the Eclipse Console.....	10-33
10-15	Output for the Cache Server Running the Proxy Service .....	10-34
10-16	A Sample Invocable Object.....	10-37
10-17	A Sample Entitled Invocation Service.....	10-39
10-18	Sample Program to Run the Access Invocation Service Example .....	10-40
10-19	Invocation Service Proxy Configuration for a Cluster-Side Cache.....	10-42
10-20	POF Configuration File with ExampleInvocable User Type .....	10-42
10-21	Client Program Response in the Eclipse Console.....	10-45
10-22	Proxy Service Response in the Eclipse Console.....	10-46
11-1	Class to Provide Timings Between Pre- and Post-commit Events .....	11-4
11-2	Class to Delay the Processing of Events .....	11-10
11-3	Cache Configuration File That Registers the TimedTraceInterceptor .....	11-12
11-4	POF Configuration File for the LazyProcessor Class .....	11-13
11-5	Class to Exercise the TimedTraceInterceptor Event Interceptor .....	11-14
11-6	Driver File for Timed Events Example .....	11-16
11-7	Output from the Cache Client.....	11-19
11-8	Class to Detect and Veto Events .....	11-21
11-9	Cache Configuration to Register the CantankerousInterceptor Class .....	11-23
11-10	Class to Exercise the TimedTraceInterceptor EventInterceptor.....	11-25
11-11	Output from the Veto Events Client .....	11-27
11-12	Output from the Cache Server .....	11-28
11-13	Class to Terminate a JVM and to Enable or Disable Logging .....	11-30
11-14	Class to Log Partition Events .....	11-33
11-15	Sample Class to Exercise the Log Partition Activity Example .....	11-34
11-16	Cache Configuration File with Event Interceptors .....	11-36
11-17	POF Configuration File for the Log Partition Events Example.....	11-38
11-18	Output from the Cache Client .....	11-39
11-19	Output from First Cache Server.....	11-39
12-1	Creating a JCache Cache Object: Inserting and Verifying Values .....	12-4
12-2	Output of MyFirstJCacheSample Program .....	12-5
12-3	Output of the DefaultCacheServer .....	12-9

12-4	Output of the MyFirstJCacheSample Application .....	12-10
12-5	Output of the CacheFactory Instance.....	12-13
13-1	Sample weblogic.xml File .....	13-11
13-2	Sample MANIFEST.MF File .....	13-11
A-1	Example Output of the Basic Data Access Example .....	A-14
A-2	Example Output of the LoaderExample .....	A-15
A-3	Example Output of the Query Example .....	A-17
A-4	Example Output of the Processor Example .....	A-19
A-5	Example Output of the Query Language Example .....	A-21
A-6	Example Output of the Password Example .....	A-23
A-7	Example Output of the Access Control Example.....	A-25
A-8	Example Output of the EventsTimingExample.....	A-30
A-9	Example Output of the VetodEventsExample .....	A-30
A-10	Output of the RedistributionEventsExample .....	A-30

## List of Tables

1-1	Network Addresses and Ports Used by Coherence.....	1-2
3-1	Methods in the NamedCache Interface .....	3-1
3-2	Methods in the CacheFactory Class .....	3-2
6-1	ReflectionExtractors and Their Equivalent createExtractor Statements .....	6-2
6-2	*Filter Statements and Their Equivalent createFilter Statements in Queries .....	6-3
6-3	Filter Statements and Their Equivalent createFilter Statements in Aggregations.....	6-3
9-1	Descriptions of Cache Types .....	9-2
9-2	Types of Read/Write Caching Supported by Coherence .....	9-13
A-1	Coherence Basic Features Examples .....	A-2
A-2	Coherence Security Examples .....	A-2
A-3	Coherence Live Events Examples.....	A-2
A-4	Directory Structure for Java.....	A-4
A-5	Directory Structure for .NET .....	A-5
A-6	Directory Structure for C++ .....	A-6
A-7	Directory Structure for Java.....	A-8
A-8	Directory Structure for .NET .....	A-10
A-9	Directory Structure for C++ .....	A-11
A-10	Data Model Classes for the Features Examples.....	A-13

## List of Figures

2-1	Workspace Launcher Dialog Box .....	2-2
2-2	The Java EE Perspective in the Menu Bar .....	2-3
2-3	New Application Client Project Dialog Box.....	2-3
2-4	Selecting Oracle Coherence in the Project Facets Dialog Box.....	2-4
2-5	Specifying a Configuration Name in the Save Preset Dialog Box .....	2-4
2-6	Creating a Coherence User Library .....	2-5
2-7	Naming the Coherence Library .....	2-6
2-8	The coherence.jar File Defined in the Coherence User Library.....	2-6
2-9	Main Tab in the Run Configurations Dialog Box.....	2-7
2-10	Coherence Tab of the Run Configurations Dialog Box .....	2-8
2-11	Common Tab of the Run Configurations Dialog Box .....	2-9
2-12	Defining a New Java Class .....	2-12
2-13	Defining a Launch Configuration for a Runnable File.....	2-13
2-14	Setting Runtime Arguments for the Launch Configuration.....	2-13
2-15	Java Process in the Windows Task Manager .....	2-14
3-1	Main Tab for the Query Client Configuration.....	3-11
4-1	Generate Getters and Setters Dialog Box.....	4-4
4-2	Generate Constructors using Fields Dialog Box.....	4-5
4-3	Coherence Tab for the Contacts Cache Server Executable.....	4-20
5-1	Adding Folders to the Project Build Path.....	5-2
5-2	Contents of the Order and Export Tab for the Loading Project .....	5-3
5-3	Classpath for the DataGenerator Program.....	5-13
5-4	Classpath for the LoaderExample Program.....	5-14
8-1	Connecting to the Database.....	8-3
8-2	Selecting the JPA Perspective in the Open Perspective Dialog Box .....	8-4
8-3	Project Facets for a JPA Project .....	8-4
8-4	Contents of the New JPA Project Dialog Box .....	8-5
8-5	Adding the EclipseLink JPA Library .....	8-6
8-6	Adding Toplink Grid, EclipseLink, and Java Persistence JAR Files to the TopLink User Library 8-7	
8-7	TopLink Libraries Added to the JPA Facet Page .....	8-7
8-8	Connection Profile Page.....	8-8
8-9	The New Connection Profile Dialog Box and the Success Message.....	8-9
8-10	Coherence Page of the New JPA Project Wizard .....	8-9
8-11	Selecting the Database Tables .....	8-10
8-12	Associations for the EMPLOYEES Table.....	8-11
8-13	Customize Default Entity Generation Page.....	8-12
8-14	Customize Individual Entities Page.....	8-13
8-15	General Tab in the persistence.xml Editor .....	8-14
8-16	Connection Properties Tab in the persistence.xml Editor.....	8-15
8-17	Logging Tab in the persistence.xml Editor .....	8-16
8-18	Properties Tab in the persistence.xml Editor .....	8-16
8-19	Main Tab for the JPA Cache Server.....	8-19
8-20	Classpath Tab for the JPA Cache Server Executable .....	8-19
8-21	Classpath Tab for the RunEmployee Executable .....	8-21
9-1	Classpath for the CoherenceCacheServer Executable .....	9-5
9-2	Order and Export Tab for Libraries for the Java Build Path .....	9-6
9-3	Classpath for the Interaction Project Cache Server.....	9-6
9-4	Classpath for the DatabaseCache Program.....	9-16
10-1	Class Path for the Security Cache Server.....	10-14
10-2	Arguments Tab for the Security Proxy Server.....	10-15
10-3	Class Path for the Proxy Server.....	10-16
10-4	Classpath Tab for the PasswordExample Program .....	10-16
10-5	Classpath Tab for the AccessControlExample Program .....	10-33

10-6	Class Path for the Cache Server and the Server Running the Proxy Service .....	10-44
10-7	Classpath Tab for the AccessInvocationServiceExample Program .....	10-44
11-1	Main Tab for the Events Server Startup Configuration .....	11-18
11-2	Classpath Tab for the Events Server Startup Configuration .....	11-18
11-3	Main Tab for the Events Client Startup Configuration .....	11-19
11-4	Classpath Tab for the Events Client Startup Configuration .....	11-19
12-1	Creating the CoherenceJCache User Library .....	12-2
12-2	Adding Coherence JCache JARS to the User Libraries .....	12-3
12-3	Coherence Configuration Page for a JCache Project.....	12-3
12-4	Classpath for the DefaultCacheServer .....	12-9
12-5	Creating an Archive File .....	12-12
12-6	Classpath for the Cache Factory Instance.....	12-13
13-1	Creating a New Machine .....	13-4
13-2	Summary of Machines.....	13-5
13-3	Adding a Server to a Machine.....	13-6
13-4	Summary of Servers Page .....	13-7
13-5	Creating a Coherence Cluster .....	13-8
13-6	Specifying a Unicast Listen Port for a Coherence Cluster .....	13-8
13-7	Choosing Coherence Cluster Members .....	13-9
13-8	Summary of Coherence Clusters .....	13-9
13-9	Enabling a Server for Coherence*Web Local Storage.....	13-10
13-10	Starting the Managed WebLogic Servers .....	13-13
13-11	Locating the Deployable Application .....	13-14
13-12	Installing the Deployment as an Application .....	13-14
13-13	Targeting the Deployment.....	13-15
13-14	Optional Settings Page of the Installation Assistant .....	13-16
13-15	Deployments Window Showing the Deployed Application.....	13-17
13-16	Counter Page with Counter Set to 1 .....	13-17
13-17	Counter Page with Counter Set to 2.....	13-18

---

---

# Preface

Oracle Coherence (Coherence) is an in-memory data grid solution that enables organizations to predictably scale mission-critical applications by providing fast access to frequently used data. Data grid software is a middleware that reliably manages data objects in memory across many servers. By automatically and dynamically partitioning data, Coherence enables continuous data availability and transactional integrity, even in the event of a server failure.

Developers can easily take advantage of the features of Coherence using the standard Java collections API to access and modify data, and use the standard JavaBeans event model to receive data change notifications.

## Audience

This tutorial is intended for software developers, architects, and administrators. It describes how to develop applications for the Oracle Coherence data grid.

## Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

### Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit

<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

## Related Documents

For more information, see the following in the Oracle Coherence documentation set:

- *Oracle Fusion Middleware Developing Applications with Oracle Coherence*
- *Oracle Fusion Middleware Developing Remote Clients for Oracle Coherence*
- *Oracle Fusion Middleware Integrating Oracle Coherence*
- *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence\*Web*
- *Oracle Fusion Middleware Managing Oracle Coherence*

- *Oracle Fusion Middleware Administering Oracle Coherence*
- *Oracle Fusion Middleware Securing Oracle Coherence*
- *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*
- *Oracle Fusion Middleware Java API Reference for Oracle Coherence*
- *Oracle Fusion Middleware .NET API Reference for Oracle Coherence*
- *Oracle Fusion Middleware C++ API Reference for Oracle Coherence*

## Conventions

The following text conventions are used in this tutorial:

<b>Convention</b>	<b>Meaning</b>
<b>boldface</b>	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
<code>monospace</code>	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

---

---

# What's New in This Guide

The following topics introduce the new and changed features of Oracle Coherence and other significant changes that are described in this guide, and provides pointers to additional information.

## New and Changed Features for 12c (12.1.3)

Oracle Coherence 12c (12.1.3) includes the following new and changed features for this document.

- Tutorial for JCache, which shows you how to use Coherence with JCache, the Java standard APIs for caching on the Java platform. The key Coherence files which support JCache are `cache-api.jar` and `coherence-jcache.jar`. The `cache-api.jar` file contains the JCache libraries. The `coherence-jcache.jar` file contains the Coherence implementation which is built on top of the JCache library. See [Chapter 12, "Working with JCache."](#)
- An appendix has been added describing the basic features, security, and events examples that are delivered with the Coherence distribution in the `examples.zip` file. These examples (in the Java, C++, and .NET languages) are designed to be built and run on the command line. See [Appendix A, "Coherence Examples in the examples.zip File."](#)

## Other Significant Changes in this Document for 12c (12.1.3)

For 12c (12.1.3), this guide has been updated in several ways. Following are the sections that have been added or changed.

- Revised all of the tutorials in the tutorial to use Coherence 12c (12.1.3).
- Revised the Coherence and JPA tutorial to use the Oracle 12c database instead of the Oracle XE database. It has also been revised to use EclipseLink 2.5.x, TopLink Grid 12c (12.1.3), and the Java Persistence 2.2.0.0 files. See [Chapter 8, "Using JPA with Coherence."](#)
- Revised the tutorial to use the Oracle 12c database instead of the Oracle XE database. It has also been revised to create and configure an Oracle Coherence cache in Eclipse to use EclipseLink 2.5.x, TopLink Grid 12c (12.1.3), and the Java Persistence 2.2.0.0 files. See [Chapter 9, "Interacting with the Cache and the Database."](#)
- Revised the session caching and managed WebLogic Servers tutorial to use WebLogic Server 12c (12.1.3). See [Chapter 13, "Caching Sessions with Managed Coherence Servers."](#)



---

---

# Installing and Configuring Coherence

This chapter describes how to install and set up your environment for running Oracle Coherence 12c (12.1.3) (Coherence).

---

---

**Note:** This tutorial uses the Eclipse Kepler 4.3.1 release with Oracle Enterprise Pack for Eclipse (OEPE) 12.1.3.0.0. For information about configuring Eclipse and OEPE for Coherence-based projects, see [Chapter 2, "Installing and Configuring Eclipse and OEPE with Coherence"](#).

---

---

This chapter contains the following sections:

- [Prerequisites](#)
- [Installing Coherence](#)
- [Testing a Coherence Installation](#)
- [Introducing the Coherence examples.zip File](#)

## Prerequisites

- You must have the privileges to install software and set system environment variables as the `oracle` user, an understanding of how to use a terminal window, including setting environment variables, and creating and moving between directories.
- You must have a working installation of the Oracle JDK or JRE (version 7 or higher). This tutorial uses the Oracle JDK 1.7.0\_25.
- You must also be running one of the following Microsoft Windows operating systems: XP, Vista, 2000, 2003, 2008 or Windows 7.

## Installing Coherence

For information on installing Oracle Coherence, see "Installing Oracle Coherence for Java" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*. This chapter provides instructions for installing Oracle Coherence by using the Oracle Universal Installer. The installer supports both a graphical mode and a silent mode. You can also choose to use the Coherence Quick Installer to perform a silent installation with no options.

---



---

**Note:** This tutorial assumes that you have used the Oracle Universal Installer graphical interface to install Oracle Coherence at `C:\Oracle\Middleware\Oracle_Home`.

---



---

The coherence folder which appears under `Oracle_Home` contains these subfolders:

- `bin`, which contains command scripts
- `doc`, which contains the product documentation
- `examples`, which contains code examples for the Java, .NET, and C++ platforms
- `lib`, which contains the required library files
- `plugins`, which contains Maven plugins that are used to integrate Coherence as part of a Maven build process

## Testing a Coherence Installation

In this exercise, you test whether your Coherence installation can cluster Java processes. This ensures that Coherence-based applications that ship with Coherence will run as expected. If Coherence is not capable of clustering on a single machine, then you must reconfigure your network and firewall settings.

To complete this exercise, you must have Coherence (Java Edition) 12c (12.1.3) installed. See "[Installing Coherence](#)" on page 1-1 for more information.

Coherence uses a variety of network addresses and ports to enable communication between clustered processes. If these addresses and ports are unavailable due to other applications using them, or because of a firewall, then Coherence may be unreliable, may fail to cluster, or may not work at all. By default, Coherence assumes that the network addresses and ports listed in [Table 1-1](#) are available:

**Table 1-1 Network Addresses and Ports Used by Coherence**

Address / Port / Type	Purpose
<code>224.version / version / Multicast</code>	Cluster member discovery and broadcast. The variable <code>version</code> represents the release version of Coherence. For example, the multicast address for the 12.1.3 release is <code>224.12.1.0</code> . The port number also reflects the release version. For example, for the 12.1.3 release, the port number is <code>12100</code> . Designing the address in this way ensures that different versions of Coherence do not cluster with each other by default.
<code>localhost / 8088+ / Unicast</code>	Interprocess communication between cluster members. ( <code>localhost</code> is the local IP address and not the loop back address.)
<code>localhost / 8089 / Unicast</code>	An additional port that is used for unicast communication. By default, the port is assigned as the next available port after the first unicast port.
<code>localhost / 7 / Unicast</code>	The default port of the IpMonitor component that is used for detecting hardware failure of cluster members is "7".

Coherence ships with two simple command-line (shell-based) applications that can be used to determine whether Coherence operates correctly.

- The *cache server* is a simple application that hosts and manages data on behalf of other applications in a cluster.
- The *cache client* is a simple application that enables a developer to access, process, and update cached data within a cluster. It also provides information about the cluster. By executing these applications either on a single host or on several hosts, you can determine whether Coherence is operating correctly locally or across a network.

When an application uses Coherence as shipped, objects placed into Coherence caches are typically stored and managed in-process within the application. However, to increase object availability, Coherence can manage objects in memory but out of the application process. This enables objects to survive possible application outages (either deliberate or accidental). To manage objects in this way, Coherence uses cache servers. The purpose of a Coherence cache server is to manage the application state in a cluster outside the application process. It is similar to a database server, but without the requirement for storage.

## To Test a Coherence Installation—Main Steps

Follow these steps to test a Coherence Installation. The steps are described in detail in the following sections.

1. [Configure and Run the Sample Cache Server Application](#)
2. [Configure and Run the Sample Cache Client Application](#)
3. [Exercise the Sample Cache Client Application](#)
4. [Troubleshooting Cache Server Clustering](#)

## Configure and Run the Sample Cache Server Application

To set up and run the sample cache server application:

1. Open a terminal window and verify that the `PATH` environment variable is set to include the Java JDK or JRE (for example, `C:\Oracle\Middleware\Oracle_Home\jdk1.7.0_25\bin`). If the `PATH` environment variable does not include the JDK or JRE `\bin` folder, then set it as follows:

- a. Set the `JAVA_HOME` environment variable to the base of the JDK or JRE installation.

```
set JAVA_HOME=\Oracle\Middleware\Oracle_Home\jdk1.7.0_25
```

- b. Include `JAVA_HOME\bin` in the `PATH` environment variable.

```
set PATH=%JAVA_HOME%\bin;%PATH%
```

2. Navigate to the folder where Coherence is installed.

```
cd C:\oracle\Middleware\Oracle_Home\coherence\bin
```

Edit the `cache-server.cmd` file to set the `COHERENCE_HOME` environment variable to point to the Coherence installation folder:

```
coherence_home=C:\oracle\Middleware\Oracle_Home\coherence
```

[Example 1-1](#) illustrates the `cache-server.cmd` file with the edited value of the `COHERENCE_HOME` environment variable.

**Example 1-1** *cache-server.cmd* File with an Edited **COHERENCE\_HOME**

```

@echo off
@
@rem This will start a cache server
@
setlocal

:config
@rem specify the Coherence installation directory
set coherence_home=c:\oracle\product\coherence

@rem specify the JVM heap size
set memory=512m

:start
if not exist "%coherence_home%\lib\coherence.jar" goto instructions

if "%java_home%"==" " (set java_exec=java) else (set java_exec=%java_
home%\bin\java)

:launch

if "%1"=="-jmx" (
    set jmxproperties=-Dtangosol.coherence.management=all -Dtangosol.coherence.
management.remote=true
    shift
)

set java_opts=-Xms%memory% -Xmx%memory% %jmxproperties%

%java_exec% -server -showversion %java_opts% -cp "%coherence_home%\lib\coherence.
jar" -Dtangosol.coherence.clusterport=3155 com.tangosol.net.DefaultCacheServer %1

goto exit

:instructions

echo Usage:
echo   ^<coherence_home^>\bin\cache-server.cmd
goto exit

:exit
endlocal
@echo on

```

3. Execute the cache server application that is located in the `coherence\bin` folder.

```
C:\oracle\product\coherence\bin>cache-server.cmd
```

When you start the first cache server, there is a slight delay because the cache server looks for an existing cluster. When the cache server determines that there are no clusters to join, it starts one. On start-up, the cache server produces output similar to the text in [Example 1-2](#).

Several important features are highlighted in the example:

- The Java JDK version number.

- Information about how configuration files are loaded. The default is to load from the `coherence.jar` file:  

```
Loaded operational configuration from
"jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/tangosol-coherence.xml"
```
- The Coherence release number: Oracle Coherence 12.1.3.0.0...
- The Coherence edition: Grid Edition: Development mode
- The multicast address. This address changes with each major Coherence version. Note the 12.1.0 in the address for Coherence 12c (12.1.3):  

```
Group{Address=224.12.1.0, Port=12100, TTL=4}
```
- The Member ID indicates the number of members in your cluster. For the purpose of this exercise, the value should be 1. ThisMember=Member (Id=1.  

```
..
```

### Example 1–2 Output from Starting a Coherence Cache Server

```
C:\Oracle\Middleware\Oracle_Home\coherence\bin>cache-server.cmd
```

```
java version "1.7.0_25"
```

```
Java(TM) SE Runtime Environment (build 1.7.0_25-b17)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)
```

```
2013-11-13 13:34:39.298/1.062 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/tangosol-coherence.xml"
```

```
2013-11-13 13:34:39.398/1.162 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/tangosol-coherence-override-dev.xml"
```

```
2013-11-13 13:34:39.398/1.162 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional configuration override "/tangosol-coherence-override.xml" is not specified
```

```
2013-11-13 13:34:39.408/1.172 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional configuration override "cache-factory-config.xml" is not specified
```

```
2013-11-13 13:34:39.408/1.172 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional configuration override "cache-factory-builder-config.xml" is not specified
```

```
2013-11-13 13:34:39.408/1.172 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional configuration override "/custom-mbeans.xml" is not specified
```

#### Oracle Coherence Version 12.1.3.0.0 Build 48392

**Grid Edition: Development mode**

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

```
2013-11-13 13:34:39.738/1.502 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/coherence-cache-config.xml"
```

```
2013-11-13 13:34:40.155/1.919 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/internal-txn-cache-config.xml"
```

```
2013-11-13 13:34:40.900/2.664 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a): Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
```

```
2013-11-13 13:34:42.310/4.074 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP bound to /130.35.99.13:8088 using SystemDatagramSocketProvider
```

```
2013-11-13 13:34:46.310/8.074 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a): Created a new cluster"cluster:0x47DB" with Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088, MachineId=47251, Location=site:machine:TPFAEFFL-LAP,process:3224, Role=CoherenceServer, Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4)
```

```
2013-11-13 13:34:46.310/8.074 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a): Started cluster Name=cluster:0x47DB
```

```
Group{Address=224.12.1.0, Port=12100, TTL=4}
```

```

MasterMemberSet (
  ThisMember=Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088,
MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:3224, Role=CoherenceServer)
  OldestMember=Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088,
MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:3224, Role=CoherenceServer)
  ActualMemberSet=MemberSet(Size=1
    Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088, MachineId=47251,
Location=site:,machine:TPFAEFL-LAP,process:3224, Role=CoherenceServer)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
  1|12.1.3|2013-11-13 13:34:42.73|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=0
  )
)

TcpRing{Connections=[]}
IpMonitor{Addresses=0}

2013-11-13 13:34:46.420/8.184 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Service Management joined the cluster with senior service member 1
2013-11-13 13:34:46.520/8.284 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-13 13:34:47.052/8.816 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=1): TcpAcceptor now listening for connections on 130.35.99.
13:8088.3
2013-11-13 13:34:47.572/9.336 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): Service DistributedCache joined the cluster with senior service member 1
2013-11-13 13:34:47.612/9.376 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): This member has become the distribution coordinator for MemberSet(Size=1
  Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088, MachineId=47251,
Location=site:,machine:TPFAEFL-LAP,process:3224, Role=CoherenceServer)
)

2013-11-13 13:34:47.652/9.416 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=ReplicatedCache,
member=1): Service ReplicatedCache joined the cluster with senior service member 1
2013-11-13 13:34:47.662/9.426 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=OptimisticCache,
member=1): Service OptimisticCache joined the cluster with senior service member 1
2013-11-13 13:34:47.672/9.436 Oracle Coherence GE 12.1.3.0.0
<D5>(thread=Invocation:InvocationService, member=1): Service InvocationService joined the cluster
with senior service member 1
2013-11-13 13:34:47.672/9.436 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1):
Services
(
  ClusterService{Name=Cluster, State=(SERVICE_STARTED, STATE_JOINED), Id=0, Version=12.1.3,
OldestMemberId=1}
  InvocationService{Name=Management, State=(SERVICE_STARTED), Id=2, Version=12.1.3,
OldestMemberId=1}
  PartitionedCache{Name=DistributedCache, State=(SERVICE_STARTED), LocalStorage=enabled,
PartitionCount=257, BackupCount=1, AssignedPartitions=257, BackupPartitions=0}
  ReplicatedCache{Name=ReplicatedCache, State=(SERVICE_STARTED), Id=4, Version=12.1.3,
OldestMemberId=1}
  Optimistic{Name=OptimisticCache, State=(SERVICE_STARTED), Id=5, Version=12.1.3, OldestMemberId=1}
  InvocationService{Name=InvocationService, State=(SERVICE_STARTED), Id=6, Version=12.1.3,
OldestMemberId=1}
)

Started DefaultCacheServer...

```

---



---

**Note:** By default, Coherence is configured to use multicast to join a cluster and to distribute cluster events. Multicast can also be used to distribute a message efficiently to multiple nodes in the cluster. You can configure Coherence to disable multicast.

---



---

The output of the `cache-server.cmd` file indicates whether you have one or more members in your cluster. The value of `Member Id` should be equal to 1:

```
...
In MasterMemberSet
ThisMember=Member(Id should be equal to 1)
...
```

If `ThisMember` has an `Id` greater than one, it means that your cache server has joined a pre-existing cluster in your network. For the purposes of these exercises, the cluster should contain only one member. Follow the steps in ["Restricting Coherence to Your Own Host"](#) on page 1-15 to restrict Coherence to your own host.

## Configure and Run the Sample Cache Client Application

To set up and run the sample cache client application:

1. Open another terminal window to start the cache client.

Verify that the `PATH` environment variable is set to include the `%JAVA_HOME%\bin` folder. If the `PATH` environment variable does not include the `%JAVA_HOME%\bin` folder, then set the variable as described in ["Configure and Run the Sample Cache Server Application"](#) on page 1-3.

2. Navigate to the `\Oracle\Middleware\Oracle_Home\coherence\bin` folder. Edit the `query.cmd` file to set the `COHERENCE_HOME` variable to point to the Coherence installation folder.

The `query.cmd` file includes a reference to the `JLine` JAR file (`jline.jar`). `JLine` is a Java library that simplifies working with console commands. The `jline.jar` file is included in the `coherence\lib` folder.

[Example 1-3](#) illustrates the `query.cmd` file, with `COHERENCE_HOME=\oracle\product\coherence`. `JLINE_HOME` is set to `%jline_home%\lib`.

### **Example 1-3 query.cmd File with an Edited COHERENCE\_HOME**

```
@echo off
@
@rem This will start a console application
@rem demonstrating the functionality of the Coherence(tm) API
@
setlocal

:config
@rem specify the Coherence installation directory
set coherence_home=C:\oracle\Middleware\Oracle_Home\coherence

@rem specify the jline installation directory
set jline_home=%coherence_home%\lib

@rem specify if the console will also act as a server
```

```

set storage_enabled=false

@rem specify the JVM heap size
set memory=64m

:start
if not exist "%coherence_home%\lib\coherence.jar" goto instructions

if "%java_home%"==" " (set java_exec=java) else (set java_exec=%java_
home%\bin\java)

:launch

if "%storage_enabled%"=="true" (echo ** Starting storage enabled console **) else
(echo **
Starting storage disabled console **)

set java_opts="-Xms%memory% -Xmx%memory%
-Dtangosol.coherence.distributed.localstorage=%storage_enabled%"

"%java_exec%" -server -showversion "%java_opts%" -cp
"%coherence_home%\lib\coherence.jar;%jline_home%\jline.jar"
com.tangosol.coherence.dslquery.QueryPlus %*

goto exit

:instructions

echo Usage:
echo   ^<coherence_home^>\bin\query.cmd
goto exit

:exit
endlocal
@echo on

```

### 3. Execute the `query.cmd` file to start the cache client.

```
query.cmd
```

The output of starting the cache client, illustrated in [Example 1–4](#), displays the basic distributed cache functionality that is built into Coherence. At the end of the output, the `CohQL>` prompt is displayed.

#### **Example 1–4 Output from Starting the Coherence Cache Client**

```

C:\Oracle\Middleware\Oracle_Home\coherence\bin>query.cmd
** Starting storage disabled console **
java version "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b17)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)

Coherence Command Line Tool

CohQL>

```

## Exercise the Sample Cache Client Application

Exercise the cache client application by entering various commands and examining the output.

### 1. Execute the following Coherence commands in the cache client:

- Enter `help` to see the list of commands that are available.
- Enter `create cache "products"` to create a cache named `products`.

The cache `products` implements the `com.tangosol.net.NamedCache` interface. A cluster can have many named caches.

[Example 1–5](#) illustrates that by using the default configuration file (`coherence-cache-config.xml`) within the supplied `coherence.jar` file, a `NamedCache` called `products` is created using the distributed scheme.

### Example 1–5 Output from Starting a Coherence Cache Client

```
CohQL> create cache "products"
2013-11-13 14:00:40.286/219.229 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/tangosol-coherence.xml"
2013-11-13 14:00:40.348/219.291 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/tangosol-coherence-override-dev.xml"
2013-11-13 14:00:40.348/219.291 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a):
Optional configuration override "/tangosol-coherence-override.xml" is not specified
2013-11-13 14:00:40.364/219.307 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a):
Optional configuration override "cache-factory-config.xml" is not specified
2013-11-13 14:00:40.364/219.307 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a):
Optional configuration override "cache-factory-builder-config.xml" is not specified
2013-11-13 14:00:40.364/219.307 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a):
Optional configuration override "/custom-mbeans.xml" is not specified

Oracle Coherence Version 12.1.3.0.0 Build 48392
  Grid Edition: Development mode
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

2013-11-13 14:00:40.723/219.666 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/coherence-cache-config.xml"
2013-11-13 14:00:41.050/219.993 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/internal-txn-cache-config.xml"
2013-11-13 14:00:41.768/220.711 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-11-13 14:00:43.063/222.006 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP
bound to /130.35.99.13:8090 using SystemDatagramSocketProvider
2013-11-13 14:00:43.671/222.614 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
Failed to satisfy the variance: allowed=16, actual=31
2013-11-13 14:00:43.671/222.614 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
Increasing allowable variance to 17
2013-11-13 14:00:44.014/222.957 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=n/a):
Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3224, Role=CoherenceServer) joined Cluster with senior
member 1
2013-11-13 14:00:44.030/222.989 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
This Member(Id=2, Timestamp=2013-11-13 14:00:43.812, Address=130.35.99.13:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3896, Role=TangosolCoherenceQueryPlus, Edition=Grid
Edition, Mode=Development, CpuCount=4, SocketCount=4) joined cluster "cluster:0x47DB" with senior
```

```

Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3224, Role=CoherenceServer, Edition=Grid Edition,
Mode=Development, CpuCount=4, SocketCount=4)
2013-11-13 14:00:44.186/223.129 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Started cluster Name=cluster:0x47DB

Group{Address=224.12.1.0, Port=3155, TTL=4}

MasterMemberSet (
  ThisMember=Member(Id=2, Timestamp=2013-11-13 14:00:43.812, Address=130.35.99.13:8090,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:3896, Role=TangosolCoherenceQueryPlus)
  OldestMember=Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:3224, Role=CoherenceServer)
  ActualMemberSet=MemberSet(Size=2
    Member(Id=1, Timestamp=2013-11-13 13:34:42.73, Address=130.35.99.13:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3224, Role=CoherenceServer)
    Member(Id=2, Timestamp=2013-11-13 14:00:43.812, Address=130.35.99.13:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3896, Role=TangosolCoherenceQueryPlus)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
  1|12.1.3|2013-11-13 13:34:42.73|JOINED,
  2|12.1.3|2013-11-13 14:00:43.812|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=0
  )
)

TcpRing{Connections=[1]}
IpMonitor{Addresses=0}

2013-11-13 14:00:44.248/223.191 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=2): Service Management joined the cluster with senior service member 1
2013-11-13 14:00:44.264/223.207 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=2):
Loaded Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/reports/report-group.xml"
2013-11-13 14:00:44.685/223.628 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=2): TcpAcceptor now listening for connections on 130.35.99.
13:8090.3
2013-11-13 14:00:45.138/224.081 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=2): Service DistributedCache joined the cluster with senior service member 1

CohQL>

```

2. Execute the following commands at the CohQL> prompt in the cache client. For definitions of these commands, see "Using Coherence Query Language" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

- Insert an entry (key-value pair) into the products cache:

```
insert into "products" key "television" value "ID-5070"
```
- Change the value of the key:

```
update "products" set value() = "ID-5080" where key() like "television"
```
- Retrieve the values in the cache:

```
select * from "products"
```
- Retrieve the value of a key that does not exist. An empty result set will be returned:

```
select key(), value() from "products" where key() is "radio"
```

- Delete an existing key in the cache. An empty result set will be returned:

```
delete from "products" where key() = "television"
```

- Delete the contents of the `products` cache. An empty result set will be returned:

```
delete from "products"
```

- Destroy the `products` cache:

```
drop cache "products"
```

- Re-create the `products` cache:

```
create cache "products"
```

- Insert more entries into the cache:

```
insert into "products" key "television" value "ID-5080"  
insert into "products" key "radio" value "ID-5090"  
insert into "products" key "MP3 Player" value "ID-5100"  
insert into "products" key "laptop" value "ID-5110"
```

- Retrieve the keys and values in the `products` cache:

```
select key(), value() from "products"
```

- Save a serialized representation of the cache in a file:

```
backup cache "products" to "products.bkup"
```

- Delete a key from the cache:

```
delete from "products" where key() = "television"
```

- Retrieve the cache contents again, notice that the deleted key and value will not be present:

```
select key(), value() from "products"
```

- Delete the contents of the cache:

```
delete from "products"
```

- Retrieve the contents of the cache. An empty result set will be returned:

```
select * from "products"
```

- Restore the cache contents from the backup file:

```
restore cache "products" from file "products.bkup"
```

- Retrieve the cache contents. Note that all of the entries will be restored and returned:

```
select key(), value() from "products"
```

- Destroy the `products` cache:

```
drop cache "products"
```

- Exit the command-line tool:

bye

[Example 1–6](#) illustrates the output of each of these commands.

**Example 1–6 Exercising Coherence Commands**

```
CohQL> create cache "products"

CohQL> insert into "products" key "television" value "ID-5070"

CohQL> update "products" set value() = "ID-5080" where key() like "television"
Results
television: true

CohQL> select * from "products"
Results
ID-5080

CohQL> select key(), value() from "products" where key() is "radio"
Results

CohQL> delete from "products" where key() = "television"
Results

CohQL> delete from "products"
Results

CohQL> drop cache "products"

CohQL> create cache "products"

CohQL> insert into "products" key "television" value "ID-5080"

CohQL> insert into "products" key "radio" value "ID-5090"

CohQL> insert into "products" key "MP3 Player" value "ID-5100"

CohQL> insert into "products" key "laptop" value "ID-5110"

CohQL> select key(), value() from "products"
Results
"television", "ID-5080"
"radio", "ID-5090"
"MP3 Player", "ID-5100"
"laptop", "ID-5110"

CohQL> backup cache "products" to "products.bkup"
WARNING: The backup command should not be used on active data set,
as it makes no provisions that ensure data consistency during the backup.
Please see the documentation for more detailed information.

CohQL> delete from "products" where key() = "television"
Results

CohQL> select key(), value() from "products"
Results
"radio", "ID-5090"
"MP3 Player", "ID-5100"
"laptop", "ID-5110"
```

```

CohQL> delete from "products"
Results

CohQL> select * from "products"
Results

CohQL> restore cache "products" from file "products.bkup"

CohQL> select key(), value() from "products"
Results
"television", "ID-5080"
"radio", "ID-5090"
"MP3 Player", "ID-5100"
"laptop", "ID-5110"

CohQL> drop cache "products"

CohQL> bye

```

In the cache server window, you should see a response similar to the following, indicating that the client (Member 2) has left the cluster:

```

...
2013-11-13 14:08:50.545/2052.309 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Cluster, member=1): TcpRing disconnected from Member(Id=2,
Timestamp=2013-11-13 14:00:43.812, Address=130.35.99.13:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3896,
Role=TangosolCoherenceQueryPlus) due to a peer departure; removing the member.
2013-11-13 14:08:50.545/2052.309 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Cluster, member=1): Member(Id=2, Timestamp=2013-11-13 14:08:50.545,
Address=130.35.99.13:8090, MachineId=47251, Location=site:,machine:TPFAEFFL-
LAP,process:3896, Role=TangosolCoherenceQueryPlus) left Cluster with senior
member 1
2013-11-13 14:08:50.545/2052.309 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:Management, member=1): Member 2 left service Management with
senior member 1
2013-11-13 14:08:50.545/2052.309 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache, member=1): Member 2 left service DistributedCache
with senior member 1

```

3. Open another terminal window and set the PATH environment variable to include %JAVA\_HOME% and %JAVA\_HOME%\bin. Enter the query.cmd command in the new terminal window to start an instance of the cache client.
4. Restart the first client with the query.cmd command. Enter the create cache "products" command. The cache server terminal window displays a message similar to the following that describes where the first client is running. member 1 is the cache server and member 3 is the restarted first client.

```

2013-11-13 14:15:24.217/27.307 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:Management, member=3): Service Management joined the cluster
with senior service member 1
2013-11-13 14:15:24.248/27.338 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=3): Loaded Reporter configuration from
"jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-13 14:15:24.654/27.744 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=3): TcpAcceptor now listening for
connections on 130.35.99.13:8090.3
2013-11-13 14:15:25.106/28.196 Oracle Coherence GE 12.1.3.0.0 <D5>

```

```
(thread=DistributedCache, member=3): Service DistributedCache joined the cluster with senior service member 1
```

```
CohQL>
```

5. Enter the `create cache "products"` command in the new terminal window to connect to the products cache. It enters the cluster as Member 4. Try to get and put values in different sessions. Notice that each client can observe changes made by the other client.

6. Terminate one of the `query.cmd` client shells (`bye`). Note that the other shell displays a message indicating that the member has left the cluster, for example:

```
2013-11-13 14:23:54.085/537.175 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Cluster, member=3): TcpRing disconnected from Member(Id=4,
Timestamp=2013-11-13 14:22:39.583, Address=130.35.99.13:8092, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:5540,
Role=TangosolCoherenceQueryPlus) due to a peer departure; removing the member.
2013-11-13 14:23:54.085/537.175 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Cluster, member=3): Member(Id=4, Timestamp=2013-11-13 14:23:54.085,
Address=130.35.99.13:8092, MachineId=47251, Location=site:,machine:TPFAEFFL-
LAP,process:5540, Role=TangosolCoherenceQueryPlus) left Cluster with senior
member 1
2013-11-13 14:23:54.085/537.175 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:Management, member=3): Member 4 left service Management with
senior member 1
2013-11-13 14:23:54.085/537.175 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache, member=3): Member 4 left service DistributedCache
with senior member 1
```

7. If you terminate each of the cache clients (`bye`), and then restart them, note that data from the previous session is still available. This is because the data is held in the cache server.
8. If you start another Coherence cache server, and then terminate the initial one that was started (using `Ctrl+C` or by closing the command window), note that the data is still available.
9. Terminate both the `query.cmd` shells and `cache-server.cmd` shell. Restart `query.cmd`. Create a cache called `test` using the command `create cache "test"`. Try to put a value into the cache, as before. Because the cache client is configured to start in storage-disabled mode, you will receive a response similar to the following:

```
com.tangosol.net.RequestPolicyException: No storage-enabled nodes exist for
service DistributedCache
```

10. Start a cache server by running the `cache-server.cmd` file. Try to insert a value again. This time, the value will be accepted.
11. Terminate all cache servers.

## Troubleshooting Cache Server Clustering

If the value of Member ID in the `cache-server.cmd` output is anything other than 1, then this indicates that the cache server has clustered with one or more other cache servers or processes running Coherence. These servers or processes can be running on the network or running locally on your host. Though this is the default behavior for Coherence—to cluster with other processes running Coherence locally or on a

network—it is strongly advised that while you perform this tutorial, you restrict Coherence to your own host.

---

**Note:** To restrict Coherence to a single host, the cache server and cache client executable files used in this tutorial use the `tangosol.coherence.clusterport` command line property set to 3155.

---

### Restricting Coherence to Your Own Host

The value of the Member ID in the output of the `cache-server.cmd` cache server command indicates whether you have one or more members in your cluster. For the purpose of this tutorial, the value of Member ID must be equal to 1:

```
...
In MasterMemberSet
ThisMember=Member(Id should be equal to 1)
...
```

If the value of Member ID is greater than 1, it means that multiple clusters are being formed in your subnet.

There are several ways to restrict Coherence to your own host. The easiest way is to use the `tangosol.coherence.clusterport` system property to declare a unique cluster port value in your cache server startup file. For example, add the following Java option to your `cache-server.cmd` file. The value assigned to the system property can be any unique value, such as the last four digits of your telephone number.

```
-Dtangosol.coherence.clusterport=3155
```

### Advanced Steps to Restrict Coherence to Your Own Host

If you follow the steps in "[Restricting Coherence to Your Own Host](#)" on page 1-15 and the `cache-server.cmd` command still fails to return a Member Id value of 1, then there might be additional problems to resolve.

Disconnect from the network or disable networking on your host. If errors or exceptions occur when starting the cache server, your network settings might need to be modified. Try each of the following one at a time, restarting the cache server after each attempt:

- If connected to a Virtual Private Network (VPN), then disconnect from it. By default, most VPN are not configured to permit multicast and some unicast traffic. In this environment, Coherence might not work, if left in the configuration in which it was shipped. Coherence can be configured to run across a VPN, but this requires some advanced settings.
- If you run a firewall, configure it to enable the specified addresses and ports.
- If you still experience problems, disconnect from all networks. This includes wireless and wired networks.
- If all the preceding options fail, set up Coherence to run on a single host.

## Introducing the Coherence examples.zip File

The Coherence distribution provides an `examples.zip` file that contains many of the examples found in this tutorial. They demonstrate many of the cache access, processing, and security features in Coherence. The examples are designed to be built

and run from the command line. The output of the examples is directed to standard output (`stdout`).

[Appendix A, "Coherence Examples in the examples.zip File"](#) provides information on building and running the examples. It also provides minimal documentation on the code in the `examples.zip` file. More detailed information about the code is embedded as comments in the individual code files.

There are a number of differences between the examples in the `examples.zip` file and the examples in this tutorial:

- The examples in the `examples.zip` must be built and run from the command line. This tutorial uses an IDE to compile and run the code.
- The examples in the `examples.zip` file demonstrate how to use basic Coherence functionality and security features in all supported languages (Java, .NET, and C++). The tutorial covers only Java implementations.
- The Java examples in the `examples.zip` file are only a subset of the Java examples presented in the tutorial.
- The Java files in the `examples.zip` file are similar to the files used in this tutorial. In many instances, the code in the tutorial has been simplified for demonstration purposes.

You can obtain the `examples.zip` file by performing a full Coherence installation with the `coherence_version.jar` or `wls_version.jar` file.

If you have already installed Coherence but without the examples, you can obtain the `examples.zip` file by running the `coherence_quick_supp_version.jar` supplemental installer file. The supplemental installer contains only API documentation and examples.

Note that the `coherence_quick_version.jar` quick installer does not install the examples.

---

---

# Installing and Configuring Eclipse and OEPE with Coherence

This chapter describes how to set up the Kepler release of the Eclipse IDE for Java EE Developers (Eclipse) and Oracle Enterprise Pack for Eclipse 12.1.3.0.0 (OEPE) to build and run Coherence-based Java applications.

This chapter contains the following sections:

- [Installing Eclipse and OEPE](#)
- [Configuring Eclipse and OEPE for Coherence](#)
- [Creating a New Project in the Eclipse IDE](#)
- [Launching a Cache Server in the Eclipse IDE](#)
- [Learning Eclipse IDE Basics](#)

## Installing Eclipse and OEPE

You can download an "all-in-one" version of OEPE (12.1.3.0.0) that bundles a preconfigured version of Eclipse Kepler and the OEPE plugins. Oracle recommends installing the "all-in-one" version.

As an alternative, you can download the OEPE distribution by itself and install it into an existing Eclipse Kepler installation. The OEPE installer includes Oracle WebLogic Server, Oracle Coherence, and the Oracle ADF Runtime.

To download either version, select the **Downloads** tab on the Oracle Enterprise Pack for Eclipse page:

<http://www.oracle.com/technetwork/developer-tools/eclipse/overview/index.html>

Follow the instructions in the installer. You can also find installation instructions at this URL:

[http://docs.oracle.com/cd/E27086\\_01/help/oracle.eclipse.tools.common.doc/html/install.html](http://docs.oracle.com/cd/E27086_01/help/oracle.eclipse.tools.common.doc/html/install.html)

This tutorial assumes that you will be installing Eclipse into an `eclipse` folder that you have created at the file system root, for example:

```
C:\eclipse\*
```

## Configuring Eclipse and OEPE for Coherence

To start and configure Eclipse for use with Coherence:

1. Open a terminal window and verify that the PATH environment variable is set to include the Java JDK or JRE, for example: C:\Oracle\Middleware\Oracle\_Home\jdk1.7.0\_25\bin. Note that you must have a working installation of Java JDK or JRE version 7 or higher.

If the PATH environment variable does not include the Java JDK or JRE \bin folder, then set the variable as follows:

- a. Set the JAVA\_HOME environment variable to the base of the JDK or JRE installation, for example:

```
set JAVA_HOME=\Oracle\Middleware\Oracle_Home\jdk1.7.0_25
```

- b. Include %JAVA\_HOME%\bin in the PATH environment variable, for example:

```
set PATH=%JAVA_HOME%\bin;%PATH%
```

2. Start Eclipse.

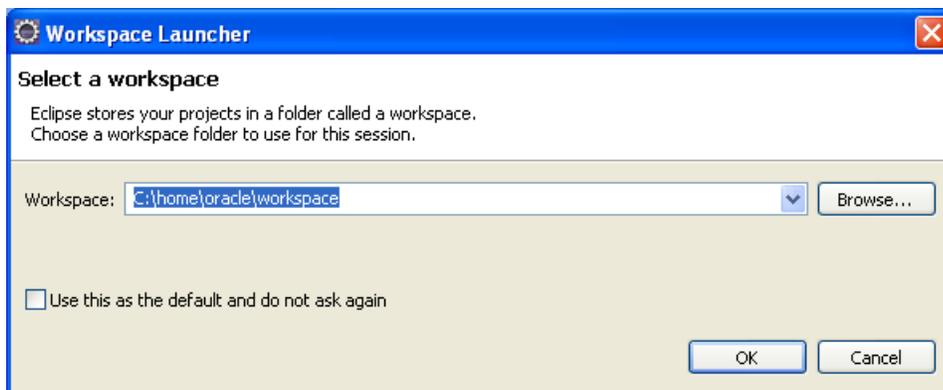
The eclipse executable is eclipse.exe. If you extracted Eclipse into a directory called eclipse, you will find eclipse.exe here:

```
C:\eclipse\eclipse.exe
```

3. If Eclipse prompts you to set or select a workspace in the **Workspace Launcher** dialog box, enter C:\home\oracle\workspace.

Figure 2–1 illustrates the **Workspace Launcher** dialog box with the path C:\home\oracle\workspace selected.

**Figure 2–1 Workspace Launcher Dialog Box**



4. After Eclipse starts, select the Java EE perspective:
  - Select **Window** then **Open Perspective**, then **Java EE**, or
  - Select the Java EE perspective icon in the menu bar

Figure 2–2 illustrates the location of the Java EE perspective icon.

Figure 2–2 The Java EE Perspective in the Menu Bar

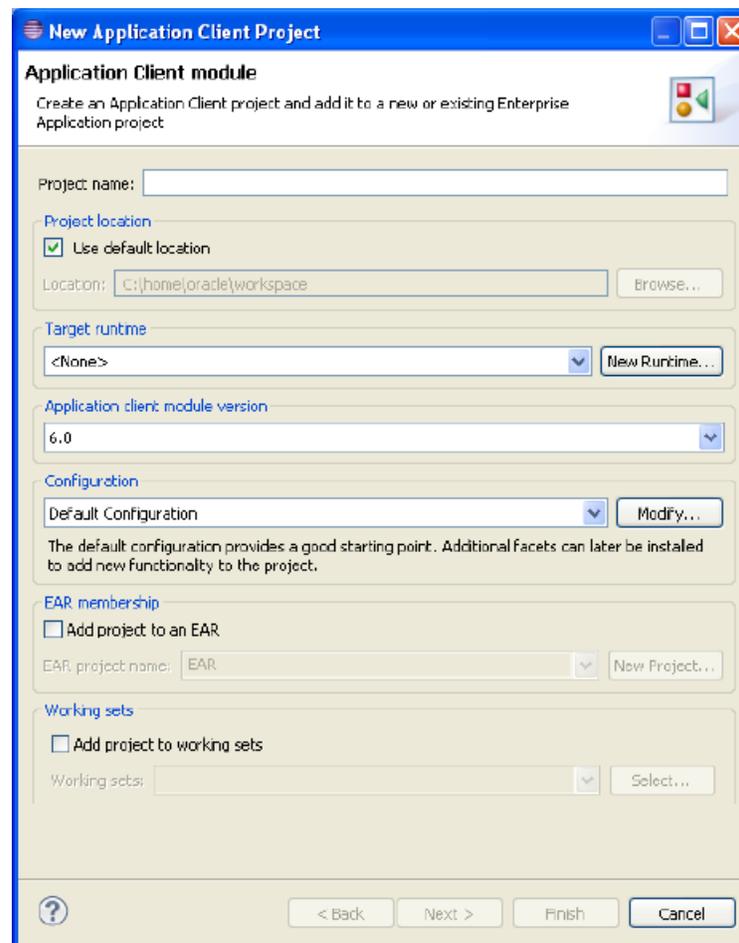


## Creating a New Project in the Eclipse IDE

To create a new project in the Eclipse IDE:

1. Select **File** then **New** then **Application Client Project**.
2. In the **New Application Client Project** dialog box, enter a value, **Coherence** for example, as the **Project name**.

Figure 2–3 New Application Client Project Dialog Box

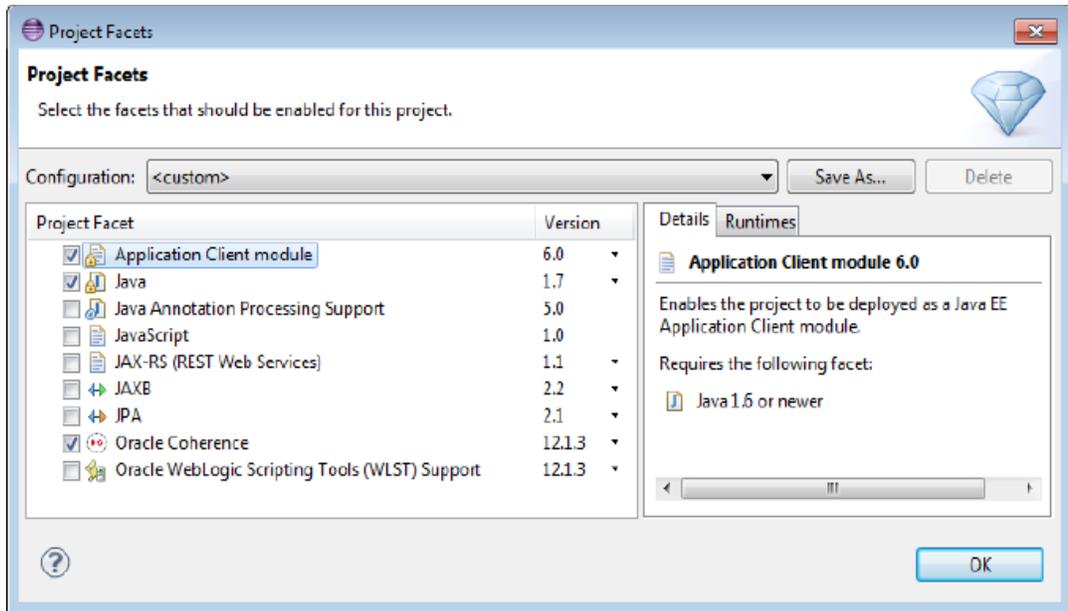


3. In the **Configuration** section, click **Modify**. In the **Project Facets** dialog box, select the Oracle Coherence check box. Select version **12.1.3** from the drop-down list if it is not already displayed.

Adding the Oracle Coherence facet automatically adds the `coherence.jar` to your project class path. It also makes these commonly used configuration files available in the **Project Explorer**:

- `coherence-cache-config.xml`, the default cache configuration file.
- `pof-config.xml`, the configuration file for Portable Object Format serialization.
- `tangosol-coherence-override.xml`, the override file for operational and runtime settings used by Coherence.

**Figure 2-4** *Selecting Oracle Coherence in the Project Facets Dialog Box*



4. Click **Save As** to enter a name for the configuration in the **Save Preset** dialog box. For example, enter `CoherenceConfig` in the **Name** field and `Includes Coherence Facet` in the **Description** field.

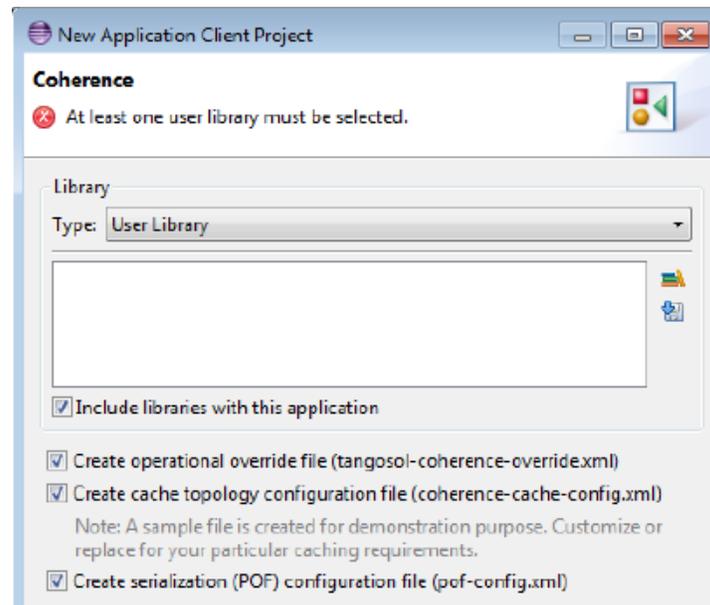
Figure 2-5 illustrates the **Save Preset** dialog box.

**Figure 2-5** *Specifying a Configuration Name in the Save Preset Dialog Box*



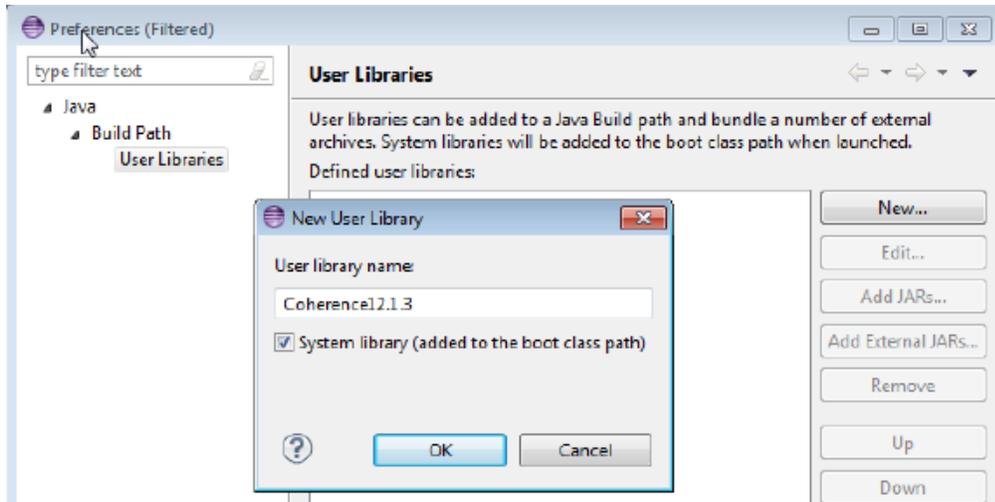
- Click **OK** to close the **Save Preset** dialog box. Click **OK** to close the **Project Facets** dialog box.
5. Click **Next** in the **Java** page of the **New Application Client Project** dialog box to accept the defaults.
  6. Deselect the **Create a default Main class** in the **Application Client Module** page. Click **Next**.
  7. In the **Coherence** page, illustrated in [Figure 2–6](#), add the Coherence 12.1.3 library as a User Library to the project.

**Figure 2–6** *Creating a Coherence User Library*



- a. Click the **Manage Libraries...** icon, then click **New** in the **Preferences** dialog box.
- b. Enter **Coherence12.1.3** in the **New User Library** dialog box, as illustrated in [Figure 2–7](#). Select the **System library (added to the boot class path)** check box. Click **OK** to close the **New User Library** dialog box.

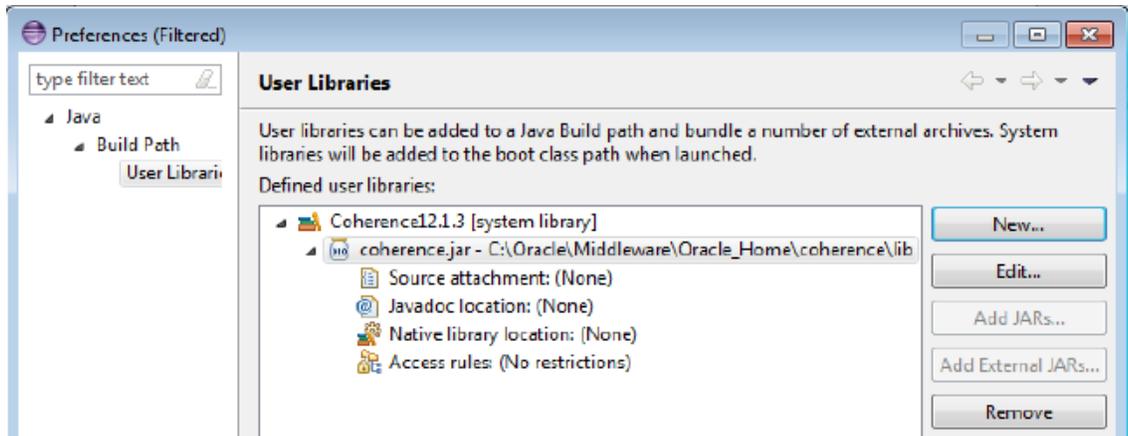
**Figure 2–7 Naming the Coherence Library**



- c. In the **Preferences** dialog box, click **Add External JARs** to add the coherence.jar file to the library. Navigate to the location of the coherence.jar file in the Coherence distribution that you downloaded to your file system in "Installing Coherence" on page 1-1.

The **Preferences** dialog box should look similar to [Figure 2–8](#). Click **OK** to close the **Preferences** dialog box.

**Figure 2–8 The coherence.jar File Defined in the Coherence User Library**



8. Select the **Coherence12.1.3** library in the **Coherence** page and click **Finish**.

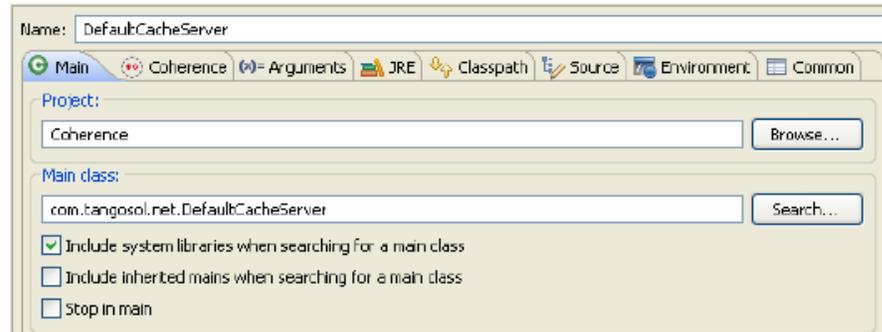
The new project and its associated files appear in the **Project Explorer** window in the Eclipse IDE.

## Launching a Cache Server in the Eclipse IDE

1. Right click the project in the Eclipse IDE. Select **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, select **Oracle Coherence** then the **New launch configuration** icon. Enter DefaultCacheServer as the name for the cache server configuration.

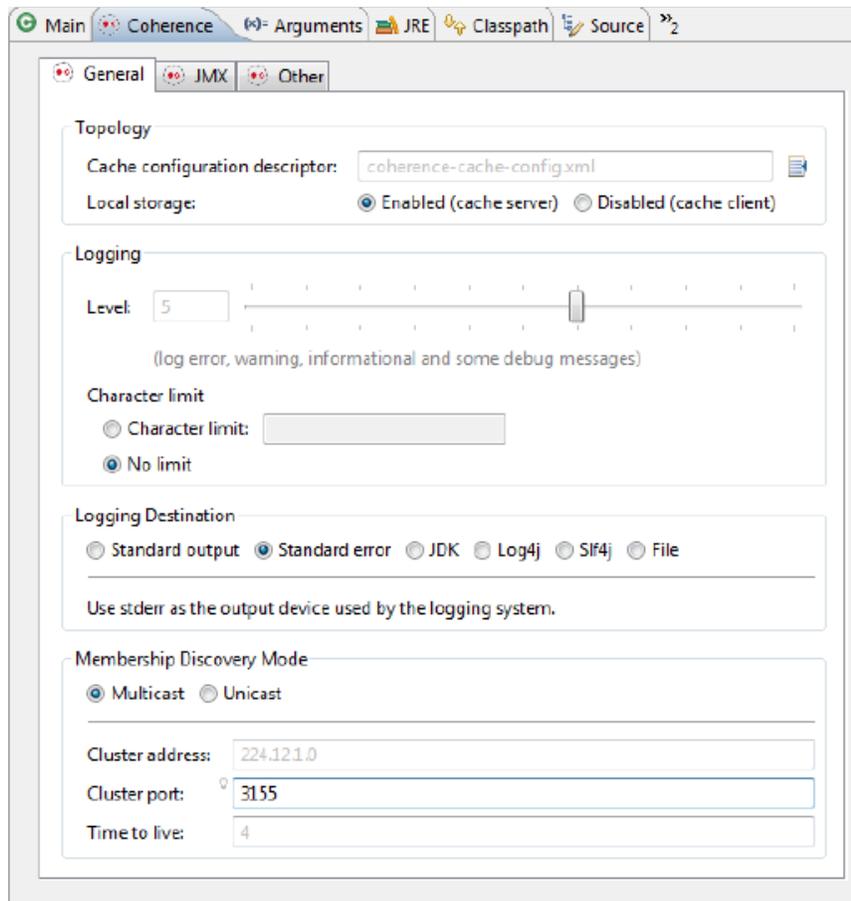
2. Under **Project**, click **Browse** and select the name of the project from the **Project Selection** dialog box.
3. Under **Main class**, select the **Include system libraries when searching for a main class** checkbox. Click the **Search** button and enter `DefaultCacheServer` in the **Select Main Type** dialog box. Select `com.tangosol.net.DefaultCacheServer` and click **OK**. Click **Apply**. The **Main** tab should look similar to [Figure 2–9](#).

**Figure 2–9 Main Tab in the Run Configurations Dialog Box**



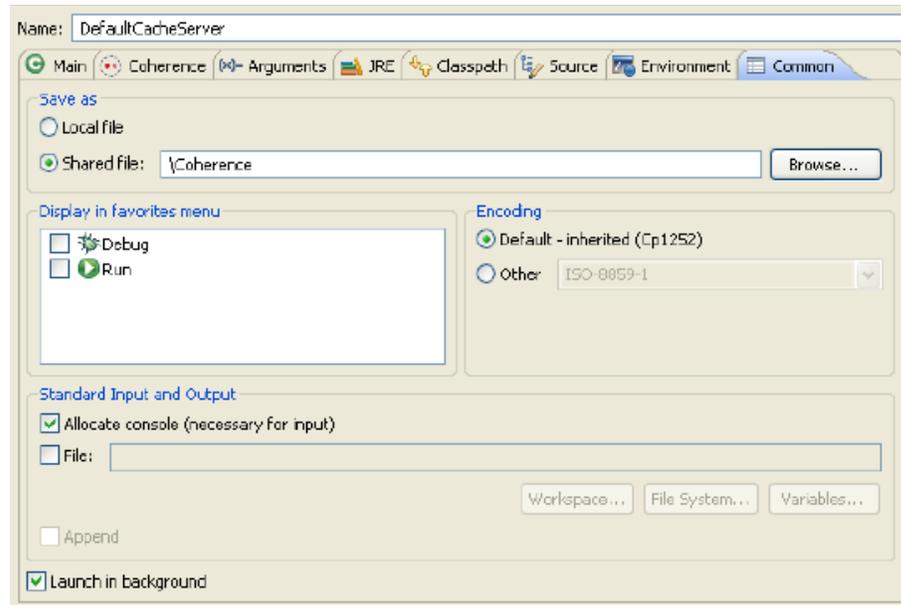
4. In the **Coherence** tab, select the **General** tab. Click the **Browse** icon to navigate to the cache configuration file if it is not already selected. Select local storage to be enabled (cache server). Enter a unique value, such as 3155 for the **Cluster port**. Click **Apply**. The **Coherence** tab should look similar to [Figure 2–10](#).

**Figure 2–10** Coherence Tab of the Run Configurations Dialog Box



5. Open the **Arguments** tab. Enter `-showversion` in the **VM Arguments** field. Click **Apply**.
6. Open the **Common** tab of the dialog box. Click the **Shared file** radio button and click **Browse** to navigate to the project. Click **Apply**. The **Common** tab should look similar to [Figure 2–11](#).

**Figure 2–11 Common Tab of the Run Configurations Dialog Box**



7. Click **Run** to start the cache server. The cache server should start and display output similar to [Example 2–1](#).

**Example 2–1 Cache Server Output in the Eclipse Console Window**

```

java version "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b17)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)

2013-11-14 11:47:51.241/0.562 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence.xml"
2013-11-14 11:47:51.366/0.687 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence-override-dev.xml"
2013-11-14 11:47:51.444/0.765 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from
"file:/C:/home/oracle/workspace/Coherence/build/classes/tangosol-coherence-override.xml"
2013-11-14 11:47:51.459/0.780 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-config.xml" is not specified
2013-11-14 11:47:51.459/0.780 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-builder-config.xml" is not specified
2013-11-14 11:47:51.459/0.780 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "/custom-mbeans.xml" is not specified

Oracle Coherence Version 12.1.3.0.0 Build 48392
Grid Edition: Development mode
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

2013-11-14 11:47:52.224/1.545 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/coherence-cache-config.xml"
2013-11-14 11:47:52.396/1.717 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/internal-txn-cache-config.xml"
2013-11-14 11:47:53.145/2.466 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
    
```

```
2013-11-14 11:47:54.128/3.449 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP
bound to /10.159.162.203:8088 using SystemDatagramSocketProvider
2013-11-14 11:47:58.107/7.428 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
Created a new cluster "cluster:0x47DB" with Member(Id=1, Timestamp=2013-11-14 11:47:54.535,
Address=10.159.162.203:8088, MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:7096,
Role=CoherenceServer, Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4)
2013-11-14 11:47:58.107/7.428 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Started cluster Name=cluster:0x47DB

Group{Address=224.12.1.0, Port=3155, TTL=4}

MasterMemberSet (
  ThisMember=Member(Id=1, Timestamp=2013-11-14 11:47:54.535, Address=10.159.162.203:8088,
MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:7096, Role=CoherenceServer)
  OldestMember=Member(Id=1, Timestamp=2013-11-14 11:47:54.535, Address=10.159.162.203:8088,
MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:7096, Role=CoherenceServer)
  ActualMemberSet=MemberSet(Size=1
    Member(Id=1, Timestamp=2013-11-14 11:47:54.535, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFL-LAP,process:7096, Role=CoherenceServer)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
  1|12.1.3|2013-11-14 11:47:54.535|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=0
  )
)

TcpRing{Connections=[]}
IpMonitor{Addresses=0}

2013-11-14 11:47:58.138/7.459 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Service Management joined the cluster with senior service member 1
2013-11-14 11:47:58.139/7.460 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-14 11:47:58.623/7.944 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=1): TcpAcceptor now listening for connections on 10.159.
162.203:8088.3
2013-11-14 11:47:59.107/8.428 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): Service DistributedCache joined the cluster with senior service member 1
2013-11-14 11:47:59.153/8.474 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): This member has become the distribution coordinator for MemberSet(Size=1
  Member(Id=1, Timestamp=2013-11-14 11:47:54.535, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFL-LAP,process:7096, Role=CoherenceServer)
)
2013-11-14 11:47:59.153/8.474 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=ReplicatedCache,
member=1): Service ReplicatedCache joined the cluster with senior service member 1
2013-11-14 11:47:59.154/8.475 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=OptimisticCache,
member=1): Service OptimisticCache joined the cluster with senior service member 1
2013-11-14 11:47:59.154/8.475 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:InvocationService, member=1): Service InvocationService joined the cluster with
senior service member 1
2013-11-14 11:47:59.154/8.475 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1):
Services
(
  ClusterService{Name=Cluster, State=(SERVICE_STARTED, STATE_JOINED), Id=0, Version=12.1.3,
OldestMemberId=1}
  InvocationService{Name=Management, State=(SERVICE_STARTED), Id=2, Version=12.1.3,
OldestMemberId=1}
```

```

    PartitionedCache{Name=DistributedCache, State=(SERVICE_STARTED), LocalStorage=enabled,
PartitionCount=257, BackupCount=1, AssignedPartitions=220, BackupPartitions=0}
    ReplicatedCache{Name=ReplicatedCache, State=(SERVICE_STARTED), Id=4, Version=12.1.3,
OldestMemberId=1}
    Optimistic{Name=OptimisticCache, State=(SERVICE_STARTED), Id=5, Version=12.1.3, OldestMemberId=1}
    InvocationService{Name=InvocationService, State=(SERVICE_STARTED), Id=6, Version=12.1.3,
OldestMemberId=1}
)

```

Started DefaultCacheServer...

## Learning Eclipse IDE Basics

This section describes common tasks that are a part of building projects in the Eclipse IDE.

- [Creating a Java Class](#)
- [Creating a Runtime Configuration](#)
- [Stopping Cache Servers](#)

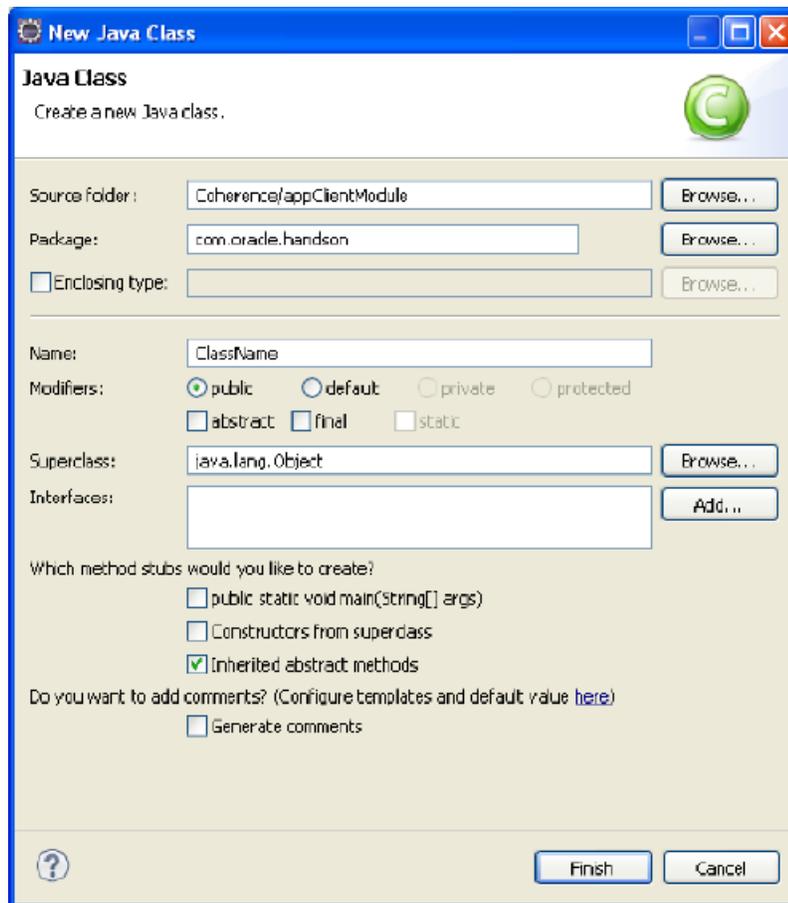
### Creating a Java Class

To create a new Java class in the Eclipse IDE:

1. Right-click the project entry in the **Project Explorer** window. Select **New** then **Class**.
2. In the **New Java Class** dialog box, enter a package name. In this tutorial, the value will typically be `com.oracle.handson`.
3. Enter the name of the class in the **Name** field.
4. Under **Which method stubs would you like to create?:**
  - Select the **public static void main(String[] args)** if you want the file to be runnable.
  - Select **Constructors from superclass** check box if you want stubs of the constructors from the new class's superclass to be added.
  - **Inherited abstract methods** should be selected by default. This option adds stubs of any abstract methods from superclasses or methods of interfaces that need to be implemented.
5. Click **OK** to create the Java class.

[Figure 2–12](#) illustrates the **New Java Class** dialog box.

Figure 2–12 Defining a New Java Class

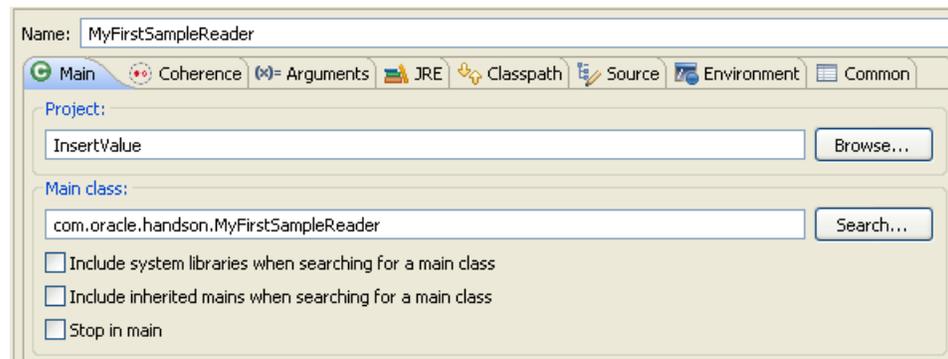


## Creating a Runtime Configuration

To create a runtime configuration for a runnable file:

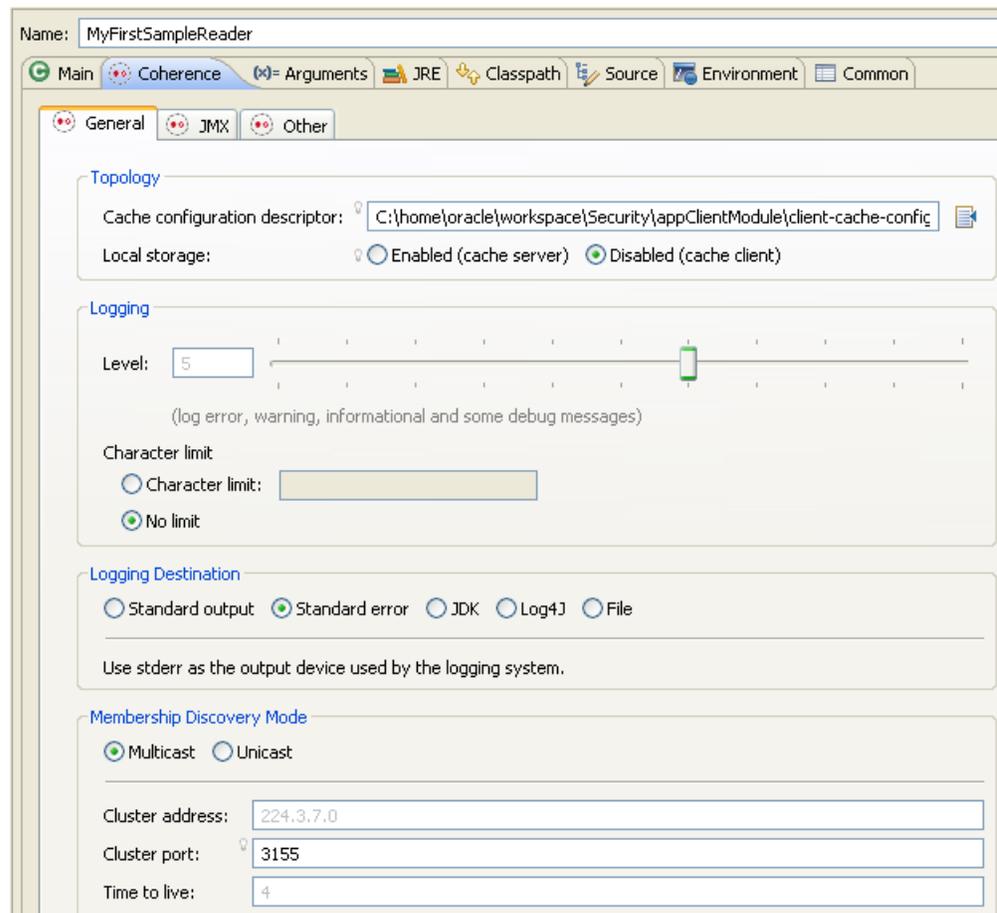
1. Right click the name of the runnable file in the **Project Explorer** window. Select **Run As** then **Run Configurations**.
2. Click **Oracle Coherence**, then the **New Launch Configuration** icon. Select the **Java Application** node. Click the **New Launch Configuration** button.
3. Enter a name for the new configuration. Under **Project** in the **Main** tab, click the **Browse** button to navigate to the name of the current project. Click the **Search** button to navigate to the name of the runnable file for which you are creating the runtime configuration. Click **Apply**.

This figure illustrates the **Main** tab of the **Run Configurations** dialog box.

**Figure 2–13 Defining a Launch Configuration for a Runnable File**

4. Click the **Coherence** tab. Under the **General** tab, you can set many of the more commonly used VM runtime arguments for the configuration. Examples of runtime arguments include the path to the cache configuration descriptor, and the local storage, log level, and cluster port values.

Figure 2–14 illustrates the **General** tab of the **Coherence** tab in the **Run Configurations** dialog box.

**Figure 2–14 Setting Runtime Arguments for the Launch Configuration**

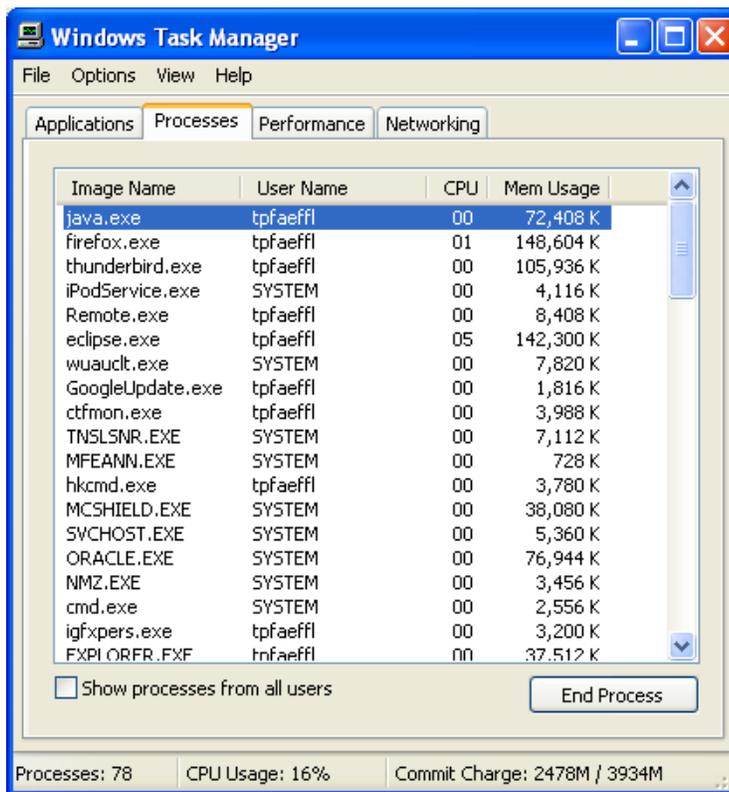
## Stopping Cache Servers

In the Eclipse IDE, you can stop any running cache server or client by clicking the **Terminate** and **Remove all terminated launches** icons in the Eclipse Console.

However, if you look in the Windows Task Manager window, you might notice that the Java process (`java.exe`) associated with the server or client is still running. To prevent any errors being thrown when the server or client is restarted, you must delete its associated Java process. Select the Java process in the Windows Task Manager window and click **End Process**.

Figure 2–15 illustrates the Windows Task Manager window with the Java process selected.

**Figure 2–15** Java Process in the Windows Task Manager



---

---

## Accessing the Data Grid from Java

In this exercise, you develop a simple, Java console-based application to access, update, and remove simple types of information from a Coherence clustered cache. You also are introduced to the Coherence `NamedCache` and `CacheFactory` Java APIs.

Using the Eclipse IDE, you will perform the following tasks:

- Create a new project
- Create a new named cache (`NamedCache` object)
- Put information into the cache and then retrieve it
- Retrieve information about the cache

This chapter contains the following sections:

- [Introduction](#)
- [Creating Your First Coherence-Based Java Program](#)
- [Creating Your First Coherence-Based Java Application](#)

### Introduction

All Coherence caches are named, have a lifetime scoped by the cluster instance in which they exist, and implement the `com.tangosol.net.NamedCache` interface. The `NamedCache` interface is an extension of the `java.util.Map` interface and holds data and resources that are shared among the cluster members. Each `NamedCache` object holds data as key/value pairs. Keys and values can be both simple and complex object types. The `NamedCache` interface provides extensions to the `Map` interface, such as locking and synchronization, storage integration, queries, event aggregations, and transactions. [Table 3-1](#) describes some of the commonly used methods within the `NamedCache` interface.

**Table 3-1** *Methods in the NamedCache Interface*

Method Name	Description
<code>void clear()</code>	Removes all entries from the <code>NamedCache</code> object.
<code>boolean containsKey(Object key)</code>	Returns <code>true</code> if the <code>NamedCache</code> object contains an entry for the key.
<code>boolean containsValue(Object value)</code>	Returns <code>true</code> if there is at least one entry with this value in the <code>NamedCache</code> object.
<code>Object get(Object key)</code>	Gets the entry from the <code>NamedCache</code> object for that key.

**Table 3–1 (Cont.) Methods in the NamedCache Interface**

Method Name	Description
Object put(Object key, Object value)	Puts an object in the cache and returns the previous value (if any).
Object remove(Object key)	Removes the mapping for this key from this map if present. Inherited from the ConcurrentMap interface.
Set entrySet()	Returns a set of key/value pairs.
Collection values()	Gets all values back as a collection.
CacheService getCacheService()	Returns the CacheService to which this NamedCache belongs.

The `com.tangosol.net.CacheFactory` class is typically used to get an instance of a `NamedCache` object. [Table 3–2](#) describes some of the more commonly used methods in the `CacheFactory` class.

**Table 3–2 Methods in the CacheFactory Class**

Method Name	Description
static Cluster ensureCluster()	Obtains a cluster object running Coherence services.
static void shutdown()	Shuts down all clustered services.
static NamedCache getCache(String cache)	Returns an instance of a cache. Either joins an existing cache in the cluster or creates the cache if this is the first member.

For a full list of methods in the `NamedCache` interface and the `CacheFactory` class, see the Javadoc in `C:\oracle\product\coherence\doc`.

## Creating Your First Coherence-Based Java Program

This section describes how to create a Java program that enables you to access, update, and remove simple types of information from a Coherence clustered cache.

1. [Create a Program to Put Values in the Cache](#)
2. [Create a Program to Get Values from the Cache](#)

### Create a Program to Put Values in the Cache

To create a Coherence Java-based application:

1. Create a new Application Client Project in Eclipse. Name the project `InsertValue`. Ensure that the folder is `C:\home\oracle\workspace\InsertValue`.

In the **Configuration** section of the **New Application Client Project** dialog box, click **Modify**. In the **Project Facets** dialog box, select **CoherenceConfig** from the **Configuration** drop down list.

See "[Creating a New Project in the Eclipse IDE](#)" on page 2-3 for detailed instructions.

2. Create your first Coherence Java program. Name the class `MyFirstSample` and select the **public static void main(String[] args)** check box in the **New Java Class** dialog box.

See ["Creating a Java Class"](#) on page 2-11 for detailed information.

3. In the Eclipse editor, write the code to create a `NamedCache` object, enter a value in the cache, and then verify the value that you entered. [Example 3-1](#) illustrates a sample program.

**Example 3-1 Creating a NamedCache Object: Inserting and Verifying Values**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

public class MyFirstSample {
    public MyFirstSample() {
    }

    public static void main(String[] args) {

        // create or get a named cache called mycache
        NamedCache myCache = CacheFactory.getCache("mycache");

        // put key, value pair into the cache.
        myCache.put("Name", "Gene Smith");

        System.out.println("Value in cache is " + myCache.get("Name"));
    }
}
```

4. Stop any running cache servers. See ["Stopping Cache Servers"](#) on page 2-14 for detailed information.
5. Run the program in the Eclipse IDE: right-click the `MyFirstSample.java` class in the editor and select **Run As** then **Run Configuration**. Double-click Oracle Coherence to create a Coherence configuration. Enter `MyFirstSample` in the **Name** field of the **Run Configuration** dialog box.

In the **Main** tab, enter `InsertValue` in the **Project** field and `com.oracle.handson.MyFirstSample` in the **Main class** field.

In the **Coherence** tab, enter a unique value, such as 3155, in the **Cluster port** field to ensure that Coherence is limited to your own host. Select **Enabled (cache server)** under **Local storage**.

Note that in the **Classpath** tab, the `coherence.jar` file should be present in the **Bootstrap Entries** list. Click **Apply**, then **Run**.

Messages similar to [Example 3-2](#) are displayed:

**Example 3-2 Output of MyFirstSample Program**

```
2013-11-14 14:46:52.652/0.343 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence.xml"
2013-11-14 14:46:52.746/0.437 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence-override-dev.xml"
2013-11-14 14:46:52.855/0.546 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from
"file:/C:/home/oracle/workspace/InsertValue/build/classes/tangosol-coherence-override.xml"
```

```
2013-11-14 14:46:52.855/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-config.xml" is not specified
2013-11-14 14:46:52.855/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-builder-config.xml" is not specified
2013-11-14 14:46:52.855/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "/custom-mbeans.xml" is not specified
```

Oracle Coherence Version 12.1.3.0.0 Build 48392

Grid Edition: Development mode

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

```
2013-11-14 14:46:53.511/1.202 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/coherence-cache-config.xml"
2013-11-14 14:46:53.652/1.343 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_
Home/coherence/lib/coherence.jar!/internal-txn-cache-config.xml"
2013-11-14 14:46:54.369/2.060 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-11-14 14:46:55.336/3.027 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP
bound to /10.159.162.203:8088 using SystemDatagramSocketProvider
2013-11-14 14:46:59.237/6.928 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
Created a new cluster "cluster:0x47DB" with Member(Id=1, Timestamp=2013-11-14 14:46:55.665,
Address=10.159.162.203:8088, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:8184,
Role=OracleHandsonMyFirstSample, Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4)
2013-11-14 14:46:59.237/6.928 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Started cluster Name=cluster:0x47DB
```

```
Group{Address=224.12.1.0, Port=3155, TTL=4}
```

```
MasterMemberSet (
```

```
  ThisMember=Member(Id=1, Timestamp=2013-11-14 14:46:55.665, Address=10.159.162.203:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:8184, Role=OracleHandsonMyFirstSample)
  OldestMember=Member(Id=1, Timestamp=2013-11-14 14:46:55.665, Address=10.159.162.203:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:8184, Role=OracleHandsonMyFirstSample)
  ActualMemberSet=MemberSet(Size=1
    Member(Id=1, Timestamp=2013-11-14 14:46:55.665, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8184, Role=OracleHandsonMyFirstSample)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
    1|12.1.3|2013-11-14 14:46:55.665|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=0
  )
)
```

```
TcpRing{Connections=[]}
```

```
IpMonitor{Addresses=0}
```

```
2013-11-14 14:46:59.270/6.961 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Service Management joined the cluster with senior service member 1
2013-11-14 14:46:59.301/6.992 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-14 14:46:59.784/7.475 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=1): TcpAcceptor now listening for connections on 10.159.
162.203:8088.3
2013-11-14 14:47:00.159/7.850 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): Service DistributedCache joined the cluster with senior service member 1
```

```
2013-11-14 14:47:00.190/7.881 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): This member has become the distribution coordinator for MemberSet(Size=1
  Member(Id=1, Timestamp=2013-11-14 14:46:55.665, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8184, Role=OracleHandsonMyFirstSample)
)
```

**Value in cache is Gene Smith**

## Create a Program to Get Values from the Cache

To create a Java class that gets the value from your cache, instead of using a put and then a get method:

1. Create another Java class with a main method named `MyFirstSampleReader`. See ["Creating a Java Class"](#) on page 2-11 for detailed information. [Example 3-3](#) illustrates a sample program.

### Example 3-3 Getting a Value from the Cache

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

public class MyFirstSampleReader {

    public MyFirstSampleReader() {
    }

    public static void main(String[] args) {
        // ensure we are in a cluster
        CacheFactory.ensureCluster();

        // create or get a named cache called mycache
        NamedCache myCache = CacheFactory.getCache("mycache");

        System.out.println("Value in cache is " + myCache.get("Name"));
    }
}
```

2. Run the `MyFirstSampleReader` class in the Eclipse IDE: right-click the `MyFirstSampleReader.java` class in the editor and select **Run As** then **Run Configuration**.

Enter `MyFirstSampleReader` in the **Name** field of the **Run Configuration** dialog box. Ensure that `InsertValue` appears in the **Project** field in the **Main** tab and `com.oracle.handson.MyFirstSampleReader` appears in the **Main class** field. In the **Coherence** tab, ensure that local storage is enabled and **Cluster port** is set to 3155 to limit Coherence to your own host. Click **Apply**, then **Run**.

[Example 3-4](#) illustrates the output from the program. Note that a null value is returned. Although the `MyFirstSample` program successfully created and populated the `NamedCache` cache, it only existed within the `MyFirstSample` process memory. When the `MyFirstSample` program terminated so did the cache.

### Example 3-4 Output of the MyFirstSampleReader Program

```
2013-11-14 14:55:44.990/0.327 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence.xml"
```

```
2013-11-14 14:55:45.099/0.436 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence-override-dev.xml"
2013-11-14 14:55:45.193/0.530 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from
"file:/C:/home/oracle/workspace/InsertValue/build/classes/tangosol-coherence-override.xml"
2013-11-14 14:55:45.209/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-config.xml" is not specified
2013-11-14 14:55:45.209/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-builder-config.xml" is not specified
2013-11-14 14:55:45.209/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "/custom-mbeans.xml" is not specified
```

```
Oracle Coherence Version 12.1.3.0.0 Build 48392
Grid Edition: Development mode
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.
```

```
2013-11-14 14:55:46.754/2.091 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP
bound to /10.159.162.203:8088 using SystemDatagramSocketProvider
2013-11-14 14:55:50.966/6.303 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
Created a new cluster "cluster:0x47DB" with Member(Id=1, Timestamp=2013-11-14 14:55:47.097,
Address=10.159.162.203:8088, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7304,
Role=OracleHandsonMyFirstSampleReader, Edition=Grid Edition, Mode=Development, CpuCount=4,
SocketCount=4)
2013-11-14 14:55:50.966/6.303 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Started cluster Name=cluster:0x47DB
```

```
Group{Address=224.12.1.0, Port=3155, TTL=4}
```

```
MasterMemberSet (
  ThisMember=Member(Id=1, Timestamp=2013-11-14 14:55:47.097, Address=10.159.162.203:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7304,
Role=OracleHandsonMyFirstSampleReader)
  OldestMember=Member(Id=1, Timestamp=2013-11-14 14:55:47.097, Address=10.159.162.203:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7304,
Role=OracleHandsonMyFirstSampleReader)
  ActualMemberSet=MemberSet(Size=1
    Member(Id=1, Timestamp=2013-11-14 14:55:47.097, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:7304, Role=OracleHandsonMyFirstSampleReader)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
  1|12.1.3|2013-11-14 14:55:47.097|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=0
  )
)
```

```
TcpRing{Connections=[]}
IpMonitor{Addresses=0}
```

```
2013-11-14 14:55:51.014/6.351 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Service Management joined the cluster with senior service member 1
2013-11-14 14:55:51.030/6.367 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-14 14:55:51.482/6.819 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=1): TcpAcceptor now listening for connections on 10.159.
162.203:8088.3
```

```

2013-11-14 14:55:51.841/7.178 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/coherence-cache-config.xml"
2013-11-14 14:55:51.966/7.303 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/internal-txn-cache-config.xml"
2013-11-14 14:55:52.168/7.505 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-11-14 14:55:52.232/7.569 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): Service DistributedCache joined the cluster with senior service member 1
2013-11-14 14:55:52.263/7.600 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): This member has become the distribution coordinator for MemberSet(Size=1
Member(Id=1, Timestamp=2013-11-14 14:55:47.097, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:7304, Role=OracleHandsonMyFirstSampleReader)
)

```

**Value in cache is null**

3. Start the `DefaultCacheServer` that you created in "[Launching a Cache Server in the Eclipse IDE](#)" on page 2-6. Ensure that `InsertValue` appears in the **Project** field in the **Main** tab and that local storage is enabled in the **Coherence** tab. Click **Apply**, then **Run**.
4. Run `MyFirstSample` to put the value in the `NamedCache`, and then run `MyFirstSampleReader` to read the value from the cache. Note the output illustrated in [Example 3-5](#). The Gene Smith value stored by `MyFirstSample` is returned by `MyFirstSampleReader`.

### **Example 3-5 Output of MyFirstSampleReader Program with a Running Cache Server**

```

2013-11-14 15:07:24.940/0.374 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence.xml"
2013-11-14 15:07:25.034/0.468 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence-override-dev.xml"
2013-11-14 15:07:25.112/0.546 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from
"file:/C:/home/oracle/workspace/InsertValue/build/classes/tangosol-coherence-override.xml"
2013-11-14 15:07:25.112/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-config.xml" is not specified
2013-11-14 15:07:25.112/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-builder-config.xml" is not specified
2013-11-14 15:07:25.112/0.546 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "/custom-mbeans.xml" is not specified

```

```

Oracle Coherence Version 12.1.3.0.0 Build 48392
Grid Edition: Development mode
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

```

```

2013-11-14 15:07:26.922/2.356 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP
bound to /10.159.162.203:8090 using SystemDatagramSocketProvider
2013-11-14 15:07:28.343/3.777 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=n/a):
Member(Id=1, Timestamp=2013-11-14 15:06:21.971, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:6980, Role=CoherenceServer) joined Cluster with senior
member 1
2013-11-14 15:07:28.343/3.777 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
This Member(Id=3, Timestamp=2013-11-14 15:07:28.156, Address=10.159.162.203:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8096, Role=OracleHandsonMyFirstSampleReader,
Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4) joined cluster "cluster:0x47DB"
with senior Member(Id=1, Timestamp=2013-11-14 15:06:21.971, Address=10.159.162.203:8088,

```

```

MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:6980, Role=CoherenceServer,
Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4)
2013-11-14 15:07:28.422/3.856 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Started cluster Name=cluster:0x47DB

Group{Address=224.12.1.0, Port=3155, TTL=4}

MasterMemberSet (
  ThisMember=Member(Id=3, Timestamp=2013-11-14 15:07:28.156, Address=10.159.162.203:8090,
MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:8096,
Role=OracleHandsonMyFirstSampleReader)
  OldestMember=Member(Id=1, Timestamp=2013-11-14 15:06:21.971, Address=10.159.162.203:8088,
MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:6980, Role=CoherenceServer)
  ActualMemberSet=MemberSet(Size=2
    Member(Id=1, Timestamp=2013-11-14 15:06:21.971, Address=10.159.162.203:8088, MachineId=47251,
Location=site:,machine:TPFAEFL-LAP,process:6980, Role=CoherenceServer)
    Member(Id=3, Timestamp=2013-11-14 15:07:28.156, Address=10.159.162.203:8090, MachineId=47251,
Location=site:,machine:TPFAEFL-LAP,process:8096, Role=OracleHandsonMyFirstSampleReader)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
  1|12.1.3|2013-11-14 15:06:21.971|JOINED,
  3|12.1.3|2013-11-14 15:07:28.156|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=1
    Member(Id=2, Timestamp=2013-11-14 15:06:53.752, Address=10.159.162.203:8090, MachineId=47251)
  )
)

TcpRing{Connections=[1]}
IpMonitor{Addresses=0}

2013-11-14 15:07:28.453/3.887 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=3): Service Management joined the cluster with senior service member 1
2013-11-14 15:07:28.469/3.903 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-14 15:07:28.906/4.340 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=3): TcpAcceptor now listening for connections on 10.159.
162.203:8090.3
2013-11-14 15:07:29.280/4.714 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/coherence-cache-config.xml"
2013-11-14 15:07:29.389/4.823 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/internal-txn-cache-config.xml"
2013-11-14 15:07:29.593/5.027 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-11-14 15:07:29.671/5.105 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=3): Service DistributedCache joined the cluster with senior service member 1
Value in cache is Gene Smith
2013-11-14 15:07:29.702/5.136 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=ShutdownHook, member=3):
ShutdownHook: stopping cluster node

```

This should not be the case for a process that joins the cluster only to perform an operation, such as entering a value and then terminating, like `MyFirstSample`. By default, all processes start as storage-enabled. The process can store data as part of the cluster. Modify the process so that it is not storage-enabled.

- a. Right click the `MyFirstSample` class, select **Run As** then **Run Configurations**. See ["Creating a Runtime Configuration"](#) on page 2-12 for detailed information.
  - b. In the Coherence tab, select **Disabled (cache client)** under **Local storage**.  
This similar to setting the Java parameter `-Dtangosol.coherence.distributed.localstorage=false`.
5. Shut down any running cache servers and rerun your `MyFirstSample` class.  
You will receive a message similar to the one in [Example 3-6](#) indicating that storage is not enabled on the cluster, because you have set this member to be storage-disabled.

### Example 3-6 Output of the `MyFirstSample` Class with Cache Storage Disabled

```

...
TcpRing{Connections=[]}
IpMonitor{Addresses=0}

2013-11-14 15:11:40.253/7.039 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Service Management joined the cluster with senior service member 1
2013-11-14 15:11:40.284/7.070 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-14 15:11:40.721/7.507 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=1): TcpAcceptor now listening for connections on 10.159.
162.203:8088.3
2013-11-14 15:11:41.080/7.866 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): Service DistributedCache joined the cluster with senior service member 1
Exception in thread "main" com.tangosol.net.RequestPolicyException: No storage-enabled nodes exist
for service DistributedCache
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
partitionedService.PartitionedCache$BinaryMap.onMissingStorage(PartitionedCache.CDB:27)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
partitionedService.PartitionedCache$BinaryMap.ensureRequestTarget(PartitionedCache.CDB:43)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
partitionedService.PartitionedCache$BinaryMap.put(PartitionedCache.CDB:23)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
partitionedService.PartitionedCache$BinaryMap.put(PartitionedCache.CDB:1)
    at com.oracle.common.collections.ConverterCollections$ConverterMap.
put(ConverterCollections.java:1531)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
partitionedService.PartitionedCache$ViewMap.put(PartitionedCache.CDB:1)
    at com.tangosol.coherence.component.util.SafeNamedCache.put(SafeNamedCache.CDB:1)
    at com.oracle.handson.MyFirstSample.main(MyFirstSample.java:16)

```

6. Restart the `DefaultCacheServer` cache server and run `MyFirstSample` and `MyFirstSampleReader` again. You should now see that the data is persisted between running the two Java examples.

## Creating Your First Coherence-Based Java Application

In this exercise, you develop a simple Java console-based application to access, update, and remove simple types of information from a Coherence clustered cache.

To perform this exercise, you must complete ["Testing a Coherence Installation"](#) on page 1-2.

Unlike client/server applications, in which client applications typically connect and disconnect from a server application, Coherence-based clustered applications simply ensure they are in a cluster, after which they can use the services of the cluster. Coherence-based applications typically do not connect to a cluster of applications; they become part of the cluster.

1. [Create the Console Application](#)
2. [Run the Console Application](#)

## Create the Console Application

To create a Java console-based application to access, update, and remove simple types of information from a Coherence clustered cache:

1. Examine the methods in the `CacheFactory` class using the Coherence Java documentation (Javadoc) that is shipped in the `... \coherence \doc` folder.
2. Write a simple Java console application (Java class) called `YourFirstCoherenceApplication` that uses the `CacheFactory` class to join a cluster (using the `ensureCluster` method), and then leave the cluster (using the `shutdown` method). See "[Creating a Java Class](#)" on page 2-11 for detailed information on creating a Java class.
  - a. Examine the methods that are available in the `NamedCache` interface using the Javadoc.
  - b. Extend your application to use the `CacheFactory` method `getCache` to acquire a `NamedCache` for the cache called `mycache` (the same cache name used in the exercise "[Testing a Coherence Installation](#)" on page 1-2).
  - c. With the `NamedCache` instance, use the `get` method to retrieve the value for the key `message` (the same key used in the exercise "[Testing a Coherence Installation](#)" on page 1-2).
  - d. Write the value to standard output using the `System.out.println(...)` method.

[Example 3-7](#) illustrates a sample Coherence-based Java application:

### **Example 3-7 A Coherence-Based Java Application**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

public class YourFirstCoherenceApplication {
    public YourFirstCoherenceApplication() {
    }

    public static void main(String[] args) {

        CacheFactory.ensureCluster();

        NamedCache myCache = CacheFactory.getCache("mycache");

        String message = (String)myCache.get("message");

        System.out.println(message);

        CacheFactory.shutdown();
    }
}
```

```

YourFirstCoherenceApplication yourfirstcoherenceapplication = new
YourFirstCoherenceApplication();
    }
}

```

## Run the Console Application

To run the Coherence application.

1. Start the `DefaultCacheServer` cache server that you created in ["Launching a Cache Server in the Eclipse IDE"](#) on page 2-6.
2. Create a run configuration for a cache client that uses Coherence Query Language and the `QueryHelper` API. Right click the project and select **Run As** then **Run Configurations**.

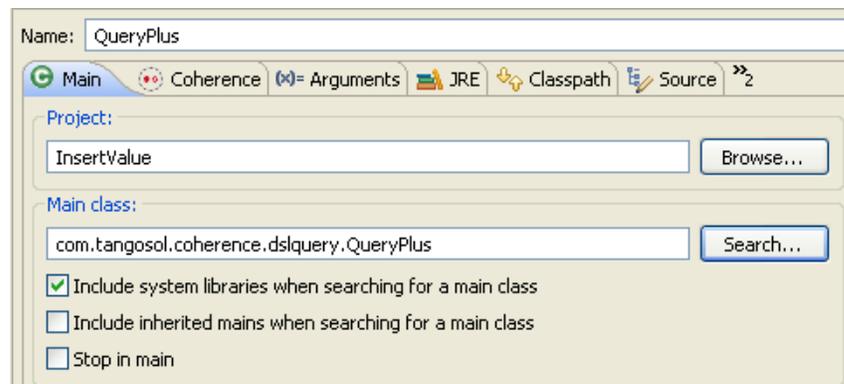
---

**Note:** For more information about Coherence Query Language, see ["Using Coherence Query Language"](#) in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

---

- a. In the **Run Configurations** dialog box click **Oracle Coherence** then the **New launch configuration** icon. Enter `QueryPlus` as the **Name** of the configuration.
- b. In the **Main** tab, under **Project** click the **Browse** button and select the **InsertValue** project. Under **Main class**, select **Include system libraries when searching for a main class** and click the **Search** button. In the Select Main Type dialog box, enter `QueryPlus` and select **QueryPlus - com.tangosol.coherence.dslquery**. Click **OK**. The **Main** tab should look similar to [Figure 3-1](#).

**Figure 3-1 Main Tab for the Query Client Configuration**



- c. In the **Coherence** tab, select **Disabled (cache client)** under **Local storage**. Enter a unique value for the **Cluster port** (the value must be the same as the value defined for the cache server you defined in the previous section).
- d. In the **Arguments** tab, enter `-showversion` in the **VM arguments** field.
- e. In the **Common** tab, select **Shared file** and click the **Browse** button to navigate to the `\InsertValue` project name. Ensure that the **Allocate console** checkbox is selected. Click **Apply**.

- Click **Run** to start the QueryPlus client. You should see output similar to [Example 3-8](#) in the Eclipse Console.

**Example 3-8 Output for the QueryPlus Cache Client**

```
java version "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b17)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)

Coherence Command Line Tool
jline library cannot be loaded, so you cannot use the arrow keys for line editing
and history.

CohQL>
```

- At the CohQL> prompt, enter the following command to create a cache named mycache and to connect to it.

```
CohQL> create cache "mycache"
```

- Create a run configuration for YourFirstCoherenceApplication. In the **Run Configurations** dialog box, enter YourFirstCoherenceApplication in the **Name** field. In the **Main** tab, enter InsertValue in the **Project** field and com.oracle.handson>YourFirstCoherenceApplication as the **Main** class.

In the Coherence tab, select **Enabled (cache server)** under **Local storage**, and enter a unique value for the **Cluster port** (this must be the same value that you used for the cache server and cache client).

- Execute YourFirstCoherenceApplication from the Eclipse IDE and view the result.

[Example 3-9](#) illustrates the output from YourFirstCoherenceApplication. The output indicates that there is no data in the cache for the message key.

**Example 3-9 Output of the YourFirstCoherenceApplication Class**

```
...
TcpRing{Connections=[2]}
IpMonitor{Addresses=0}

2013-11-14 15:41:33.008/3.793 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=3): Service Management joined the cluster with senior service member 1
2013-11-14 15:41:33.039/3.824 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-14 15:41:33.570/4.355 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=3): TcpAcceptor now listening for connections on 10.159.
162.203:8092.3
2013-11-14 15:41:33.975/4.760 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/coherence-cache-config.xml"
2013-11-14 15:41:34.132/4.917 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/internal-txn-cache-config.xml"
2013-11-14 15:41:34.475/5.260 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-11-14 15:41:34.538/5.323 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=3): Service DistributedCache joined the cluster with senior service member 1
null
```

```

2013-11-14 15:41:44.601/15.386 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=n/a): Service DistributedCache left the cluster
2013-11-14 15:41:44.601/15.386 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=n/a): Service Management left the cluster
2013-11-14 15:41:44.651/15.436 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=n/a):
Service Cluster left the cluster

```

7. Using the running QueryPlus cache client in the Eclipse Console, change the key message. For example, enter the following at the CohQL> prompt:

```
CohQL> insert into "mycache" key "message" value "hello"
```

Rerun `YourFirstCoherenceApplication` from the Eclipse IDE to see the changed values. [Example 3–10](#) illustrates that the cache now holds the value `hello` for the key message.

### Example 3–10 Output of the `YourFirstCoherenceApplication` Class with a New Key Value

```

...
TcpRing{Connections=[2]}
IpMonitor{Addresses=0}

2013-11-14 15:45:01.006/3.986 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=4): Service Management joined the cluster with senior service member 1
2013-11-14 15:45:01.037/4.017 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=4): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-14 15:45:01.583/4.563 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=4): TcpAcceptor now listening for connections on 10.159.
162.203:8092.3
2013-11-14 15:45:01.973/4.953 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=4): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/coherence-cache-config.xml"
2013-11-14 15:45:02.083/5.063 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=4): Loaded
cache configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/internal-txn-cache-config.xml"
2013-11-14 15:45:02.286/5.266 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=4):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-11-14 15:45:02.364/5.344 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=4): Service DistributedCache joined the cluster with senior service member 1
hello
2013-11-14 15:45:12.397/15.377 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=n/a): Service DistributedCache left the cluster
2013-11-14 15:45:12.398/15.378 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=n/a): Service Management left the cluster
2013-11-14 15:45:12.447/15.427 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=n/a):
Service Cluster left the cluster

```

8. In the run configuration for `YourFirstCoherenceApplication`, change the value of the **Local storage** from **Enabled** to **Disabled**. Notice that the output is the same as the previous run.
9. Shut down your cache server and cache client instances. Restart the cache server and then rerun `YourFirstCoherenceApplication` (with the new value for **Local storage**). Note the output is now null.
10. If you change the value of the message key in your application (using the `put` method), verify that the new value available through the cache client.
  - a. For example, comment out the `get` method and add the `put` method.

```
//String message = (String)myCache.get("message");  
String message = (String)myCache.put("message", "bye");
```

- b.** Run `YourFirstCoherenceApplication`.
- c.** Run the `get` command in the `QueryPlus` cache client.

```
select key(), value() from "mycache" where key() is "message"
```

The output `bye` is displayed.

---

---

## Working with Complex Objects

In this exercise, you work with complex objects located in the cache. It highlights the use of the Coherence `PofReader`, `PofWriter`, and `PortableObject` API. Using Eclipse, you create a new `Contact` class, and then store and retrieve `Contact` objects in the cache using Portable Object Format (POF) serialization.

This chapter contains the following sections:

- [Introduction](#)
- [Creating and Caching Complex Objects](#)

### Introduction

Until now, you have been putting and getting `String` objects as the value in a `NamedCache` cache. Many of the implementations of the `get` and `put` methods in the Coherence Java API define the values and keys to be of type `Object`, for example:

```
public java.lang.Object get(java.lang.Object oKey)
public void put(java.lang.Object oKey, java.lang.Object oValue)
```

Any object can be used as a value or key. This enables you to store complex objects as values in the cache.

Because Coherence might send the object across the wire, the object must be serializable. Object serialization is the process of saving an object's state into a sequence of bytes, and then rebuilding (deserializing) the bytes into an active object at a future time. For example, objects that implement the `java.io.Serializable` interface are serializable.

As an alternative to using the Java `java.io.Serializable` interface, you can improve performance by using Coherence's own class for high-performance serialization, `com.tangosol.io.pof.PortableObject`. `PortableObject` format is up to six times faster than the standard `Serializable` and the serialized result set is smaller.

The `PortableObject` interface provides two simple methods, `readExternal` and `writeExternal`, that permit you to explicitly read and write serialized object attributes from the provided `PofReader` and `PofWriter` streams respectively. By taking control over the serialization format, Coherence provides a way to improve the performance of the process. Using POF reduces the size of the resulting binary file. The size of the binary file is often 5 to 10 times smaller, and the conversion to or from the binary file can be between 5 and 20 times faster, depending on the size of the object.

## Creating and Caching Complex Objects

In this exercise, you create a `Contact` object that contains names, addresses, dates of birth, and telephone numbers for employees. You also use POF serialization to put the objects in the cache and retrieve them by implementing the `PortableObject` interface.

1. [Create the Data Objects](#)
2. [Create the Complex Object](#)
3. [Create the Driver Class](#)
4. [Create the POF and Cache Configuration Files](#)
5. [Run the Sample Project](#)

### Create the Data Objects

This section describes how to create two data objects that will later be incorporated into another data object. An `Address` object will provide employee address information and a `PhoneNumber` object will provide telephone contact information.

1. Create an `Address` object to store address information for an employee.
  - a. Create a new Application Client Project in Eclipse called `Contacts`. Ensure that the `CoherenceConfig` is selected in the **Configuration** field on the opening page and the **Create a default main** is *not* selected on the Application Client module page.

See ["Creating a New Project in the Eclipse IDE"](#) on page 2-3 for detailed information.
  - b. Create a new Java class called `Address`. Ensure that the **Default Package** is `com.oracle.handson`. Do not select the **Main Method** check box. See ["Creating a Java Class"](#) on page 2-11 for detailed information.
  - c. Write the class to use the `PortableObject` interface for data serialization. In the Eclipse code editor, change your generated `Address` class to implement `com.tangosol.io.pof.PortableObject`. Add an `import` statement for the `PortableObject` interface.
  - d. Import the `com.tangosol.io.pof.PofReader`, `com.tangosol.io.pof.PofWriter` and `java.io.IOException` classes required by the `PortableObject` interface.
  - e. Add the default public constructor for `Address` that is required by the `PortableObject` interface.
  - f. Enter the following private attributes for your `Address` class. You can add others if you like.

```
— String Street1
— String Street2
— String City
— String State
— String Country
```

At this point, the `Address` class should look similar to the following:

```
package com.oracle.handson;
```

```
import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PortableObject;
import com.tangosol.io.pof.PofWriter;

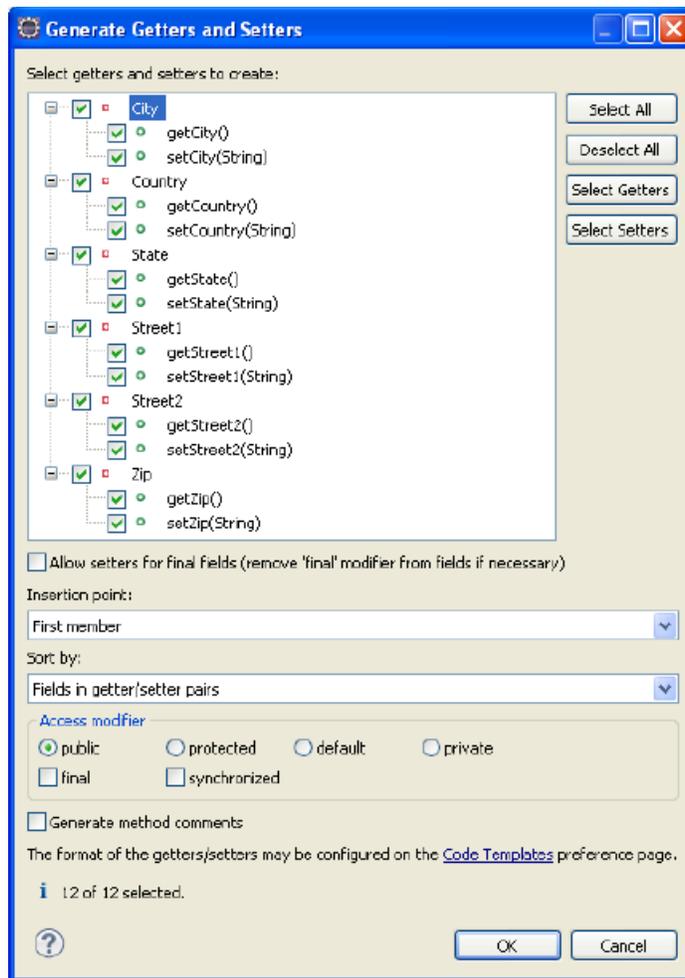
import java.io.IOException;

public class Address implements PortableObject
{
    private String Street1;
    private String Street2;
    private String City;
    private String State;
    private String Zip;
    private String Country;
    /**
     * Default constructor (necessary for PortableObject implementation).
     */
    public Address()
    {
    }
}
```

- g.** Eclipse can generate the default `get` and `set` methods for your attributes. From the **Source** menu, select **Generate Getters and Setters**. Click **Select All** to select all of the attributes in the class. All of the attributes are now automatically selected. Click **OK** to continue.

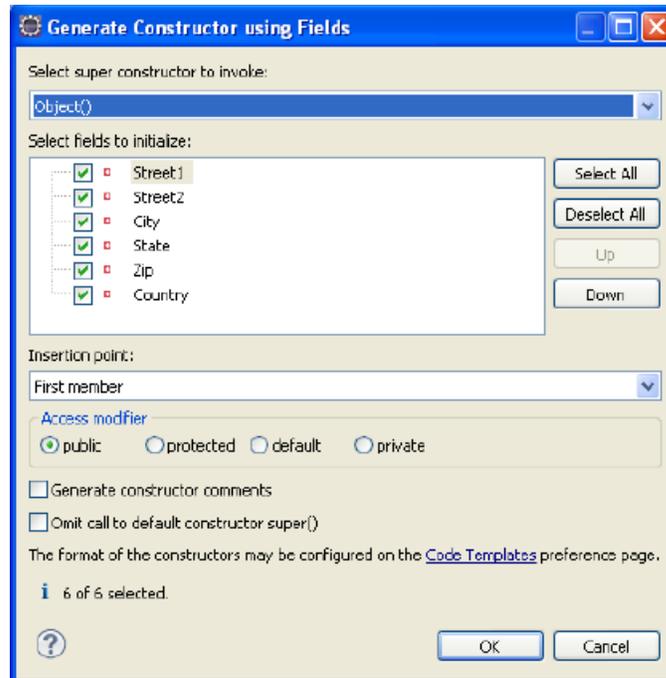
[Figure 4–1](#) illustrates the **Generate Getters and Setters** dialog box with the generated accessors for the `Address` class.

**Figure 4–1 Generate Getters and Setters Dialog Box**



- h. You can also generate the default constructor and equals methods automatically. From the **Source** menu, select **Generate Constructor using Fields**, click **Select All**, and then click **OK**.

Figure 4–2 illustrates the **Generate Constructor using Fields** dialog box with the **Street1**, **Street2**, **City**, **State**, **Zip**, and **Country** fields selected.

**Figure 4–2** Generate Constructors using Fields Dialog Box

Add "this" to the members of the generated constructor. The generated constructor should then look similar to the following:

```
public Address(String Street1, String Street2, String City, String
State, String Zip, String Country) {
    super();
    this.Street1 = Street1;
    this.Street2 = Street2;
    this.City = City;
    this.State = State;
    this.Zip = Zip;
    this.Country = Country;
}
```

- i. Implement the `readExternal` and `writeExternal` methods as required by the `PortableObject` interface. For example, the following implementation of the `readExternal` method enables the values for the street, city, state, and country to be read as POF objects.

```
public void readExternal(PofReader reader)
    throws IOException
{
    setStreet1(reader.readString(0));
    setStreet2(reader.readString(1));
    setCity(reader.readString(2));
    setState(reader.readString(3));
    setZip(reader.readString(4));
    setCountry(reader.readString(5));
}
```

- j. Implement the `equals`, `hashCode`, and `toString` object methods.

---

---

**Note:** Cache keys and values must be serializable (for example, `java.io.Serializable`). Cache keys must also provide an implementation of the `hashCode()` and `equals()` methods, and those methods must return consistent results across cluster nodes. This implies that the implementation of `hashCode()` and `equals()` must be based solely on the object's serializable state (that is, the object's nontransient fields); most built-in Java types, such as `String`, `Integer` and `Date`, meet this requirement. Some cache implementations (specifically the partitioned cache) use the serialized form of the key objects for equality testing, which means that keys for which the `equals()` method returns `true` must serialize identically; most built-in Java types meet this requirement.

---

---

To support these methods, import the `com.tangosol.util.Base` and `com.tangosol.util.HashHelper` classes. The `Base` class provides support for the `equals` method. The `HashHelper` class contains helper functions for calculating hash code values for any group of Java intrinsics.

The following code illustrates a sample implementation of the `equals()` method:

```
public boolean equals(Object oThat)
{
    if (this == oThat)
    {
        return true;
    }
    if (oThat == null)
    {
        return false;
    }

    Address that = (Address) oThat;
    return Base.equals(getStreet1(), that.getStreet1()) &&
        Base.equals(getStreet2(), that.getStreet2()) &&
        Base.equals(getCity(), that.getCity()) &&
        Base.equals(getState(), that.getState()) &&
        Base.equals(getZip(), that.getZip()) &&
        Base.equals(getCountry(), that.getCountry());
}
```

The following code illustrates a sample implementation of the `hashCode()` method:

```
public int hashCode()
{
    return HashHelper.hash(getStreet1(),
        HashHelper.hash(getStreet2(),
            HashHelper.hash(getZip(), 0)));
}
```

The following code illustrates a sample implementation of the `toString()` method:

```
public String toString()
{
    return getStreet1() + "\n" +
        getStreet2() + "\n" +
```

```

        getCity() + ", " + getState() + " " + getZip() + "\n" +
        getCountry();
    }

```

k. The resulting class should look similar to [Example 4-1](#).

#### **Example 4-1 Implementation of an Address Class**

```

package com.oracle.handson;

import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PortableObject;
import com.tangosol.io.pof.PofWriter;

import com.tangosol.util.Base;
import com.tangosol.util.HashHelper;

import java.io.IOException;

public class Address implements PortableObject
{
    private String Street1;
    private String Street2;
    private String City;
    private String State;
    private String Zip;
    private String Country;

    /**
     * Default constructor (necessary for PortableObject implementation).
     */
    public Address() {
    }

    public Address(String Street1, String Street2, String City, String State,
        String Zip, String Country)
    {
        super();
        this.Street1 = Street1;
        this.Street2 = Street2;
        this.City    = City;
        this.State   = State;
        this.Zip     = Zip;
        this.Country = Country;
    }

    //----- accessors-----

    public void setStreet1(String Street1)
    {
        this.Street1 = Street1;
    }

    public String getStreet1()
    {
        return Street1;
    }

    public void setStreet2(String Street2)
    {
        this.Street2 = Street2;
    }

```

```

    }

    public String getStreet2()
    {
        return Street2;
    }

    public void setCity(String City)
    {
        this.City = City;
    }

    public String getCity()
    {
        return City;
    }

    public void setState(String State)
    {
        this.State = State;
    }

    public String getState()
    {
        return State;
    }

    public void setZip(String Zip)
    {
        this.Zip = Zip;
    }

    public String getZip()
    {
        return Zip;
    }

    public void setCountry(String Country)
    {
        this.Country = Country;
    }

    public String getCountry()
    {
        return Country;
    }
// ----- PortableObject Interface-----

    public void readExternal(PofReader reader)
        throws IOException
    {
        setStreet1(reader.readString(0));
        setStreet2(reader.readString(1));
        setCity(reader.readString(2));
        setState(reader.readString(3));
        setZip(reader.readString(4));
        setCountry(reader.readString(5));
    }

    public void writeExternal(PofWriter writer)

```

```

        throws IOException
    {
        writer.writeString(0, getStreet1());
        writer.writeString(1, getStreet2());
        writer.writeString(2, getCity());
        writer.writeString(3, getState());
        writer.writeString(4, getZip());
        writer.writeString(5, getCountry());
    }
// ----- Object methods -----

public boolean equals(Object oThat)
{
    if (this == oThat)
    {
        return true;
    }
    if (oThat == null)
    {
        return false;
    }

    Address that = (Address) oThat;
    return Base.equals(getStreet1(), that.getStreet1()) &&
        Base.equals(getStreet2(), that.getStreet2()) &&
        Base.equals(getCity(), that.getCity()) &&
        Base.equals(getState(), that.getState()) &&
        Base.equals(getZip(), that.getZip()) &&
        Base.equals(getCountry(), that.getCountry());
}

public int hashCode()
{
    return HashHelper.hash(getStreet1(),
        HashHelper.hash(getStreet2(),
            HashHelper.hash(getZip(), 0)));
}

public String toString()
{
    return getStreet1() + "\n" +
        getStreet2() + "\n" +
        getCity() + ", " + getState() + " " + getZip() + "\n" +
        getCountry();
}
}

```

**2.** Create a `PhoneNumber` class to store telephone contact data.

- a.** Create a new Java class called `PhoneNumber`. Do not include a `main` method. See "[Creating a Java Class](#)" on page 2-11 for detailed information.
- b.** Use the `PortableObject` interface for data serialization. In the Eclipse code editor, change your generated `PhoneNumber` class to implement `PortableObject`. Add an `import` statement for the `com.tangosol.io.pof.PortableObject` interface.

- c. Import the `com.tangosol.io.pof.PofReader`, `com.tangosol.io.pof.PofWriter`, and `java.io.IOException` classes required by the `PortableObject` interface.
- d. Add the default public constructor for the `PhoneNumber` class that is required by the `PortableObject` interface.
- e. Enter the following private attributes for your `PhoneNumber` class. You can add others.

```
—short AccessCode
—short CountryCode
—short AreaCode
—int LocalNumber
```

- f. Eclipse can generate the default `get` and `set` methods for your attributes. From the **Source** menu, select **Generate Getters and Setters**. Click **Select All** to select all of the attributes in the class. All of the attributes are now automatically selected. Click **OK** to continue.
- g. You can also generate the default constructor and `equals` methods automatically. From the **Source** menu, select **Generate Constructor using Fields**, click **Select All**, and then click **OK**.

Add "this" to the members of the generated constructor. The generated constructor then looks similar to the following:

```
public PhoneNumber(short AccessCode, short CountryCode, short AreaCode,
    int LocalNumber) {
    super();
    this.AccessCode = AccessCode;
    this.CountryCode = CountryCode;
    this.AreaCode = AreaCode;
    this.LocalNumber = LocalNumber;
}
```

- h. Implement the `readExternal` and `writeExternal` methods as required by the `PortableObject` interface.
- i. Implement the `equals`, `hashCode` and `toString` object methods.
- j. The resulting class looks similar to [Example 4-2](#).

#### **Example 4-2 Implementation of a PhoneNumber Class**

```
package com.oracle.handson;

import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PofWriter;
import com.tangosol.io.pof.PortableObject;

import com.tangosol.util.HashHelper;

import java.io.IOException;

public class PhoneNumber implements PortableObject
{

    private short AccessCode;
    private short CountryCode;
    private short AreaCode;
```

```
private int LocalNumber;

//----- constructors -----

/**
 * Default constructor (necessary for PortableObject implementation).
 */
public PhoneNumber() {
}

public PhoneNumber(short AccessCode, short CountryCode, short AreaCode,
                  int LocalNumber)
{
    super();
    this.AccessCode = AccessCode;
    this.CountryCode = CountryCode;
    this.AreaCode = AreaCode;
    this.LocalNumber = LocalNumber;
}

//----- accessors-----

public void setAccessCode(short AccessCode)
{
    this.AccessCode = AccessCode;
}

public short getAccessCode()
{
    return AccessCode;
}

public void setCountryCode(short CountryCode)
{
    this.CountryCode = CountryCode;
}

public short getCountryCode()
{
    return CountryCode;
}

public void setAreaCode(short AreaCode)
{
    this.AreaCode = AreaCode;
}

public short getAreaCode()
{
    return AreaCode;
}

public void setLocalNumber(int LocalNumber)
{
    this.LocalNumber = LocalNumber;
}

public int getLocalNumber()
{

```

```

        return LocalNumber;
    }
// ----- PortableObject Interface-----

public void readExternal(PofReader reader)
    throws IOException
    {
    setAccessCode(reader.readShort(0));
    setCountryCode(reader.readShort(1));
    setAreaCode(reader.readShort(2));
    setLocalNumber(reader.readInt(3));
    }

public void writeExternal(PofWriter writer)
    throws IOException
    {
    writer.writeShort(0, getAccessCode());
    writer.writeShort(1, getCountryCode());
    writer.writeShort(2, getAreaCode());
    writer.writeInt(3, getLocalNumber());
    }
// ----- Object methods -----

/**
 * {@inheritDoc}
 */
public boolean equals(Object oThat)
    {
    if (this == oThat)
        {
        return true;
        }
    if (oThat == null)
        {
        return false;
        }

    PhoneNumber that = (PhoneNumber) oThat;
    return getAccessCode() == that.getAccessCode() &&
        getCountryCode() == that.getCountryCode() &&
        getAreaCode() == that.getAreaCode() &&
        getLocalNumber() == that.getLocalNumber();
    }

/**
 * {@inheritDoc}
 */
public int hashCode()
    {
    return HashHelper.hash(getAreaCode(),
        HashHelper.hash(getLocalNumber(), 0));
    }

/**
 * {@inheritDoc}
 */
public String toString()
    {
    return "+" + getAccessCode() + " " + getCountryCode() + " "

```

```

        + getAreaCode() + " " + getLocalNumber();
    }
}

```

## Create the Complex Object

The `Contact` object provides the name, address, and telephone information of employees by incorporating the `Address` and `PhoneNumber` data objects.

1. Create a new Java class called `Contact`. Do not include a `main` method.  
See "[Creating a Java Class](#)" on page 2-11 for more information.
2. Because the class uses the `PortableObject` interface for data serialization, change your generated `Contact` class to implement `PortableObject` in the Eclipse code editor. Add an import statement for the `com.tangosol.io.pof.PortableObject` interface.
3. Import the `com.tangosol.io.pof.PofReader` and `com.tangosol.io.pof.PofWriter` and `java.io.IOException` classes required by `PortableObject`.
4. Add the default public constructor for `Contact` that is required by `PortableObject`.
5. Enter the following private attributes for your `Contact` class. You can add others.
  - `String FirstName`
  - `String LastName`
  - `Address HomeAddress`
  - `Address WorkAddress`
  - `Map TelephoneNumbers`
  - `java.sql.Date BirthDate`
6. Eclipse can generate the default `get` and `set` methods for your attributes. From the **Source** menu, select **Generate Getters and Setters**. Click **Select All** to select all of the attributes in the class. Click **OK** to continue.
7. Create an accessor, `getAge`, to calculate the age of an employee:

```

public int getAge()
{
    return (int) ((System.currentTimeMillis() - BirthDate.getTime()) /
MILLIS_IN_YEAR);
}

```

8. Add a definition for `MILLIS_IN_YEAR`.  

```
public static final long MILLIS_IN_YEAR = 1000L * 60L * 60L * 24L * 365L;
```
9. You can generate the default constructor automatically. From the **Source** menu, select **Generate Constructor using Fields**, click **Select All**, and then click **OK**.

The generated constructor looks similar to the following:

```

public Contact(String FirstName, String LastName, Address HomeAddress,
Address WorkAddress, Map TelephoneNumbers, Date BirthDate) {
    super();
    this.FirstName = FirstName;
    this.LastName = LastName;
    this.HomeAddress = HomeAddress;
    this.WorkAddress = WorkAddress;
}

```

```
        this.TelephoneNumbers = TelephoneNumbers;
        this.BirthDate = BirthDate;
    }
```

10. Implement the `readExternal` and `writeExternal` methods as required by the `PortableObject` interface.
11. Implement the `equals`, `hashCode`, and `toString` object methods.
12. The resulting class looks similar to [Example 4-3](#).

**Example 4-3 Sample Contact Class**

```
package com.oracle.handson;

import com.tangosol.io.pof.PortableObject;

import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PofWriter;

import java.io.IOException;

import java.sql.Date;

import java.util.Iterator;
import java.util.Map;

public class Contact implements PortableObject
{
    private String FirstName;
    private String LastName;
    private Address HomeAddress;
    private Address WorkAddress;
    private Map TelephoneNumbers;
    private java.sql.Date BirthDate;

    // ----- constructors -----

    /**
     * Default constructor (necessary for PortableObject implementation).
     */
    public Contact()
    {
    }

    public Contact(String FirstName, String LastName, Address HomeAddress,
                  Address WorkAddress, Map TelephoneNumbers, Date BirthDate)
    {
        super();
        this.FirstName = FirstName;
        this.LastName = LastName;
        this.HomeAddress = HomeAddress;
        this.WorkAddress = WorkAddress;
        this.TelephoneNumbers = TelephoneNumbers;
        this.BirthDate = BirthDate;
    }

    // ----- accessors -----

    public void setFirstName(String FirstName)
    {
```

```
        this.FirstName = FirstName;
    }

    public String getFirstName()
    {
        return FirstName;
    }

    public void setLastName(String LastName)
    {
        this.LastName = LastName;
    }

    public String getLastName()
    {
        return LastName;
    }

    public void setHomeAddress(Address HomeAddress)
    {
        this.HomeAddress = HomeAddress;
    }

    public Address getHomeAddress()
    {
        return HomeAddress;
    }

    public void setWorkAddress(Address WorkAddress)
    {
        this.WorkAddress = WorkAddress;
    }

    public Address getWorkAddress()
    {
        return WorkAddress;
    }

    public void setTelephoneNumbers(Map TelephoneNumbers)
    {
        this.TelephoneNumbers = TelephoneNumbers;
    }

    public Map getTelephoneNumbers()
    {
        return TelephoneNumbers;
    }

    public void setBirthDate(Date BirthDate)
    {
        this.BirthDate = BirthDate;
    }

    public Date getBirthDate()
    {
        return BirthDate;
    }
    /**
    * Get age.
    *

```

```

    * @return age
    */
    public int getAge()
    {
        return (int) ((System.currentTimeMillis() - BirthDate.getTime()) /
            MILLIS_IN_YEAR);
    }

    // ----- PortableObject interface -----

    /**
     * {@inheritDoc}
     */
    public void readExternal(PofReader reader)
        throws IOException
    {
        setFirstName(reader.readString(0));
        setLastName(reader.readString(1));
        setHomeAddress((Address) reader.readObject(2));
        setWorkAddress((Address) reader.readObject(3));
        setTelephoneNumbers(reader.readMap(4, null));
        setBirthDate(new Date(reader.readLong(5)));
    }

    /**
     * {@inheritDoc}
     */
    public void writeExternal(PofWriter writer)
        throws IOException
    {
        writer.writeString(0, getFirstName());
        writer.writeString(1, getLastName());
        writer.writeObject(2, getHomeAddress());
        writer.writeObject(3, getWorkAddress());
        writer.writeMap(4, getTelephoneNumbers());
        writer.writeLong(5, getBirthDate().getTime());
    }

    // ----- Object methods -----

    /**
     * {@inheritDoc}
     */
    public String toString()
    {
        StringBuffer sb = new StringBuffer(getFirstName())
            .append(" ")
            .append(getLastName())
            .append("\nAddresses")
            .append("\nHome: ").append(getHomeAddress())
            .append("\nWork: ").append(getWorkAddress())
            .append("\nTelephone Numbers");

        for (Iterator iter = TelephoneNumbers.entrySet().iterator();
            iter.hasNext(); )
        {
            Map.Entry entry = (Map.Entry) iter.next();
            sb.append("\n")

```

```

        .append(entry.getKey()).append(": ").append(entry.getValue());
    }
    return sb.append("\nBirth Date: ").append(getBirthDate()).toString();
}
/**
 * Approximate number of millis in a year ignoring things such as leap
 * years. Suitable for example use only.
 */
public static final long MILLIS_IN_YEAR = 1000L * 60L * 60L * 24L * 365L;
}

```

## Create the Driver Class

Create a driver class called `ContactDriver` to put `Contact` entries into the cache and retrieve them.

1. Create a new Java class called `ContactDriver` in the `Contacts` project. Ensure that it includes a `main` method.

See ["Creating a Java Class"](#) on page 2-11 for detailed information.

2. In the `ContactDriver` class, create a new `NamedCache` called `contact` and put a new instance of the `Contact` object in it. Get the `Contact` object from the cache and ensure that the two objects are identical. [Example 4-4](#) illustrates a sample implementation of this class.

### Example 4-4 Sample `ContactDriver` Class

```

package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import java.sql.Date;

import java.util.GregorianCalendar;
import java.util.Map;
import java.util.HashMap;

public class ContactDriver {

    public ContactDriver()
    {
    }

    public static void main(String[] args) {
        NamedCache contact = CacheFactory.getCache("contact");

        Address homeAddress = new Address ("4157 Wash Ave", "Suite 4",
                                           "Burlingame", "CA", "94407", "USA");
        Address workAddress = new Address ("500 Oracle Pkwy", "MS989",
                                           "Redwood Shores", "CA", "94065", "USA");
        Date date = new java.sql.Date(new GregorianCalendar(2011, 05, 01).
getTime().getTime());
        PhoneNumber phonenumber = new PhoneNumber ((short)11, (short)650,
(short)506, 7000);
        Map map = new HashMap();
        map.put("home", phonenumber);

        Contact con1 = new Contact("Tom", "Dunn", homeAddress, workAddress,

```

```

        map, date);

    contact.put(con1.getFirstName(), con1);

    Contact con2 = (Contact)contact.get(con1.getFirstName());

    if (con2.getFirstName().equals(con1.getFirstName()))
    {
        System.out.println("They are the same!!");
    }
}
}

```

## Create the POF and Cache Configuration Files

To use POF serialization, you must register your user-defined objects in a POF configuration file. The configuration associates the class of a user-defined object with a numeric value. You must also specify POF serialization and the name of the POF configuration file in the cache configuration file.

1. Create a POF configuration file for the `Contact`, `Address`, and `PhoneNumber` objects.
  - a. To create a POF configuration file for your data types, use the `pof-config.xml` file that was created with your project.
  - b. Define `<user-type>` elements for the `Contact`, `Address`, and `PhoneNumber` objects, assign type IDs 1001, 1002, and 1003 to them and provide their full class names. The file must include the `coherence-pof-config.xml` file which reserves the first 1000 IDs for Coherence data types.
  - c. Rename the file and save it as `contacts-pof-config.xml` (it will be saved to the `C:\home\oracle\workspace\Contacts\appClientModule` folder). [Example 4-5](#) illustrates a sample `contacts-pof-config.xml` file.

### Example 4-5 POF Configuration File

```

<?xml version="1.0"?>
<pof-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://xmlns.oracle.com/coherence/coherence-pof-config"
  xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-pof-config
http://xmlns.oracle.com/coherence/coherence-pof-config/1.2/coherence-pof-config.
xsd">

  <user-type-list>

    <!-- coherence POF user types -->
    <include>coherence-pof-config.xml</include>

    <!-- com.tangosol.examples package -->
    <user-type>
      <type-id>1001</type-id>
      <class-name>com.oracle.handson.Contact</class-name>
    </user-type>
    <user-type>
      <type-id>1002</type-id>
      <class-name>com.oracle.handson.Address</class-name>
    </user-type>
    <user-type>
      <type-id>1003</type-id>
      <class-name>com.oracle.handson.PhoneNumber</class-name>

```

```

    </user-type>
  </user-type-list>
  <allow-interfaces>true</allow-interfaces>
  <allow-subclasses>true</allow-subclasses>
</pof-config>

```

2. Create a cache configuration file. You can use the `coherence-cache-config.xml` file in the **Project Explorer**, which is based on the `coherence-cache-config.xsd` file. You can also find a copy of `coherence-cache-config.xml` file in the `coherence.jar` file.
  - a. Use `ExamplesPartitionedPofScheme` as the `scheme-name` and `PartitionedPofCache` as the `service-name`.
  - b. The `serializer` section is responsible for mapping a POF serialized object to an appropriate serialization routine. You will declare the location of the POF configuration file in the next section.
  - c. Rename the file in the **Project Explorer** and save it as `contacts-cache-config.xml` (it will be saved to the `C:\home\oracle\workspace\Contacts\appClientModule` folder). [Example 4-6](#) illustrates a sample `contacts-cache-config.xml` file.

#### Example 4-6 Cache Configuration File

```

<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
              xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-
cache-config coherence-cache-config.xsd">
  <defaults>
    <serializer>pof</serializer>
  </defaults>

  <caching-scheme-mapping>
    <cache-mapping>
      <cache-name>*</cache-name>
      <scheme-name>ExamplesPartitionedPofScheme</scheme-name>
    </cache-mapping>
  </caching-scheme-mapping>

  <caching-schemes>
    <distributed-scheme>
      <scheme-name>ExamplesPartitionedPofScheme</scheme-name>
      <service-name>PartitionedPofCache</service-name>
      <thread-count>5</thread-count>
      <backing-map-scheme>
        <local-scheme>
          <!-- each node will be limited to 32MB -->
          <high-units>250MB</high-units>
          <unit-calculator>binary</unit-calculator>
        </local-scheme>
      </backing-map-scheme>
      <autostart>true</autostart>
    </distributed-scheme>
  </caching-schemes>
</cache-config>

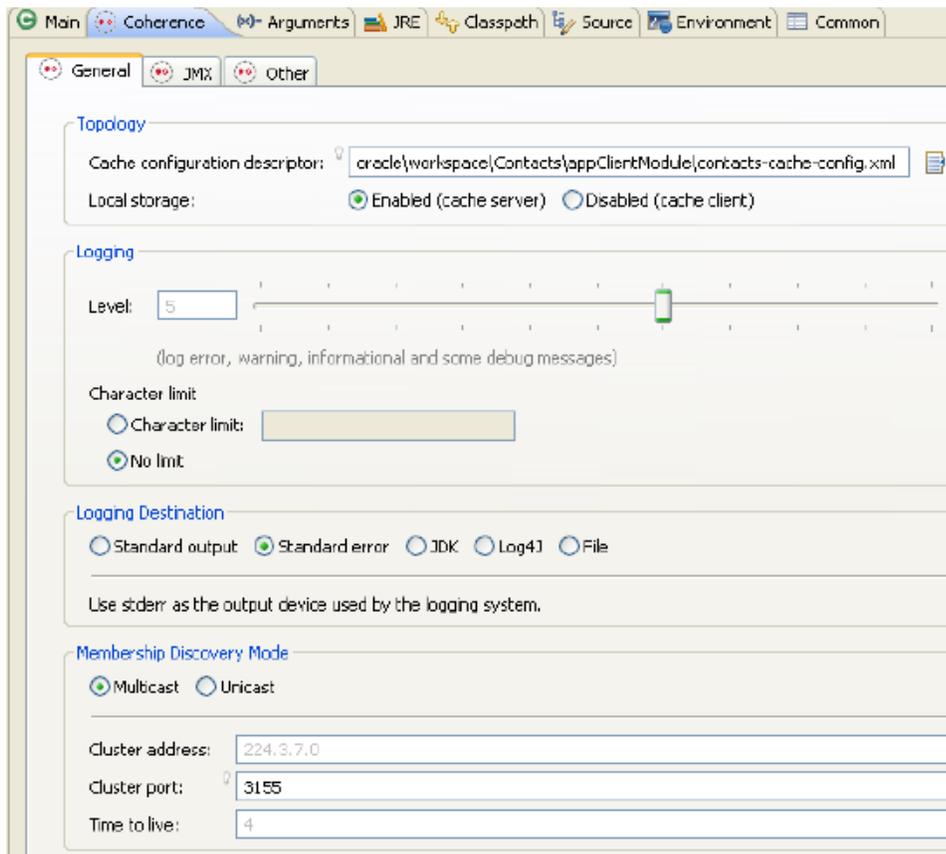
```

## Run the Sample Project

The contacts project requires a cache server to be running. The cache server and the application must reference the same POF and cache configuration files. This section describes how to create run configurations for the cache server and the application.

1. Stop any cache servers that are running. See "Stopping Cache Servers" on page 2-14 for detailed information.
2. Create an executable file to start the cache server.
  - a. Right click the project and select **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, enter `ContactsCacheServer` in the **Name** field.
  - b. In the **Main** tab, ensure that **Contacts** appears in the **Project** field. Select the **Include system libraries when searching for a main class** checkbox and click the **Search** button. In the **Select Main Type** dialog box, enter `DefaultCacheServer` and select the **DefaultCacheServer - com.tangosol.net** class. Click **OK** in the **Select Main Type** dialog box, then **Apply** in the **Run Configurations** dialog box.
  - c. In the **General** tab of the **Coherence** tab, select **Absolute File Path** in the **Topology** field and browse to the `contacts-cache-config.xml` file in the `C:\home\oracle\workspace\Contacts\appClientModule` directory. Ensure that **Enabled (cache server)** is selected. Enter a unique value for the **Cluster port**. Click **Apply**. The **General** tab should look similar to [Figure 4-3](#).

**Figure 4-3** Coherence Tab for the Contacts Cache Server Executable



- d. In the **Other** tab of the **Coherence** tab, scroll down to the `tangosol.pof.config` item. Enter the path to the POF configuration file: `C:\home\oracle\workspace\Contacts\appClientModule\contacts-pof-config.xml`.
- e. In the **Arguments** tab, enter `-showversion` under **VM Arguments**.
- f. In the **Classpath** tab, ensure that the **Contacts** project appears under **User Entries**. If it does not, click the **Add Projects** button to add the **Contacts** project.
- g. In the **Common** tab, select **Shared file** and browse for the `\Contacts` project.
- h. Click **Run** to start the `Contacts` cache server. The output should be similar to [Example 4-7](#).

#### **Example 4-7 Output from the Contacts Cache Server**

```
java version "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b17)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)

2013-11-21 11:04:21.818/0.312 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence.xml"
2013-11-21 11:04:21.912/0.406 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/tangosol-coherence-override-dev.xml"
2013-11-21 11:04:22.006/0.500 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "file:/C:/home/oracle/workspace/Contacts/build/classes/tangosol-
coherence-override.xml"
2013-11-21 11:04:22.006/0.500 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-config.xml" is not specified
2013-11-21 11:04:22.006/0.500 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-builder-config.xml" is not specified
2013-11-21 11:04:22.006/0.500 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "/custom-mbeans.xml" is not specified

Oracle Coherence Version 12.1.3.0.0 Build 48392
  Grid Edition: Development mode
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

2013-11-21 11:04:22.631/1.125 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "file:/C:/home/oracle/workspace/Contacts/appClientModule/contacts-
cache-config.xml"
2013-11-21 11:04:23.691/2.185 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-11-21 11:04:24.457/2.951 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP
bound to /130.35.99.28:8088 using SystemDatagramSocketProvider
2013-11-21 11:04:28.294/6.788 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
Created a new cluster "cluster:0x47DB" with Member(Id=1, Timestamp=2013-11-21 11:04:24.769,
Address=130.35.99.28:8088, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:1344,
Role=CoherenceServer, Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4)
2013-11-21 11:04:28.294/6.788 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Started cluster Name=cluster:0x47DB

Group{Address=224.12.1.0, Port=3155, TTL=4}

MasterMemberSet (
  ThisMember=Member(Id=1, Timestamp=2013-11-21 11:04:24.769, Address=130.35.99.28:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:1344, Role=CoherenceServer)
```

```

    OldestMember=Member(Id=1, Timestamp=2013-11-21 11:04:24.769, Address=130.35.99.28:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:1344, Role=CoherenceServer)
    ActualMemberSet=MemberSet(Size=1
        Member(Id=1, Timestamp=2013-11-21 11:04:24.769, Address=130.35.99.28:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:1344, Role=CoherenceServer)
    )
    MemberId|ServiceVersion|ServiceJoined|MemberState
    1|12.1.3|2013-11-21 11:04:24.769|JOINED
    RecycleMillis=1200000
    RecycleSet=MemberSet(Size=0
    )
)

```

```

TcpRing{Connections=[]}
IpMonitor{Addresses=0}

```

```

2013-11-21 11:04:28.295/6.789 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Service Management joined the cluster with senior service member 1
2013-11-21 11:04:28.328/6.822 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-21 11:04:28.718/7.212 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=1): TcpAcceptor now listening for connections on 130.35.99.
28:8088.3
2013-11-21 11:04:29.061/7.555 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=DistributedCache:PartitionedPofCache, member=1): Loaded POF configuration from
"file:/C:/home/oracle/workspace/Contacts/appClientModule/contacts-pof-config.xml"
2013-11-21 11:04:29.076/7.570 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=DistributedCache:PartitionedPofCache, member=1): Loaded included POF configuration from
"jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/coherence-pof-config.xml"
2013-11-21 11:04:29.123/7.617 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): Service PartitionedPofCache joined the
cluster with senior service member 1
2013-11-21 11:04:29.139/7.633 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=1):
Services
(
    ClusterService{Name=Cluster, State=(SERVICE_STARTED, STATE_JOINED), Id=0, Version=12.1.3,
OldestMemberId=1}
    InvocationService{Name=Management, State=(SERVICE_STARTED), Id=2, Version=12.1.3,
OldestMemberId=1}
    PartitionedCache{Name=PartitionedPofCache, State=(SERVICE_STARTED), LocalStorage=enabled,
PartitionCount=257, BackupCount=1, AssignedPartitions=0, BackupPartitions=0}
)

```

Started DefaultCacheServer...

### 3. Create a run configuration for the ContactDriver executable.

- Right click the ContactDriver.java file in the **Project Explorer** and select **Run As** then **Run Configurations**.
- In the **Run Configurations** dialog box, double-click the Oracle Coherence node. Enter **ContactsDriver** in the **Name** field. In the **Main** tab, browse for **Contacts** in the **Project** field, and **com.oracle.handson.ContactDriver** in the **Main class** field. Click **Apply**.
- In the **General** tab of the **Coherence** tab, browse for the absolute path to the **contacts-cache-config.xml** file in the **Cache configuration descriptor** field. Select the **Disable (cache client)** radio button. In the **Cluster port** field, enter **3155**

- In the **Other** tab of the **Coherence** tab, scroll down to the **tangosol.pof.config** field and replace the value with the absolute path to the `contacts-pof-config.xml` POF configuration file.
  - In the **Classpath** tab, ensure that **Contacts (default classpath)** appears under **User Entries**.
4. Click **Run** to run the `ContactDriver` configuration. The output of the `Contacts` example should look similar to [Example 4-8](#).

In this example, the `Contact` object is converted to POF at run time. Using POF provides significant performance improvements in CPU time and the size of the generated binary file.

#### **Example 4-8 Output of the Contacts Example in the Eclipse IDE**

```
2013-11-21 11:26:44.085/0.312 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded operational configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/tangosol-coherence.xml"
```

```
2013-11-21 11:26:44.163/0.390 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded operational overrides from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/tangosol-coherence-override-dev.xml"
```

```
2013-11-21 11:26:44.256/0.483 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded operational overrides from "file:/C:/home/oracle/workspace/Contacts/build/classes/tangosol-coherence-override.xml"
```

```
2013-11-21 11:26:44.256/0.483 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional configuration override "cache-factory-config.xml" is not specified
```

```
2013-11-21 11:26:44.256/0.483 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional configuration override "cache-factory-builder-config.xml" is not specified
```

```
2013-11-21 11:26:44.256/0.483 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional configuration override "/custom-mbeans.xml" is not specified
```

```
Oracle Coherence Version 12.1.3.0.0 Build 48392
```

```
Grid Edition: Development mode
```

```
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.
```

```
2013-11-21 11:26:44.897/1.124 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
```

```
Loaded cache configuration from "file:/C:/home/oracle/workspace/Contacts/appClientModule/contacts-cache-config.xml"
```

```
2013-11-21 11:26:45.942/2.169 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
```

```
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
```

```
2013-11-21 11:26:47.004/3.231 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP bound to /130.35.99.28:8090 using SystemDatagramSocketProvider
```

```
2013-11-21 11:26:47.816/4.043 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=n/a):
```

```
Member(Id=1, Timestamp=2013-11-21 11:26:21.557, Address=130.35.99.28:8088, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7096, Role=CoherenceServer) joined Cluster with senior member 1
```

```
2013-11-21 11:26:47.817/4.044 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
```

```
This Member(Id=2, Timestamp=2013-11-21 11:26:47.66, Address=130.35.99.28:8090, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:6432, Role=OracleHandsonContactDriver, Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4) joined cluster "cluster:0x47DB" with senior
```

```
Member(Id=1, Timestamp=2013-11-21 11:26:21.557, Address=130.35.99.28:8088, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7096, Role=CoherenceServer, Edition=Grid Edition, Mode=Development, CpuCount=4, SocketCount=4)
```

```
2013-11-21 11:26:48.067/4.294 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
```

```
Started cluster Name=cluster:0x47DB
```

```
Group{Address=224.12.1.0, Port=3155, TTL=4}
```

```
MasterMemberSet (
```

```

ThisMember=Member(Id=2, Timestamp=2013-11-21 11:26:47.66, Address=130.35.99.28:8090,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:6432, Role=OracleHandsonContactDriver)
  OldestMember=Member(Id=1, Timestamp=2013-11-21 11:26:21.557, Address=130.35.99.28:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7096, Role=CoherenceServer)
  ActualMemberSet=MemberSet(Size=2
    Member(Id=1, Timestamp=2013-11-21 11:26:21.557, Address=130.35.99.28:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:7096, Role=CoherenceServer)
    Member(Id=2, Timestamp=2013-11-21 11:26:47.66, Address=130.35.99.28:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:6432, Role=OracleHandsonContactDriver)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
    1|12.1.3|2013-11-21 11:26:21.557|JOINED,
    2|12.1.3|2013-11-21 11:26:47.66|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=0
)
)

TcpRing{Connections=[1]}
IpMonitor{Addresses=0}

```

```

2013-11-21 11:26:48.068/4.295 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=2): Service Management joined the cluster with senior service member 1
2013-11-21 11:26:48.099/4.326 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=2): Loaded
Reporter configuration from "jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.
jar!/reports/report-group.xml"
2013-11-21 11:26:48.505/4.732 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=2): TcpAcceptor now listening for connections on 130.35.99.
28:8090.3
2013-11-21 11:26:48.848/5.075 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=DistributedCache:PartitionedPofCache, member=2): Loaded POF configuration from
"file:/C:/home/oracle/workspace/Contacts/appClientModule/contacts-pof-config.xml"
2013-11-21 11:26:48.879/5.106 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=DistributedCache:PartitionedPofCache, member=2): Loaded included POF configuration from
"jar:file:/C:/Oracle/Middleware/Oracle_Home/coherence/lib/coherence.jar!/coherence-pof-config.xml"
2013-11-21 11:26:48.910/5.137 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=2): Service PartitionedPofCache joined the
cluster with senior service member 1
They are the same!!

```

5. If you go back and look at the cache server output, you can see messages reporting the Contacts cache client joining the cluster, completing its work, then leaving the cluster. See [Example 4-9](#).

**Example 4-9 Contacts Cache Server Displaying the Arrival and Departure of the Contacts Client**

```

...
Started DefaultCacheServer...

2013-11-21 11:26:25.942/7.726 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): This member has become the distribution
coordinator for MemberSet(Size=1
  Member(Id=1, Timestamp=2013-11-21 11:26:21.557, Address=130.35.99.28:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:7096, Role=CoherenceServer)
)
2013-11-21 11:26:47.816/29.600 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2013-11-21 11:26:47.66, Address=130.35.99.28:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:6432, Role=OracleHandsonContactDriver) joined Cluster
with senior member 1

```

```
2013-11-21 11:26:48.099/29.883 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 joined Service Management with senior member 1
2013-11-21 11:26:48.941/30.725 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): Member 2 joined Service
PartitionedPofCache with senior member 1
2013-11-21 11:26:49.114/30.898 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
TcpRing disconnected from Member(Id=2, Timestamp=2013-11-21 11:26:47.66, Address=130.35.99.28:8090,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:6432, Role=OracleHandsonContactDriver)
due to a peer departure; removing the member.
2013-11-21 11:26:49.114/30.898 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2013-11-21 11:26:49.114, Address=130.35.99.28:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:6432, Role=OracleHandsonContactDriver) left Cluster
with senior member 1
2013-11-21 11:26:49.114/30.898 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 left service Management with senior member 1
2013-11-21 11:26:49.114/30.898 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): Member 2 left service PartitionedPofCache
with senior member 1
```



---

---

## Loading Data Into a Cache

In this exercise, you learn how to populate a Coherence cache with domain objects that are read from text files. It highlights the use of APIs from the Coherence `aggregator`, `extractor`, and `filter` packages.

This chapter contains the following sections:

- [Introduction](#)
- [Populating a Cache with Domain Objects](#)
- [Querying and Aggregating Data in the Cache](#)

### Introduction

Until now, you put objects into the cache and retrieved them individually. Each call to the `put` method can result in increased network traffic, especially for partitioned and replicated caches. Additionally, each call to a `put` method returns the object it just replaced in the cache, which adds more unnecessary overhead. By using the `putAll` method, loading the cache can be made much more efficient.

To perform the tasks in this chapter, you must first complete the project described in [Chapter 4, "Working with Complex Objects."](#) You must also be familiar with using `java.io.BufferedReader` to read text files, `java.lang.String.split` method to parse text files, and `java.text.SimpleDateFormat` to parse dates.

### Populating a Cache with Domain Objects

This exercise shows you how to create a console application that populates a Coherence cache with domain objects. The application will use the Coherence `com.tangosol.io.pof.PortableObject` implementation to serialize the objects into Portable Object format (POF).

In the exercise, you create a key that helps to get the `Contact` object, a generator to provide data for the cache, and a loader to load the cache.

1. [Create a Class with the Key for the Domain Objects](#)
2. [Edit the POF Configuration File](#)
3. [Create the Data Generator](#)
4. [Create a Console Application to Load the Cache](#)
5. [Run the Cache Loading Example](#)

## Create a Class with the Key for the Domain Objects

To create a class that contains the key for a domain object:

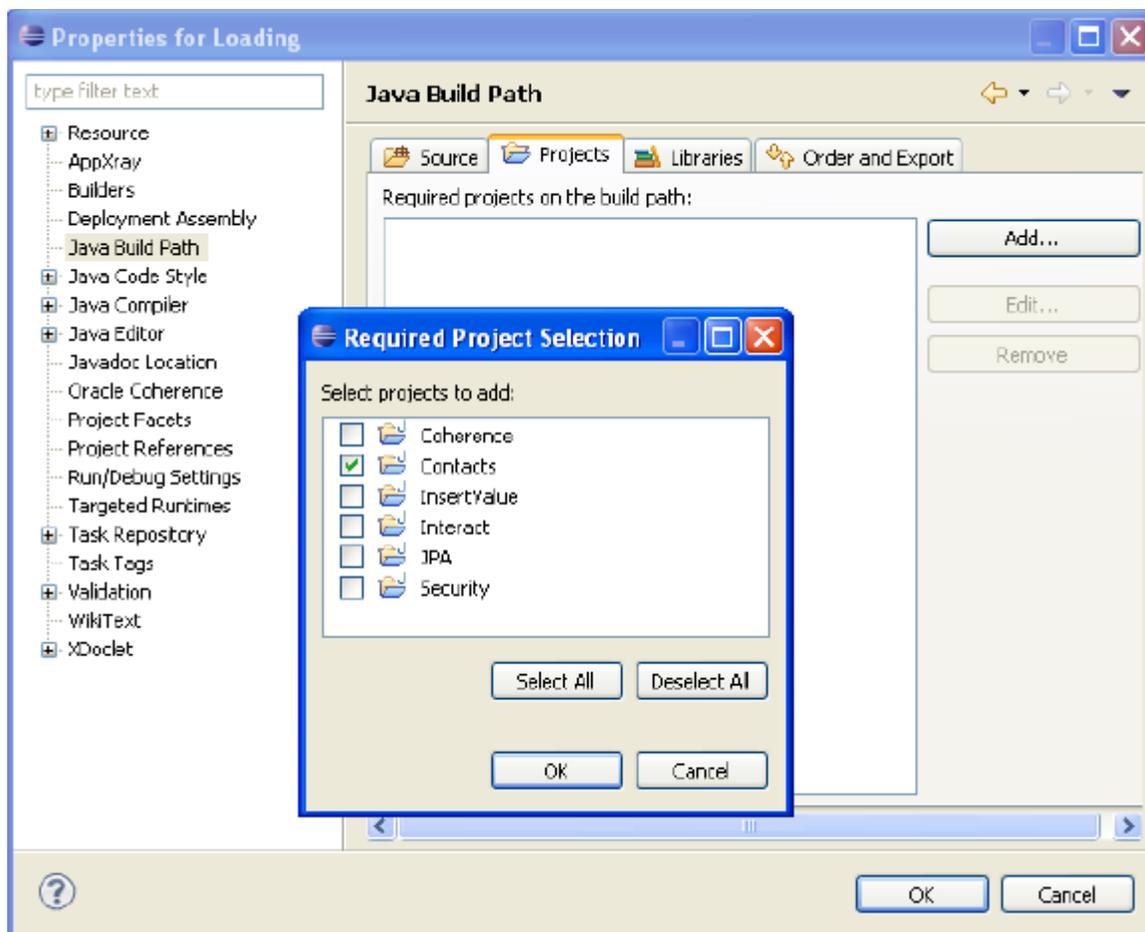
1. Create a new Application Client Project called **Loading**. Select **CoherenceConfig** from the **Configuration** drop-down list. In the Application Client Module page of the New Application Client Project wizard, deselect the Create a default Main class checkbox.

See "[Creating and Caching Complex Objects](#)" on page 4-2 for information on creating a new project.

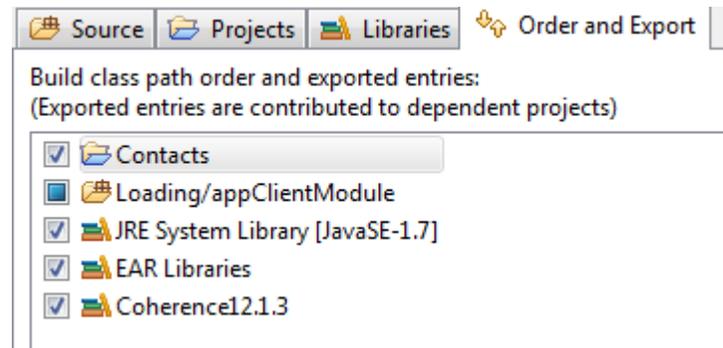
2. Add the classes and files related to the **Address**, **PhoneNumber**, and **Contact** classes that you created in an earlier exercise (**Contacts**). These files can be found in `c:\home\oracle\workspace\Contacts\appClientModule` and `c:\home\oracle\workspace\Contacts\build\classes` directories.

Right click the **Loading** project in the **Project Explorer** and select **Properties**. In the **Properties for Loading** dialog box, select **Java Build Path**. In the **Projects** tab, click **Add**. Select the **Contacts** project from the **Required Project Selection** dialog box, as illustrated in [Figure 5-1](#).

**Figure 5-1** Adding Folders to the Project Build Path



In the **Order and Export** tab, use the **Up** and **Down** buttons to move **Contacts** to the top of the list. The contents of the tab should look similar to [Figure 5-2](#).

**Figure 5–2 Contents of the Order and Export Tab for the Loading Project**

3. Create a `ContactId` class that provides a key to the employee for whom information is tracked. See "Creating a Java Class" on page 2-11 for detailed information on creating a Java class.

Create the contact ID based on the employee's first name and last name. This object acts as the key to get the `Contact` object.

Because this class uses POF serialization, it must implement the `PortableObject` interface, the `writeExternal` and `readExternal` `PortableObject` methods, and the `equals`, `hashCode`, and `toString` object methods.

---

**Note:** Cache keys and values must be serializable (for example, `java.io.Serializable`). Cache keys must also provide an implementation of the `hashCode` and `equals` methods, and those methods must return consistent results across cluster nodes. This implies that the implementation of the `hashCode` and `equals` object methods must be based solely on the object's serializable state (that is, the object's non-transient fields). Most built-in Java types, such as `String`, `Integer` and `Date`, meet this requirement. Some cache implementations (specifically the partitioned cache) use the serialized form of the key objects for equality testing, which means that keys for which the `equals` method returns true must serialize identically; most built-in Java types meet this requirement.

---

[Example 5–1](#) illustrates a possible implementation of the `ContactId` class.

#### **Example 5–1 Simple Contact ID Class**

```
package com.oracle.handson;

import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PofWriter;
import com.tangosol.io.pof.PortableObject;

import com.tangosol.util.Base;
import com.tangosol.util.HashHelper;

import java.io.IOException;

/**
 * ContactId is a key to the person for whom information is
 * tracked.
 */
```

```

*/
public class ContactId implements PortableObject
{
    // ----- constructors -----

    /**
     * Default constructor (necessary for PortableObject implementation).
     */
    public ContactId()
    {
    }

    /**
     * Construct a contact person.
     */
    public ContactId(String FirstName, String LastName)
    {
        super();
        this.FirstName = FirstName;
        this.LastName = LastName;
    }

    // ----- accessors -----

    /**
     * Return the first name.
     */
    public String getFirstName()
    {
        return FirstName;
    }

    /**
     * Return the last name.
     */
    public String getLastName()
    {
        return LastName;
    }

    // ----- PortableObject interface -----

    public void readExternal(PofReader reader)
        throws IOException
    {
        FirstName = reader.readString(0);
        LastName = reader.readString(1);
    }

    public void writeExternal(PofWriter writer)
        throws IOException
    {
        writer.writeString(0, FirstName);
        writer.writeString(1, LastName);
    }

    // ----- Object methods -----

```

```

public boolean equals(Object oThat)
{
    if (this == oThat)
    {
        return true;
    }
    if (oThat == null)
    {
        return false;
    }

    ContactId that = (ContactId) oThat;
    return Base.equals(getFirstName(), that.getFirstName()) &&
        Base.equals(getLastName(), that.getLastName());
}

public int hashCode()
{
    return HashHelper.hash(getFirstName(),
        HashHelper.hash(getLastName(), 0));
}

public String toString()
{
    return getFirstName() + " " + getLastName();
}

// ----- data members -----

/**
 * First name.
 */
private String FirstName;

/**
 * Last name.
 */
private String LastName;
}

```

## Edit the POF Configuration File

Edit the POF configuration file. Add a `<user-type>` entry for the `ContactId` class to the `contacts-pof-config.xml` file. The file looks similar to [Example 5-2](#).

### **Example 5-2 POF Configuration File with the ContactId Entry**

```

<?xml version="1.0"?>

<pof-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://xmlns.oracle.com/coherence/coherence-pof-config"
    xsi:schemaLocation="http://xmlns.oracle.
com/coherence/coherence-pof-config coherence-pof-config.xsd">
    <user-type-list>

        <!-- coherence POF user types -->
        <include>coherence-pof-config.xml</include>

        <!-- com.tangosol.examples package -->

```

```
<user-type>
  <type-id>1001</type-id>
  <class-name>com.oracle.handson.Contact</class-name>
</user-type>
<user-type>
  <type-id>1002</type-id>
  <class-name>com.oracle.handson.Address</class-name>
</user-type>
<user-type>
  <type-id>1003</type-id>
  <class-name>com.oracle.handson.PhoneNumber</class-name>
</user-type>
<user-type>
  <type-id>1004</type-id>
  <class-name>com.oracle.handson.ContactId</class-name>
</user-type>
</user-type-list>
<allow-interfaces>true</allow-interfaces>
<allow-subclasses>true</allow-subclasses>
</pof-config>
```

## Create the Data Generator

Create a Java class named `DataGenerator` to generate random employee contact names and addresses. See ["Creating a Java Class"](#) on page 2-11 for detailed information.

Use the `Address`, `PhoneNumber`, and `Contact` classes that you created in an earlier exercise. Use `java.util.Random` to generate some random names, addresses, telephone numbers, and ages.

[Example 5-3](#) illustrates a possible implementation of the data generator. This implementation creates a text file, `contacts.csv`, that contains the employee contact information.

### **Example 5-3 Sample Data Generation Class**

```
package com.oracle.handson;

import com.oracle.handson.Address;
import com.oracle.handson.Contact;
import com.oracle.handson.PhoneNumber;
import com.tangosol.util.Base;

import java.io.BufferedWriter;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;

import java.sql.Date;

import java.util.Collections;
import java.util.Random;

/**
 * DataGenerator is a generator of sample contacts.
 */
```

```

public class DataGenerator
{
    // ----- static methods -----

    /**
     * Generate contacts.
     */
    public static void main(String[] asArg)
        throws IOException
    {
        String      sFile = asArg.length > 0 ? asArg[0] : FILENAME;
        int         cCon  = asArg.length > 1 ? Integer.parseInt(asArg[1]) : 1000;
        OutputStream out   = new FileOutputStream(sFile);

        generate(out, cCon);
        out.close();
    }

    /**
     * Generate the contacts and write them to a file.
     */
    public static void generate(OutputStream out, int cContacts)
        throws IOException
    {
        PrintWriter writer = new PrintWriter(new BufferedWriter(
            new OutputStreamWriter(out)));

        for (int i = 0; i < cContacts; ++i)
        {
            StringBuffer sb = new StringBuffer(256);

            //contact person
            sb.append("John, ")
              .append(getRandomName())
              .append(',');

            // home and work addresses
            sb.append(Integer.toString(Base.getRandom().nextInt(999)))
              .append(" Beacon St., ") /*street1, empty street2*/
              .append(getRandomName()) /*random city name*/
              .append(',')
              .append(getRandomState())
              .append(',')
              .append(getRandomZip())
              .append(",US,Yoyodyne Propulsion Systems,")
              .append("330 Lectroid Rd.,Grover's Mill,")
              .append(getRandomState())
              .append(',')
              .append(getRandomZip())
              .append(",US,");

            // home and work telephone numbers
            sb.append("home, ")
              .append(Base.toDelimitedString(getRandomPhoneDigits(), ", "))
              .append(",work, ")
              .append(Base.toDelimitedString(getRandomPhoneDigits(), ", "))
              .append(',');

            // random birth date in millis before or after the epoch
            sb.append(getRandomDateInMillis());
        }
    }
}

```

```
        writer.println(sb);
    }
    writer.flush();
}

/**
 * Return a random name.
 *
 */
private static String getRandomName()
{
    Random rand = Base.getRandom();
    int    cCh = 4 + rand.nextInt(7);
    char[] ach = new char[cCh];

    ach[0] = (char) ('A' + rand.nextInt(26));
    for (int of = 1; of < cCh; ++of)
    {
        ach[of] = (char) ('a' + rand.nextInt(26));
    }
    return new String(ach);
}

/**
 * Return a random phone muber.
 * The phone number includes access, country, area code, and local
 * number.
 *
 */
private static int[] getRandomPhoneDigits()
{
    {
        Random rand = Base.getRandom();
        return new int[]
        {
            11,                // access code
            rand.nextInt(99),  // country code
            rand.nextInt(999), // area code
            rand.nextInt(9999999) // local number
        };
    }
}

/**
 * Return a random Phone.
 *
 */
private static PhoneNumber getRandomPhone()
{
    {
        int[] anPhone = getRandomPhoneDigits();

        return new PhoneNumber((short)anPhone[0], (short)anPhone[1],
            (short)anPhone[2], anPhone[3]);
    }
}

/**
 * Return a random Zip code.
 *
 */
private static String getRandomZip()
{

```

```

        return Base.toDecString(Base.getRandom().nextInt(99999), 5);
    }

    /**
     * Return a random state.
     */
    private static String getRandomState()
    {
        return STATE_CODES[Base.getRandom().nextInt(STATE_CODES.length)];
    }

    /**
     * Return a random date in millis before or after the epoch.
     */
    private static long getRandomDateInMillis()
    {
        return (Base.getRandom().nextInt(40) - 20) * Contact.MILLIS_IN_YEAR;
    }

    /**
     * Generate a Contact with random information.
     */
    public static Contact getRandomContact()
    {
        return new Contact("John",
            getRandomName(),
            new Address("1500 Boylston St.", null, getRandomName(),
                getRandomState(), getRandomZip(), "US"),
            new Address("8 Yawkey Way", null, getRandomName(),
                getRandomState(), getRandomZip(), "US"),
            Collections.singletonMap("work", getRandomPhone()),
            new Date(getRandomDateInMillis()));
    }

    // ----- constants -----

    /**
     * US Postal Service two letter postal codes.
     */
    private static final String[] STATE_CODES =
    {
        "AL", "AK", "AS", "AZ", "AR", "CA", "CO", "CT", "DE", "OF", "DC",
        "FM", "FL", "GA", "GU", "HI", "ID", "IL", "IN", "IA", "KS", "KY",
        "LA", "ME", "MH", "MD", "MA", "MI", "MN", "MS", "MO", "MT", "NE",
        "NV", "NH", "NJ", "NM", "NY", "NC", "ND", "MP", "OH", "OK", "OR",
        "PW", "PA", "PR", "RI", "SC", "SD", "TN", "TX", "UT", "VT", "VI",
        "VA", "WA", "WV", "WI", "WY"
    };

    /**
     * Default contacts file name.
     */
    public static final String FILENAME = "contacts.csv";
}

```

The `DataGenerator` class generates, at random, information about an employee. The information includes the employee's name, home address, work address, home telephone number, work telephone number, and the employee's age. The information

is stored in a CSV-formatted file named `contacts.csv`. This file follows the standard rules for a CSV-formatted file, where there is one record per line, and individual fields within the record are separated by commas.

[Example 5-4](#) illustrates the first few entries of the `contacts.csv` file.

#### **Example 5-4 Contents of the `contacts.csv` File**

```
John,Dvcqbvcpc,669 Beacon St.,,Tetuvusz,CA,68457,US,Yoyodyne Propulsion Systems,330 Lectroid Rd.
,Grover's Mill,KS,30344,US,home,11,98,183,8707139,work,11,96,425,1175949,63072000000
John,Fxsr,549 Beacon St.,,Jutaswaby,MI,16315,US,Yoyodyne Propulsion Systems,330 Lectroid Rd.
,Grover's Mill,CT,60309,US,home,11,20,40,3662989,work,11,41,187,3148474,-189216000000
John,Gmyrolvfyd,73 Beacon St.,,Lpnztf,AR,82667,US,Yoyodyne Propulsion Systems,330 Lectroid Rd.
,Grover's Mill,NY,42297,US,home,11,22,17,8579970,work,11,35,338,9286245,-567648000000
John,Efmpjlbj,85 Beacon St.,,Wyswpb,AK,29590,US,Yoyodyne Propulsion Systems,330 Lectroid Rd.
,Grover's Mill,NJ,06219,US,home,11,20,744,4331451,work,11,39,911,9104590,-157680000000
...
```

## **Create a Console Application to Load the Cache**

Create a Java class called `LoaderExample`. See ["Creating a Java Class"](#) on page 2-11 for detailed information.

Implement the class to load the cache with employee data generated by the program described in ["Create the Data Generator"](#) on page 5-6. Use input streams and buffered readers to load the employee information in the `contacts.csv` file into a single Coherence cache.

Add code to parse the employee information in the data file. After you have this information, create the individual contacts to put into the cache. To conserve processing effort and minimize network traffic, use the `putAll` method to load the cache.

[Example 5-5](#) illustrates a possible implementation of the `LoaderExample` class.

#### **Example 5-5 Sample Cache Loading Program**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import com.oracle.handson.ContactId;
import com.oracle.handson.Address;
import com.oracle.handson.PhoneNumber;
import com.oracle.handson.Contact;

import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

import java.sql.Date;

import java.util.HashMap;
import java.util.Map;

/**
 * LoaderExample loads contacts into the cache from a file.
```

```

*
*/
public class LoaderExample
{
    // ----- static methods -----

    /**
     * Load contacts.
     */
    public static void main (String[] asArg)
        throws IOException
    {
        String sFile = asArg.length > 0 ? asArg[0] : DataGenerator.FILENAME;
        String sCache = asArg.length > 1 ? asArg[1] : CACHENAME;

        System.out.println("input file: " + sFile);
        System.out.println("cache name: " + sCache);

        new LoaderExample().load(CacheFactory.getCache(sCache),
            new FileInputStream(sFile));
        CacheFactory.shutdown();
    }

    /**
     * Load cache from stream.
     */
    public void load(NamedCache cache, InputStream in)
        throws IOException
    {
        BufferedReader reader = new BufferedReader(new InputStreamReader(in));
        Map mapBatch = new HashMap(1024);
        String sRecord;
        int cRecord = 0;

        while ((sRecord = reader.readLine()) != null)
        {
            // parse record
            String[] asPart = sRecord.split(",");
            int ofPart = 0;
            String sFirstName = asPart[ofPart++];
            String sLastName = asPart[ofPart++];
            ContactId id = new ContactId(sFirstName, sLastName);
            Address addrHome = new Address(
                /*streetline1*/ asPart[ofPart++],
                /*streetline2*/ asPart[ofPart++],
                /*city*/ asPart[ofPart++],
                /*state*/ asPart[ofPart++],
                /*zip*/ asPart[ofPart++],
                /*country*/ asPart[ofPart++]);
            Address addrWork = new Address(
                /*streetline1*/ asPart[ofPart++],
                /*streetline2*/ asPart[ofPart++],
                /*city*/ asPart[ofPart++],
                /*state*/ asPart[ofPart++],
                /*zip*/ asPart[ofPart++],
                /*country*/ asPart[ofPart++]);
            Map mapTelNum = new HashMap();

            for (int c = asPart.length - 1; ofPart < c; )

```

```

        {
            mapTelNum.put(/*type*/ asPart[ofPart++], new PhoneNumber(
                /*access code*/ Short.parseShort(asPart[ofPart++]),
                /*country code*/ Short.parseShort(asPart[ofPart++]),
                /*area code*/ Short.parseShort(asPart[ofPart++]),
                /*local num*/ Integer.parseInt(asPart[ofPart++]));
        }
        Date dtBirth = new Date(Long.parseLong(asPart[ofPart]));

        // Construct Contact and add to batch
        Contact con1 = new Contact(sFirstName, sLastName, addrHome,
            addrWork, mapTelNum, dtBirth);

        System.out.println(con1);
        mapBatch.put(id, con1);

        ++cRecord;
        if (cRecord % 1024 == 0)
        {
            // load batch
            cache.putAll(mapBatch);
            mapBatch.clear();
            System.out.print('.');
            System.out.flush();
        }
    }

    if (!mapBatch.isEmpty())
    {
        // load final batch
        cache.putAll(mapBatch);
    }

    System.out.println("Added " + cRecord + " entries to cache");
}

// ----- constants -----

/**
 * Default cache name.
 */
public static final String CACHENAME = "ContactsCache";
}

```

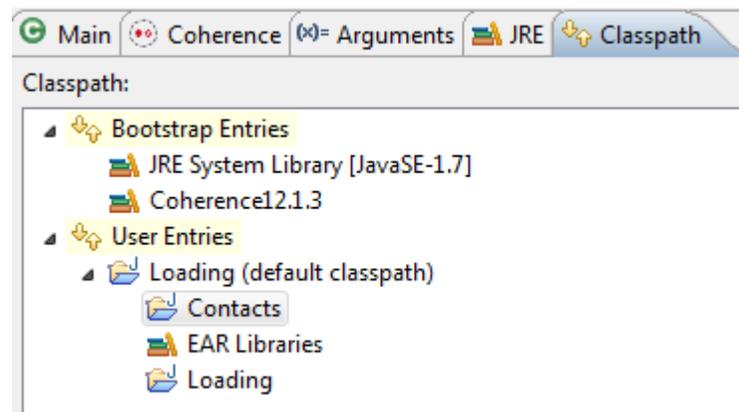
## Run the Cache Loading Example

To run the cache loading example:

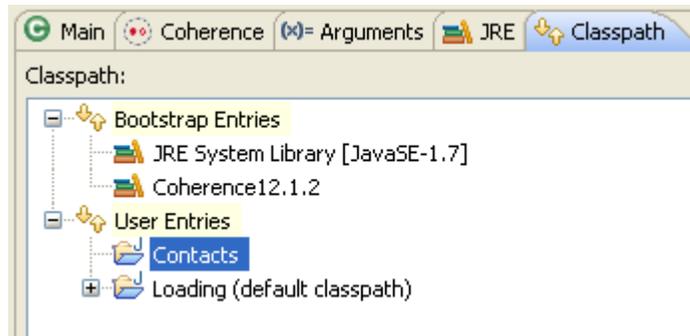
1. Stop any running cache servers. See ["Stopping Cache Servers"](#) on page 2-14 for detailed information.
2. Start a cache server with the `ContactsCacheServer` file.
  - a. Right click the project and select **Run As** then **Run Configurations** to edit the `ContactsCacheServer` configuration. In the **Main** tab, click **Browse** and select the **Loading** project.
  - b. In the **Classpath** tab, select **User Entries** and click **Add Projects**. In the **Project Selection** dialog box, select the **Loading** project. Click **OK**.

- c. In the **Common** tab, click **Shared file** and browse for the **Loading** project.
  - d. Click **Apply**, then **Run**.
3. Create a run configuration for DataGenerator. Right click DataGenerator in the **Project Explorer** and select **Run As**. In the **Run Configurations** dialog box select **Oracle Coherence** and click the **New Configuration** icon.
    - a. In the **Name** field, enter DataGenerator.
    - b. In the **Project** field in the **Main** tab, enter Loading. In the **Main class** field, enter `com.oracle.handson.DataGenerator`.
    - c. In the **General** tab of the **Coherence** tab, browse to the `c:\home\oracle\workspace\Contacts\appClientModule\contacts-cache-config.xml` file in the **Cache configuration descriptor** field. Select the **Disabled (cache client)** button. Enter 3155 in the **Cluster port** field. Click **Apply**.  
In the **Other** tab, scroll down to the **tangosol.pof.config** field. Enter the absolute path to the POF configuration file `contacts-pof-config.xml`. Click **Apply**.
    - d. In the **Common** tab, select **Shared file** and browse for the **Loading** directory.
    - e. Examine the contents of **Loading** in the **Classpath** tab. **Contacts** should appear as one of the entries under **Loading**, as in [Figure 5-3](#).

**Figure 5-3 Classpath for the DataGenerator Program**



4. Create a run configuration for LoaderExample.
  - In the **Name** field, enter LoaderGenerator
  - In the **Project** field in the **Main** tab, enter Loading. In the **Main class** field, enter `com.oracle.handson.LoaderExample`.
  - In the **General** tab of the **Coherence** tab, browse to the `c:\home\oracle\workspace\Contacts\appClientModule\contacts-cache-config.xml` file in the **Cache configuration descriptor** field. Select the **Disabled (cache client)** button. Enter 3155 in the **Cluster port** field. Click **Apply**.  
In the **Other** tab, scroll down to the **tangosol.pof.config** field. Enter the absolute path to the POF configuration file `contacts-pof-config.xml`. Click **Apply**.
  - Examine the contents of the **Classpath** tab. **Contacts** should appear as one of the entries under **Loading**, as in [Figure 5-4](#). If it does not appear, click **User Entries**, then add the project with the **Add Project** button.

**Figure 5–4 Classpath for the LoaderExample Program**

5. Run the `DataGenerator` configuration from the **Run Configurations** dialog box. Then run the `LoaderExample` configuration.

The `DataGenerator` run configuration generates a data file; you should not see any output or response in the Eclipse console window. The `LoaderGenerator` run configuration sends a stream of employee contact information, similar to [Example 5–6](#), to the Eclipse console window.

#### **Example 5–6 Output from the Sample Cache Loading Program**

...

```
John Itqx
Addresses
Home: 58 Beacon St.
```

```
Fcfz, NY 84297
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, RI 29443
US
Telephone Numbers
work: +11 93 148 8101744
home: +11 70 29 5079314
Birth Date: 1975-12-30
```

```
John Ysxydw
Addresses
Home: 623 Beacon St.
```

```
Ohirim, PR 93378
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, FL 82251
US
Telephone Numbers
work: +11 5 799 6030449
home: +11 14 213 7774277
Birth Date: 1957-01-03
```

```
John Htmzlvoig
Addresses
Home: 700 Beacon St.
```

```
Ogkqwbnvj, SC 45698
US
```

```

Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, MI 74783
US
Telephone Numbers
work: +11 65 692 375834
home: +11 73 881 452999
Birth Date: 1973-12-30
Added 1000 entries to cache
2013-11-21 12:52:47.980/5.742 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=n/a): Service PartitionedPofCache left the
cluster
...

```

## Querying and Aggregating Data in the Cache

This exercise introduces querying and aggregating data in a cache. This exercise shows you how to:

- Query the cache for specific data
- Aggregate information within the cache

After putting complex objects in the named caches, you can query and aggregate information within the grid. The `com.tangosol.util.QueryMap` interface provides methods for managing the values or keys within a cache. You can use filters to restrict your results. You can also define indexes to optimize your queries.

Because Coherence serializes information when storing it, you also have the overhead of deserializing when querying information. When indexes are added, the *values* kept in the index itself are deserialized and, therefore, are quicker to access. The *objects* that are indexed are always kept serialized.

Some of the often-used methods in the `QueryMap` interface are:

- `Set entrySet(Filter filter)`, which returns a set of entries that are contained in the map that satisfy the filter
- `addIndex(ValueExtractor extractor, boolean fOrdered, Comparator comparator)`, which adds an index
- `Set keySet(Filter filter)`, which is similar to `entrySet`, but returns keys, not values

It is important to note that filtering occurs at Cache Entry Owner level. In a partitioned topology, filtering can be performed in parallel because it is the primary partitions that do the filtering. The `QueryMap` interface uses the `Filter` classes. You can find more information on these classes in the API for the `com.tangosol.util.filter` package.

All Coherence `NamedCache` objects implement the `com.tangosol.util.QueryMap` interface. This enables `NamedCache` objects to support searching for keys or entries in a cache that satisfy a specified condition. The condition can be represented as an object that implements the `com.tangosol.util.Filter` interface.

The `com.tangosol.util.filter` package contains several predefined classes that provide implementations of standard query expressions. Examples of these classes include `GreaterFilter`, `GreaterEquals`, `LikeFilter`, `NotEqualsFilter`, `InFilter`, and so on. You can use these filters to construct object-based equivalents of most SQL `WHERE` clause expressions.

---

---

**Note:** Coherence does not provide a `SQLFilter` because it is unlikely that the objects placed in a cache are modeled in a relational manner, that is, using rows and columns (as they are typically represented in a database). Additionally, it is common that objects placed in a cache are not easily modeled using relational models, for example, large BLOBS.

---

---

The `Filter` classes use standard Java method reflection to represent test conditions. For example, the following filter represents the condition where the value returned from the `getHomeAddress.getState` method on an object in a cache is for Massachusetts (MA):

```
(new EqualsFilter("getHomeAddress.getState", "MA");
```

If the object tested with this filter does not have a `get` method, then the test fails.

Here are additional examples of using the `Filter` classes:

- Return a set of people who live in a city whose name begins with the letter S:

```
Set sPeople = cache.entrySet(new LikeFilter("getHomeAddress.getCity", "S%");
```

- Return a set containing people over the age of 42:

```
final int nAge = 42;  
// Find all contacts who are older than nAge  
Set sSeniors = cache.entrySet(new GreaterFilter("getAge", nAge));
```

In addition to the `entrySet` and `keySet` methods defined by the `QueryMap` interface, Coherence lets you define indexes to improve query performance by using the `addIndex` method. Unlike relational database systems, where indexes are defined according to well-known and strictly enforced collections of named columns (that is, a schema), Coherence does not have a schema. Because there is no schema, you define indexes differently from a traditional database.

To define the values that are to be indexed for each object placed in a cache, Coherence introduces the concept of a value extractor. The `com.tangosol.util.ValueExtractor` interface defines an `extract` method. If given an object parameter, a `ValueExtractor` implementation returns a value based on the parameter.

A simple example of a `ValueExtractor` implementation is the `com.tangosol.util.extractor.ReflectionExtractor` interface, which uses reflection to return the result of a method call on an object, for example:

```
new ReflectionExtractor("getCity")
```

You can use value extractors throughout the Coherence API. Typically, however, they are used to define indexes.

An especially useful type of extractor is the `ChainedExtractor`. This is a composite `ValueExtractor` implementation based on an array of extractors. Extractors in the array are applied sequentially, from left to right. The result of a previous extractor becomes the target object for a next extractor, for example:

```
new ChainedExtractor(new ReflectionExtractor("getHomeAddress"), new  
ReflectionExtractor("getState"))
```

This example assumes that the `HomeAddress` and `State` objects belong to a complex `Contact` object. `ChainedExtractor` first uses reflection to call the

getHomeAddress method on each cached Contact object, and then uses reflection to call the getState method on the set of returned HomeAddress objects.

1. [Create the Class to Query Cache Data](#)
2. [Run the Query Example](#)
3. [Edit the Query Example to Perform Aggregations](#)
4. [Run the Query and Aggregation Example](#)

## Create the Class to Query Cache Data

Create a new Java class called QueryExample to perform your queries. Ensure that the class has a main method. See ["Creating a Java Class"](#) on page 2-11 for detailed information.

Use the entrySet method to get the employee contact information for:

- All employees who live in Massachusetts:

```
cache.entrySet(new EqualsFilter("getHomeAddress.getState", "MA"));
```

- All employees who live in Massachusetts and work elsewhere:

```
cache.entrySet(new AndFilter(
    new EqualsFilter("getHomeAddress.getState", "MA"),
    new NotEqualsFilter("getWorkAddress.getState", "MA")));
```

- All employees whose city name begins with an S:

```
cache.entrySet(new LikeFilter("getHomeAddress.getCity", "S%"));
```

- All employees whose last name begins with an S and live in Massachusetts. Use both the key and value in the query. Note that the cache entries use ContactId objects as the keys. You can use the KeyExtractor to get these values. KeyExtractor is a specialized value extractor that indicates that a query be run against the key objects, rather than the values:

```
cache.entrySet(new AndFilter(
    new LikeFilter(new KeyExtractor("getLastName"), "S%",
        (char) 0, false),
    new EqualsFilter("getHomeAddress.getState", "MA")));
```

- All employees who are older than a specified age. Hint:

```
final int nAge = 42;
setResults = cache.entrySet(new GreaterFilter("getAge", nAge));
```

Use indexes to improve performance. Hint: Find the addIndex method in the Javadoc for the QueryMap interface.

[Example 5-7](#) illustrates a possible implementation of the QueryExample class.

### **Example 5-7 Sample QueryExample Class**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import com.tangosol.util.extractor.ChainedExtractor;
import com.tangosol.util.extractor.KeyExtractor;
```

```

import com.tangosol.util.extractor.ReflectionExtractor;

import com.tangosol.util.filter.AlwaysFilter;
import com.tangosol.util.filter.AndFilter;
import com.tangosol.util.filter.EqualsFilter;
import com.tangosol.util.filter.GreaterFilter;
import com.tangosol.util.filter.LikeFilter;
import com.tangosol.util.filter.NotEqualsFilter;

import java.util.Iterator;
import java.util.Set;

/**
 * QueryExample runs sample queries for contacts.
 *
 */
public class QueryExample{

    // ----- QueryExample methods -----

    public static void main(String[] args) {
        NamedCache cache = CacheFactory.getCache("ContactsCache");
        query(cache);
    }
    /**
     * Perform the example queries
     *
     */
    public static void query(NamedCache cache)
    {
        // Add indexes to make queries more efficient
        ReflectionExtractor reflectAddrHome =
            new ReflectionExtractor("getHomeAddress");

        // Add an index for the age
        cache.addIndex(new ReflectionExtractor("getAge"), true, null);

        // Add index for state within home address
        cache.addIndex(new ChainedExtractor(reflectAddrHome,
            new ReflectionExtractor("getState")), true, null);

        // Add index for state within work address
        cache.addIndex(new ChainedExtractor(
            new ReflectionExtractor("getWorkAddress"),
            new ReflectionExtractor("getState")), true, null);

        // Add index for city within home address
        cache.addIndex(new ChainedExtractor(reflectAddrHome,
            new ReflectionExtractor("getCity")), true, null);

        // Find all contacts who live in Massachusetts
        Set setResults = cache.entrySet(new EqualsFilter(
            "getHomeAddress.getState", "MA"));
        printResults("MA Residents", setResults);

        // Find all contacts who live in Massachusetts and work elsewhere
        setResults = cache.entrySet(new AndFilter(
            new EqualsFilter("getHomeAddress.getState", "MA"),
            new NotEqualsFilter("getWorkAddress.getState", "MA")));
    }
}

```

```

printResults("MA Residents, Work Elsewhere", setResults);

// Find all contacts whose city name begins with 'S'
setResults = cache.entrySet(new LikeFilter("getHomeAddress.getCity",
    "S%"));
printResults("City Begins with S", setResults);

final int nAge = 42;
// Find all contacts who are older than nAge
setResults = cache.entrySet(new GreaterFilter("getAge", nAge));
printResults("Age > " + nAge, setResults);

// Find all contacts with last name beginning with 'S' that live
// in Massachusetts. Uses both key and value in the query.
setResults = cache.entrySet(new AndFilter(
    new LikeFilter(new KeyExtractor("getLastName"), "S%",
        (char) 0, false),
    new EqualsFilter("getHomeAddress.getState", "MA")));
printResults("Last Name Begins with S and State Is MA", setResults);

}

/**
 * Print results of the query
 *
 * @param sTitle    the title that describes the results
 * @param setResults a set of query results
 */
private static void printResults(String sTitle, Set setResults)
{
    System.out.println(sTitle);
    for (Iterator iter = setResults.iterator(); iter.hasNext(); )
    {
        System.out.println(iter.next());
    }
}
}

```

## Run the Query Example

To run the query example:

1. Stop all running cache servers. See ["Stopping Cache Servers"](#) on page 2-14 for more information.
2. Create a run configuration for the `QueryExample` class.
  - a. Right click `QueryExample` in the **Project Navigator** and select **Run As** then **Run Configurations**. Select **Oracle Coherence** and click the **New launch configuration** icon.
  - b. In the **Name** field, enter `QueryExample`.
  - c. In the **Project** field in the **Main** tab, enter `Loading`. In the **Main class** field, enter `com.oracle.handson.QueryExample`.
  - d. In the **General** tab of the **Coherence** tab, browse to the `c:\home\oracle\workspace\Contacts\appClientModule\contacts-cache-config.xml` file in the **Cache configuration descriptor** field. Select the **Disabled (cache client)** button. Enter `3155` in the **Cluster port** field. Click **Apply**.

In the **Other** tab, scroll down to the **tangosol.pof.config** field. Enter the absolute path to the POF configuration file `contacts-pof-config.xml`. Click **Apply**.

- e. Examine the contents of the **Classpath** tab. The **Contacts** project should appear under the **Loading** project in **User Entries**.
3. Restart the `ContactsCacheServer`.
4. Run the `DataGenerator` class, the `LoaderExample` class, then the `QueryExample` class.

After printing all of the contact information in the cache, the `QueryExample` file displays the results of the queries. The results should look similar to [Example 5-8](#).

**Example 5-8 Results of the QueryExample Program**

```
...
MA Residents
ConverterEntry{Key="John Ueccc", Value="John Ueccc
Addresses
Home: 285 Beacon St.

Oxtqwisgti, MA 41063
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, VA 28107
US
Telephone Numbers
work: +11 48 835 1876678
home: +11 78 482 1247744
Birth Date: 1972-12-30"}
...
MA Residents, Work Elsewhere
ConverterEntry{Key="John Ueccc", Value="John Ueccc
Addresses
Home: 285 Beacon St.

Oxtqwisgti, MA 41063
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, VA 28107
US
Telephone Numbers
work: +11 48 835 1876678
home: +11 78 482 1247744
Birth Date: 1972-12-30"}
...
City Begins with S
ConverterEntry{Key="John Frepojf", Value="John Frepojf
Addresses
Home: 851 Beacon St.

Swnsfng, PR 00734
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, AR 97794
US
```

```

Telephone Numbers
work: +11 10 474 781020
home: +11 29 575 9284939
Birth Date: 1985-12-27"}
...
Age > 42
ConverterEntry{Key="John Qghbguy", Value="John Qghbguy"}
Addresses
Home: 49 Beacon St.

Dvftzpq, PR 34220
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, MD 28247
US
Telephone Numbers
work: +11 39 563 7113013
home: +11 58 910 4667915
Birth Date: 1961-01-02"}
...
Last Name Begins with S and State Is MA
ConverterEntry{Key="John Snnfg", Value="John Snnfg"}
Addresses
Home: 178 Beacon St.

Hbpeak, MA 64081
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, OK 92191
US
Telephone Numbers
work: +11 56 322 7307404
home: +11 33 928 3075361
Birth Date: 1978-12-29"}
...

```

## Edit the Query Example to Perform Aggregations

Add code to the `QueryExample.java` file to perform aggregations on the cache data. An entry aggregator (`com.tangosol.util.InvocableMap.EntryAggregator`) enables you to perform operations on all or a specific set of objects and return an aggregation. `EntryAggregator` instances are agents that execute services in parallel against the data within the cluster. Aggregations are performed in parallel and can benefit from the addition of cluster members.

There are two ways of aggregating: aggregate over a collection of keys or by specifying a filter. [Example 5-9](#) illustrates the `EntryAggregator` methods that perform these tasks.

### **Example 5-9** *Methods to Aggregate Over Keys or by Specifying Filters*

```
Object aggregate(Collection keys, InvocableMap.entryAggregator agg)
```

```
Object aggregate(Filter filter, InvocableMap.entryAggregator agg)
```

To add aggregations to return data for filtering:

1. Using aggregations, write code in the `QueryExample` class to calculate the following:
  - The number of employees that are older than a specified age. Use the `GreaterFilter` and the `Count` class:
 

```
cache.aggregate(new GreaterFilter("getAge", nAge), new Count())
```
  - Lowest age in the set of employees. Use the `AlwaysFilter` and the `LongMin` class:
 

```
cache.aggregate(AlwaysFilter.INSTANCE, new LongMin("getAge"))
```
  - Highest age in the set of employees. Use the `AlwaysFilter` and the `LongMax` class:
 

```
cache.aggregate(AlwaysFilter.INSTANCE, new LongMax("getAge"))
```
  - Average age of employees. Use the `AlwaysFilter` and the `DoubleAverage` class:
 

```
cache.aggregate(AlwaysFilter.INSTANCE, new DoubleAverage("getAge"))
```

2. Import the `Count`, `DoubleAverage`, `LongMax`, and `LongMin` aggregator classes.

```
import com.tangosol.util.aggregator.Count;
import com.tangosol.util.aggregator.DoubleAverage;
import com.tangosol.util.aggregator.LongMax;
import com.tangosol.util.aggregator.LongMin;
```

The `QueryExample.java` file looks similar to [Example 5–10](#).

**Example 5–10 QueryExample with Aggregation**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import com.tangosol.util.aggregator.Count;
import com.tangosol.util.aggregator.DoubleAverage;
import com.tangosol.util.aggregator.LongMax;
import com.tangosol.util.aggregator.LongMin;

import com.tangosol.util.extractor.ChainedExtractor;
import com.tangosol.util.extractor.KeyExtractor;
import com.tangosol.util.extractor.ReflectionExtractor;

import com.tangosol.util.filter.AlwaysFilter;
import com.tangosol.util.filter.AndFilter;
import com.tangosol.util.filter.EqualsFilter;
import com.tangosol.util.filter.GreaterFilter;
import com.tangosol.util.filter.LikeFilter;
import com.tangosol.util.filter.NotEqualsFilter;

import java.util.Iterator;
import java.util.Set;

/**
 * QueryExample runs sample queries for contacts.
 */
public class QueryExample{
```

```

// ----- QueryExample methods -----
public static void main(String[] args) {
    NamedCache cache = CacheFactory.getCache("ContactsCache");
    query(cache);
}
/**
 * Perform the example queries
 *
 */
public static void query(NamedCache cache)
{
    // Add indexes to make queries more efficient
    ReflectionExtractor reflectAddrHome =
        new ReflectionExtractor("getHomeAddress");

    cache.addIndex(new ReflectionExtractor("getAge"), true, null);
    cache.addIndex(new ChainedExtractor(reflectAddrHome,
        new ReflectionExtractor("getState")), true, null);
    cache.addIndex(new ChainedExtractor(
        new ReflectionExtractor("getWorkAddress"),
        new ReflectionExtractor("getState")), true, null);
    cache.addIndex(new ChainedExtractor(reflectAddrHome,
        new ReflectionExtractor("getCity")), true, null);

    // Find all contacts who live in Massachusetts
    Set setResults = cache.entrySet(new EqualsFilter(
        "getHomeAddress.getState", "MA"));
    printResults("MA Residents", setResults);

    // Find all contacts who live in Massachusetts and work elsewhere
    setResults = cache.entrySet(new AndFilter(
        new EqualsFilter("getHomeAddress.getState", "MA"),
        new NotEqualsFilter("getWorkAddress.getState", "MA")));
    printResults("MA Residents, Work Elsewhere", setResults);

    // Find all contacts whose city name begins with 'S'
    setResults = cache.entrySet(new LikeFilter("getHomeAddress.getCity",
        "S%"));
    printResults("City Begins with S", setResults);

    final int nAge = 42;
    // Find all contacts who are older than nAge
    setResults = cache.entrySet(new GreaterFilter("getAge", nAge));
    printResults("Age > " + nAge, setResults);

    // Find all contacts with last name beginning with 'S' that live
    // in Massachusetts. Uses both key and value in the query.
    setResults = cache.entrySet(new AndFilter(
        new LikeFilter(new KeyExtractor("getLastName"), "S%",
            (char) 0, false),
        new EqualsFilter("getHomeAddress.getState", "MA")));
    printResults("Last Name Begins with S and State Is MA", setResults);

    // Count contacts who are older than nAge
    System.out.println("count > " + nAge + ": " + cache.aggregate(
        new GreaterFilter("getAge", nAge), new Count()));

    // Find minimum age

```

```

        System.out.println("min age: " + cache.aggregate(AlwaysFilter.INSTANCE,
            new LongMin("getAge")));

        // Calculate average age
        System.out.println("avg age: " + cache.aggregate(AlwaysFilter.INSTANCE,
            new DoubleAverage("getAge")));

        // Find maximum age
        System.out.println("max age: " + cache.aggregate(AlwaysFilter.INSTANCE,
            new LongMax("getAge")));
    }

    /**
     * Print results of the query
     */
    private static void printResults(String sTitle, Set setResults)
    {
        System.out.println(sTitle);
        for (Iterator iter = setResults.iterator(); iter.hasNext(); )
        {
            System.out.println(iter.next());
        }
    }
}

```

## Run the Query and Aggregation Example

To query the cache and aggregate the results:

1. Stop the Contacts cache server, ContactsCacheServer. See ["Stopping Cache Servers"](#) on page 2-14 for more information.
2. Restart ContactsCacheServer.
3. Run the DataGenerator, LoaderExample, and QueryExample applications.

The output should look similar to [Example 5-11](#) in the Eclipse Console.

### **Example 5-11 Output from the Aggregators**

```

...
Last Name Begins with S and State Is MA
ConverterEntry{Key="John Sqmyas", Value="John Sqmyas
Addresses
Home: 594 Beacon St.

Jxaxt, MA 10081
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, NJ 95236
US
Telephone Numbers
work: +11 70 837 4723938
home: +11 2 227 6816592
Birth Date: 1977-12-29"}
count > 42: 482
min age: 22
avg age: 41.627
max age: 61

```

---

## Simplifying Cache Calls and Aggregations

This exercise illustrates how you can simplify Java expressions that use aggregations and filters by using the Coherence `createFilter` and `createExtractor` factory methods in the `QueryHelper` class.

In [Chapter 5, "Loading Data Into a Cache,"](#) you created a file, `QueryExample.java`, that used a variety of filters, such as `AlwaysFilter`, `LikeFilter`, and `EqualsFilter`, to pull and aggregate information from the cache. The file also created indexes on the data by using a variety of specialized `ValueExtractors`: `ChainedExtractors`, `KeyExtractors`, and `ReflectionExtractors`. This created some verbose Java statements. These statements can be simplified by using the `QueryHelper` API.

This chapter contains the following sections:

- [Introduction](#)
- [Simplifying the Query Example](#)
- [Rerunning the Query Example](#)

### Introduction

To simplify filter and extractor statements, and the way in which you interact with the Coherence caches, Coherence provides the `QueryHelper` API. `QueryHelper` (`com.tangosol.util.QueryHelper`) is a utility class that provides a set of `createFilter` and `createExtractor` factory methods that can build instances of `Filter` and `ValueExtractor`. The methods in the class accept a `String` data type that specifies the creation of rich Filters in a format that is familiar to anyone who understands SQL `WHERE` clauses.

For example, the following statement uses `createFilter(String s)` to construct a filter for employees who live in Massachusetts but work in another state.

```
..
QueryHelper.createFilter("homeAddress.state = 'MA' and workAddress.state !=
'MA'")
...
```

This statement is more simple and easier to read than the equivalent filter and extractor statement using the Coherence API:

```
new AndFilter(new EqualsFilter("getHomeAddress.getState", "MA"),
    new NotEqualsFilter("getWorkAddress.getState", "MA"))
```

For more information, see the Javadoc for the `QueryHelper` API. For information on the syntax of the `WHERE` clause within the Coherence Query Language, see "Using Coherence Query Language" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

## Simplifying the Query Example

This section describes how you can simplify the indexes, cache calls, and aggregations in the `QueryExample.java` file that you created in the previous chapter.

1. Import the `QueryHelper` API into the `QueryExample.java` file as a static class.
 

```
import static com.tangosol.util.QueryHelper.*;
```
2. Comment out the imports for the `ChainedExtractor`, `KeyExtractor`, and `ReflectionExtractor` classes.
3. Comment out the imports for the `AlwaysFilter`, `AndFilter`, `EqualsFilter`, `GreaterFilter`, `LikeFilter`, and `NotEqualsFilter` classes.
4. In the `cache.addIndex` statements, replace instances of `ReflectionExtractor` with `createExtractor` from the `QueryHelper` API.

[Table 6–1](#) lists the `ReflectionExtractor` instances and their `createExtractor` equivalents.

**Table 6–1 ReflectionExtractors and Their Equivalent createExtractor Statements**

Replace This ReflectionExtractor Statement ...	With the Equivalent createExtractor Statement ...
<code>cache.addIndex(new ReflectionExtractor("getAge"), true, null);</code>	<code>cache.addIndex(createExtractor("age"), true, null);</code>
<code>cache.addIndex(new ChainedExtractor(reflectAddrHome, new ReflectionExtractor("getState")), true, null);</code>	<code>cache.addIndex(createExtractor("homeAddress.state"), false, null);</code>
<code>cache.addIndex(new ChainedExtractor(new ReflectionExtractor("getWorkAddress"), new ReflectionExtractor("getState")), true, null);</code>	<code>cache.addIndex(createExtractor("workAddress.state"), false, null);</code>
<code>cache.addIndex(new ChainedExtractor(reflectAddrHome, new ReflectionExtractor("getCity")), true, null);</code>	<code>cache.addIndex(createExtractor("homeAddress.city"), true, null);</code>

5. Replace the calls to the `*Filter` methods in the `setResults` statements with calls to `createFilter` with the appropriate Coherence Query Language.

[Table 6–2](#) lists the `*Filter` instances and their `createFilter` equivalents.

**Table 6–2** \*Filter Statements and Their Equivalent createFilter Statements in Queries

Replace This *Filter Statement ...	With the Equivalent createFilter Statement ...
<pre>Set setResults = cache.entrySet(new EqualsFilter("getHomeAddress.getState", "MA"));</pre>	<pre>Set setResults = cache.entrySet(createFilter("homeAddress .state = 'MA'"));</pre>
<pre>Set setResults = cache.entrySet(new AndFilter(new EqualsFilter("getHomeAddress.getState", "MA"), new NotEqualsFilter("getWorkAddress.getState", "MA")));</pre>	<pre>setResults = cache.entrySet( createFilter("homeAddress.state is 'MA' and workAddress is not 'MA'"));</pre>
<pre>Set setResults = cache.entrySet(new LikeFilter("getHomeAddress.getCity", "S%"));</pre>	<pre>Set setResults = cache.entrySet(createFilter("homeAddress .city like 'S%'"));</pre>
<pre>Set setResults = cache.entrySet(new GreaterFilter("getAge", nAge));</pre>	<pre>// Initialize nAge and aEnv final int nAge = 42;  Object[] aEnv = new Object[] {new Integer(nAge)};  ...  Set setResults = cache.entrySet(createFilter("age &gt; ?1", aEnv));</pre>
<pre>Set setResults = cache.entrySet(new AndFilter(new LikeFilter(new KeyExtractor("getLastName"), "S%", (char) 0, false), new EqualsFilter("getHomeAddress.getState", "MA")));</pre>	<pre>Set setResults = cache.entrySet(createFilter("key(lastNam e) like 'S%' and homeAddress.state = 'MA'"));</pre>

- Replace the calls to the \*Filter methods in the aggregate statements with calls to createFilter with the appropriate Coherence Query Language.

Table 6–3 lists the \*Filter instances and their createFilter equivalents.

**Table 6–3** Filter Statements and Their Equivalent createFilter Statements in Aggregations

Replace This *Filter Statement ...	With the Equivalent createFilter Statement ...
<pre>System.out.println("count &gt; " + nAge + ": "+ cache.aggregate(new GreaterFilter("getAge", nAge), new Count()));</pre>	<pre>System.out.println("count &gt; " + nAge + ": " + cache.aggregate(createFilter("age &gt; ?1", aEnv), new Count()));</pre>
<pre>System.out.println("min age: " + cache.aggregate(AlwaysFilter.INSTANCE, new LongMin("getAge")));</pre>	<pre>Filter always = createFilter("true"); System.out.println("min age: " + cache.aggregate(always, new LongMin("getAge")));</pre>
<pre>System.out.println("avg age: " + cache.aggregate(AlwaysFilter.INSTANCE, new DoubleAverage("getAge")));</pre>	<pre>System.out.println("avg age: " + cache.aggregate(always, new DoubleAverage("getAge")));</pre>
<pre>System.out.println("max age: " + cache.aggregate(AlwaysFilter.INSTANCE, new LongMax("getAge")));</pre>	<pre>System.out.println("max age: " + cache.aggregate(always, new LongMax("getAge")));</pre>

When you are finished with the code replacements, QueryExample.java looks similar to [Example 6–1](#).

**Example 6-1 Edited QueryExample File**

```

package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import com.tangosol.util.Filter;
import static com.tangosol.util.QueryHelper.*;

import com.tangosol.util.aggregator.Count;
// import com.tangosol.util.extractor.ChainedExtractor;
// import com.tangosol.util.extractor.KeyExtractor;
// import com.tangosol.util.extractor.ReflectionExtractor;

// import com.tangosol.util.aggregator.Count;
import com.tangosol.util.aggregator.DoubleAverage;
import com.tangosol.util.aggregator.LongMax;
import com.tangosol.util.aggregator.LongMin;

// import com.tangosol.util.filter.AlwaysFilter;
// import com.tangosol.util.filter.AndFilter;
// import com.tangosol.util.filter.EqualsFilter;
// import com.tangosol.util.filter.GreaterFilter;
// import com.tangosol.util.filter.LikeFilter;
// import com.tangosol.util.filter.NotEqualsFilter;

import java.util.Iterator;
import java.util.Set;

/**
 * QueryExample runs sample queries for contacts.
 *
 */
public class QueryExample{

    // ----- QueryExample methods -----

    public static void main(String[] args) {
        NamedCache cache = CacheFactory.getCache("ContactsCache");
        query(cache);
    }
    /**
     * Perform the example queries
     */
    public static void query(NamedCache cache)
    {
        // Add indexes to make queries more efficient
        // ReflectionExtractor reflectAddrHome =
        //     new ReflectionExtractor("getHomeAddress");

        // Add an index for the age
        // cache.addIndex(new ReflectionExtractor("getAge"), true, null);
cache.addIndex(createExtractor("age"), true, null);

        // Add index for state within home address
        // cache.addIndex(new ChainedExtractor(reflectAddrHome,
        //     new ReflectionExtractor("getState")), true, null);

```

```

cache.addIndex(createExtractor("homeAddress.state"), false, null);

// Add index for state within work address
// cache.addIndex(new ChainedExtractor(
//     new ReflectionExtractor("getWorkAddress"),
//     new ReflectionExtractor("getState")), true, null);
cache.addIndex(createExtractor("workAddress.state"), false, null);

// Add index for city within home address
// cache.addIndex(new ChainedExtractor(reflectAddrHome,
//     new ReflectionExtractor("getCity")), true, null);
cache.addIndex(createExtractor("homeAddress.city"), true, null);

// Find all contacts who live in Massachusetts
// Set setResults = cache.entrySet(new EqualsFilter(
//     "getHomeAddress.getState", "MA"));
Set setResults = cache.entrySet(createFilter("homeAddress.state = 'MA'"));
    printResults("MA Residents", setResults);

// Find all contacts who live in Massachusetts and work elsewhere
// setResults = cache.entrySet(new AndFilter(
//     new EqualsFilter("getHomeAddress.getState", "MA"),
//     new NotEqualsFilter("getWorkAddress.getState", "MA")));
setResults = cache.entrySet(createFilter("homeAddress.state is 'MA' and
workAddress is not 'MA'"));
    printResults("MA Residents, Work Elsewhere", setResults);

// Find all contacts whose city name begins with 'S'
// setResults = cache.entrySet(new LikeFilter("getHomeAddress.getCity",
//     "S%"));
setResults = cache.entrySet(createFilter("homeAddress.city like 'S%'"));
    printResults("City Begins with S", setResults);

final int nAge = 42;
Object[] aEnv = new Object[] {new Integer(nAge)};
// Find all contacts who are older than nAge
// setResults = cache.entrySet(new GreaterFilter("getAge", nAge));
setResults = cache.entrySet(createFilter("age > ?1", aEnv));
    printResults("Age > " + nAge, setResults);

// Find all contacts with last name beginning with 'S' that live
// in Massachusetts. Uses both key and value in the query.
// setResults = cache.entrySet(new AndFilter(
//     new LikeFilter(new KeyExtractor("getLastName"), "S%",
//     (char) 0, false),
//     new EqualsFilter("getHomeAddress.getState", "MA")));
setResults = cache.entrySet(createFilter("key(lastName) like 'S%' and
homeAddress.state = 'MA'"));
    setResults = cache.entrySet(createFilter("key().lastName like 'S%' and
    homeAddress.state = 'MA'"));
    printResults("Last Name Begins with S and State Is MA", setResults);

// Count contacts who are older than nAge
// System.out.println("count > " + nAge + ": "+
//     cache.aggregate(new GreaterFilter("getAge", nAge), new Count()));
System.out.println("count > " + nAge + ": " + cache.aggregate(
    createFilter("age > ?1", aEnv), new Count()));

// Find minimum age
// System.out.println("min age: " + cache.aggregate(AlwaysFilter.INSTANCE,

```

```

new LongMin("getAge"));
    Filter always = createFilter("true");
    System.out.println("min age: " + cache.aggregate(always, new
LongMin("getAge")));

    // Calculate average age
    // System.out.println("avg age: " + cache.aggregate(AlwaysFilter.INSTANCE,
new DoubleAverage("getAge"));
    System.out.println("avg age: " + cache.aggregate(always, new
DoubleAverage("getAge")));

    // Find maximum age
    // System.out.println("max age: " +
    // cache.aggregate(AlwaysFilter.INSTANCE, new LongMax("getAge"));
    System.out.println("max age: " + cache.aggregate(always, new
LongMax("getAge")));

System.out.println("-----QueryLanguageExample completed-----");

    }

/**
 * Print results of the query
 *
 * @param sTitle    the title that describes the results
 * @param setResults a set of query results
 */
private static void printResults(String sTitle, Set setResults)
{
    System.out.println(sTitle);
    for (Iterator iter = setResults.iterator(); iter.hasNext(); )
    {
        System.out.println(iter.next());
    }
}
}

```

## Rerunning the Query Example

To rerun the query example:

1. Stop any running cache servers. See ["Stopping Cache Servers"](#) on page 2-14 for more information.
2. Restart the `ContactsCacheServer`.
3. Run the `DataGenerator`, `LoaderExample`, and `QueryExample` files.
4. After printing all of the contact information in the cache, it displays the results of the queries. The results should look similar to the following examples.

---



---

**Note:** The `DataGenerator` program produces random names, cities and birthdates. Your output may be different.

---



---

[Example 6-2](#) illustrates the output of the MA Residents filter.

### **Example 6-2** Output of the MA Residents Filter

...

**MA Residents**

```

ConverterEntry{Key="John Hwdrrls", Value="John Hwdrrls
Addresses
Home: 369 Beacon St.

Fetggv, MA 24372
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, NE 84499
US
Telephone Numbers
work: +11 88 331 2307913
home: +11 64 86 2489621
Birth Date: 1976-12-29"}
...

```

[Example 6-3](#) illustrates the output of the MA Residents, Work Elsewhere filter.

**Example 6-3 Output of the MA Residents, Work Elsewhere Filter**

```

...
MA Residents, Work Elsewhere
ConverterEntry{Key="John Hwdrrls", Value="John Hwdrrls
Addresses
Home: 369 Beacon St.

Fetggv, MA 24372
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, NE 84499
US
Telephone Numbers
work: +11 88 331 2307913
home: +11 64 86 2489621
Birth Date: 1976-12-29"}
...

```

[Example 6-4](#) illustrates the output of the City Begins with S filter.

**Example 6-4 Output of the City Begins with S Filter**

```

...
City Begins with S
ConverterEntry{Key="John Pzek", Value="John Pzek
Addresses
Home: 309 Beacon St.

Saqrqy, OH 81353
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, CT 78117
US
Telephone Numbers
work: +11 28 790 2035988
home: +11 61 470 7634708
Birth Date: 1971-12-31"}
...

```

[Example 6-5](#) illustrates the output of the age greater than 42 filter.

**Example 6-5 Output of the Age Greater than 42 Filter**

```
...
Age > 42
ConverterEntry{Key="John Gddurqqziy", Value="John Gddurqqziy
Addresses
Home: 613 Beacon St.

Cxyskdo, DE 28968
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, SD 07959
US
Telephone Numbers
work: +11 31 768 5136041
home: +11 87 22 3851589
Birth Date: 1958-01-03"}
...
```

[Example 6-6](#) illustrates the output of the Last Name Begins with S and State is MA filter and the output of the aggregators.

**Example 6-6 Output of the State and Age Aggregators**

```
Last Name Begins with S and State Is MA
ConverterEntry{Key="John Syaqlolj1", Value="John Syaqlolj1
Addresses
Home: 810 Beacon St.

Rgtaljwph, MA 07471
US
Work: 200 Newbury St.
Yoyodyne, Ltd.
Boston, MA 02116
US
Telephone Numbers
work: +11 37 18 1767648
home: +11 98 155 1073866
Birth Date: 1974-12-30"}
...
count > 42: 446
min age: 22
avg age: 41.126
max age: 61
```

---

---

## Listening for Changes and Modifying Data

In this exercise, you set up listeners to observe data changes within a Coherence cache. It highlights the use of the Coherence `ObservableMap`, `MapEvent`, `EventListener`, `EntryProcessor`, and `AbstractMapListener` APIs. You also learn how entry processors can be used to modify and process entries in the Coherence cache.

This chapter contains the following sections:

- [Introduction](#)
- [Creating a Cache Listener](#)
- [Responding to Changes in the Cache](#)

### Introduction

The `com.tangosol.util.ObservableMap` interface enables you to observe and act on the changes made to cache entries. It extends `java.util.EventListener` and uses the standard JavaBeans event model. All types of `NamedCache` instances implement this interface. To listen for an event, you register a `MapListener` (`com.tangosol.util.MapListener`) instance on the cache. `MapListener` instances are called on the client; that is, the listener code is executed in your client process.

There are multiple ways to listen for events:

- Listen for all events
- Listen for all events that satisfy a filter
- Listen for events on a particular object key

These listener tasks can be performed on a `NamedCache` by the `addMapListener` methods listed in [Example 7-1](#).

#### **Example 7-1 Listener Methods on a NamedCache**

```
void addMapListener(MapListener listener)

void addMapListener(MapListener listener, Filter filter, boolean fLite)

void addMapListener(MapListener listener, Object oKey, boolean fLite)
```

The `com.tangosol.util.MapEvent` class captures the object key, and the old and new values. You can specify a *Lite* event, in which the new and old values might not be present. [Example 7-2](#) describes a pattern for registering these methods against a `NamedCache`. This has been done as an anonymous class. You can use the `getOldValue` or `getNewValue` methods in the `MapEvent` class to get the entry for which the event gets fired.

**Example 7-2 Code Pattern for Registering an Event**

```

namedCache.addMapListener(new MapListener() {
    public void entryDeleted(MapEvent mapEvent)
    {
        // TODO... handle deletion event
    }
    public void entryInserted(MapEvent mapEvent)
    {
        // TODO... handle inserted event
    }
    public void entryUpdated(MapEvent mapEvent)
    {
        // TODO... handle updated event } }
})

```

## Creating a Cache Listener

This section describes how to create a Java class that listens on a `NamedCache` and responds to any changes it detects.

1. [Create a Class to Listen for Changes in the Cache](#)
2. [Run the Cache Listener Example](#)

### Create a Class to Listen for Changes in the Cache

In the `Loading` project, create the class that will listen for a new `Contact` object entry. Name the class `ObserverExample` and ensure that it has a `main` method. See ["Creating a Java Class"](#) on page 2-11 for detailed information.

Within this class, add a listener to display a message whenever a new `Contact` is updated to the cache. For example, use the following code to keep the Java process running until you read from the console. Otherwise, your program will immediately exit.

```

BufferedReader console = new BufferedReader(new InputStreamReader(System.in));
String text = console.readLine();

```

Within the class, create an inner class to extend `AbstractMapListener`. Implement the methods to insert, update, and delete the cache values. In this case, most of the work should be done in the `entryUpdated` method, based on the old and new values contained in a `MapEvent`.

[Example 7-3](#) illustrates a possible implementation of a listener class.

**Example 7-3 Sample Listener Class**

```

package com.oracle.handson;

import com.tangosol.net.NamedCache;

import com.tangosol.util.AbstractMapListener;
import com.tangosol.util.MapEvent;

import com.oracle.handson.Contact;

import com.tangosol.net.CacheFactory;

import java.io.IOException;

```

```

/**
 * ObserverExample observes changes to contacts.
 */
public class ObserverExample
{

    public ObserverExample()
    {
    }

    // ----- ObserverExample methods -----

    public static void main(String[] args) {
        NamedCache cache = CacheFactory.getCache("ContactsCache");
        new ObserverExample().observe(cache);
        try {
            System.in.read();
        } catch (IOException e) {
        }
    }

    /**
     * Observe changes to the contacts.
     *
     * @param cache target cache
     */
    public void observe(NamedCache cache)
    {
        cache.addMapListener(new ContactChangeListener());
    }

    // ----- inner class: ContactChangeListener -----

    public class ContactChangeListener
        extends AbstractMapListener
    {
        // ----- MapListener interface -----

        public void entryInserted(MapEvent event)
        {
            System.out.println(event);
        }

        public void entryUpdated(MapEvent event)
        {
            Contact contactOld = (Contact)event.getOldValue();
            Contact contactNew = (Contact)event.getNewValue();
            StringBuffer sb = new StringBuffer();

            if (!contactOld.getHomeAddress().equals(
                contactNew.getHomeAddress()))
            {
                sb.append("Home address ");
            }

            if (!contactOld.getWorkAddress().equals(
                contactNew.getWorkAddress()))
            {
                sb.append("Work address ");
            }

            if (!contactOld.getTelephoneNumbers().equals(
                contactNew.getTelephoneNumbers()))

```

```

        {
            sb.append("Telephone ");
        }
        if (contactOld.getAge() != contactNew.getAge())
        {
            sb.append("Birthdate ");
        }
        sb.append("was updated for ").append(event.getKey());
        System.out.println(sb);
    }

    public void entryDeleted(MapEvent event)
    {
        System.out.println(event.getKey());
    }
}
}

```

## Run the Cache Listener Example

To run the Cache Listener example:

1. Create a run configuration for `ObserverExample`. Right click `ObserverExample` in the **Project Explorer** and select **Run As**. In the **Run Configurations** dialog box select **Oracle Coherence** and click the **New Configuration** icon.
  - a. In the **Name** field, enter `ObserverExample`.
  - b. In the **Project** field in the **Main** tab, enter `Loading`. In the **Main class** field, enter `com.oracle.handson.ObserverExample`.
  - c. In the **General** tab of the **Coherence** tab, browse to the `c:\home\oracle\workspace\Contacts\appClientModule\contacts-cache-config.xml` file in the **Cache configuration descriptor** field. Select the **Disabled (cache client)** button. Enter `3155` in the **Cluster port** field. Click **Apply**.  
In the **Other** tab, scroll down to the `tangosol.pof.config` field. Enter the absolute path to the POF configuration file `contacts-pof-config.xml`. Click **Apply**.
  - d. In the **Common** tab, select **Shared file** and browse for the **Loading** directory.
2. Stop any running cache servers. See "[Stopping Cache Servers](#)" on page 2-14 for more information.
3. Start the `ContactsCacheServer`.
4. Load the cache by running the `LoaderExample` program from Eclipse. If you now run the `ObserverExample`, the program waits for input, as illustrated in [Example 7-4](#).

In "[Responding to Changes in the Cache](#)" on page 7-5, you create a program that modifies entries in the cache and returns the changed records.

### Example 7-4 Listener Program Waiting for Events

```

...
MasterMemberSet (
    ThisMember=Member(Id=3, Timestamp=2013-11-21 11:29:45.272, Address=130.35.99.6:8090,
MachineId=47251, Location=site:,machine:TPFAEFL-LAP,process:4752,
Role=OracleHandsonObserverExample)

```

```

    OldestMember=Member(Id=1, Timestamp=2013-11-21 11:29:02.843, Address=130.35.99.6:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:6736, Role=CoherenceServer)
    ActualMemberSet=MemberSet(Size=2
        Member(Id=1, Timestamp=2013-11-21 11:29:02.843, Address=130.35.99.6:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:6736, Role=CoherenceServer)
        Member(Id=3, Timestamp=2013-11-21 11:29:45.272, Address=130.35.99.6:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:4752, Role=OracleHandsonObserverExample)
    )
    MemberId|ServiceVersion|ServiceJoined|MemberState
        1|12.1.3|2013-11-21 11:29:02.843|JOINED,
        3|12.1.3|2013-11-21 11:29:45.272|JOINED
    RecycleMillis=1200000
    RecycleSet=MemberSet(Size=1
        Member(Id=2, Timestamp=2013-11-21 11:29:23.048, Address=130.35.99.6:8090, MachineId=47251)
    )
)

TcpRing{Connections=[1]}
IpMonitor{Addresses=0}

```

```

2013-11-21 11:29:45.713/4.273 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=3): Service Management joined the cluster with senior service member 1
2013-11-21 11:29:45.734/4.304 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=3): Loaded
Reporter configuration from "jar:file:/C:/Oracle/coherence/lib/coherence.jar!/reports/report-group.
xml"
2013-11-21 11:29:46.071/4.631 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=3): TcpAcceptor now listening for connections on 130.35.99.
6:8090.3
2013-11-21 11:29:46.395/4.955 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=DistributedCache:PartitionedPofCache, member=3): Loaded POF configuration from
"file:/C:/home/oracle/workspace/Contacts/appClientModule/contacts-pof-config.xml"
2013-11-21 11:29:46.422/4.982 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=DistributedCache:PartitionedPofCache, member=3): Loaded included POF configuration from
"jar:file:/C:/Oracle/coherence/lib/coherence.jar!/coherence-pof-config.xml"
2013-11-21 11:29:46.459/5.019 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=3): Service PartitionedPofCache joined the
cluster with senior service member 1

```

## Responding to Changes in the Cache

In this section, you will create a Java class to modify entries in the cache and return the changed records.

Until now, to perform actions on the entries in a cache, you used the `put` and `get` operations. However, there is a better way to perform operations on data that ensure consistent behavior when concurrent data access is required. Entry processors (`com.tangosol.util.InvokeableMap.EntryProcessor`) are agents that perform processing against entries. The entries are processed directly where the data is being held. The processing you perform can change the data: it can create, update, remove data, or only perform calculations. The processing can occur in parallel in a partitioned cache with multiple nodes, so it is scalable. Processing in the cache also saves I/O expense because data is not pulled to the client for processing.

Entry processors that work on the same key are logically queued. This allows lock-free (high performance) processing. The `com.tangosol.util.InvokeableMap` interface (which the `NamedCache` implements) has the following methods for operating on data:

- `Object invoke(Object oKey, InvocableMap.EntryProcessor processor)`, which invokes the passed `EntryProcessor` against an individual object and returns the result of the invocation.
- `Map invokeAll(Collection keys, InvocableMap.EntryProcessor processor)`, which invokes the `EntryProcessor` against the collection of keys and returns the result for each invocation.
- `Map invokeAll(Filter filter, InvocableMap.EntryProcessor processor)`, which invokes the `EntryProcessor` against the entries that match the filter and returns the result for each invocation.

---

**Note:** `EntryProcessor` classes must be available in the class path for each cluster node.

---

To create an entry process, you can extend `com.tangosol.util.processes.AbstractProcessor` and implement the `process()` method. For example, the following code creates an `EntryProcessor` instance to change the work address of employees in the `Contacts` data set:

```
public static class OfficeUpdater extends AbstractProcessor
    implements PortableObject
{
    ...
    public Object process(InvocableMap.Entry entry)
    {
        Contact contact = (Contact) entry.getValue();
        contact.setWorkAddress(m_addrWork);
        entry.setValue(contact);
        return null;
    }
}
```

To invoke the `OfficeUpdater` class, you can use the `invokeAll` method with the name of the `OfficeUpdater` class as one of its arguments.

```
cache.invokeAll(new EqualsFilter("getHomeAddress.getState", "MA"),
    new OfficeUpdater(addrWork));
```

In this exercise, you create a Java class with `EntryProcessor` instances that update entries in the cache. The `ObserverExample` class created in ["Create a Class to Listen for Changes in the Cache"](#) on page 7-2 will detect these changes and display the changed records.

1. [Create a Class to Update Entries in the Cache](#)
2. [Edit the POF Configuration File](#)
3. [Run the Cache Update Example](#)

## Create a Class to Update Entries in the Cache

To create a file to update entries in the cache:

1. Create a class that updates entries in the cache.
 

In the `Loading` project, create a class called `ProcessorExample` with a `main` method that updates the address of a `Contact` object in the cache. See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Write code to find the records of the `Contacts` object that live in Massachusetts and update their work addresses to an in-state office.

Include an inner class that implements the `PortableObject` interface (for serializing and deserializing data from the cache) and contains an `EntryProcessor` instance to set the work addresses. Use methods from the `Filter` class to isolate the `Contacts` members whose home addresses are in Massachusetts.

[Example 7-5](#) illustrates a possible implementation of the `ProcessorExample` class.

**Example 7-5 Sample Program to Update an Object in the Cache**

```
package com.oracle.handson;

import com.tangosol.net.NamedCache;

import com.tangosol.util.filter.EqualsFilter;
import com.tangosol.util.processor.AbstractProcessor;
import com.tangosol.util.InvocableMap;

import com.tangosol.io.pof.PortableObject;
import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PofWriter;

import com.oracle.handson.Address;
import com.oracle.handson.Contact;

import com.tangosol.net.CacheFactory;

import java.io.IOException;

/**
 * ProcessorExample executes an example EntryProcessor.
 */
public class ProcessorExample
{
    public ProcessorExample()
    {
    }

    public static void main(String[] args)
    {
        NamedCache cache = CacheFactory.getCache("ContactsCache");
        new ProcessorExample().execute(cache);
    }

    // ----- ProcessorExample methods -----

    public void execute(NamedCache cache)
    {
        // People who live in Massachusetts moved to an in-state office
        Address addrWork = new Address("200 Newbury St.", "Yoyodyne, Ltd.",
            "Boston", "MA", "02116", "US");

        cache.invokeAll(new EqualsFilter("getHomeAddress.getState", "MA"),
            new OfficeUpdater(addrWork));
    }

    // ----- nested class: OfficeUpdater -----

    /**
     * OfficeUpdater updates a contact's office address.
     */
}
```

```

public static class OfficeUpdater
    extends AbstractProcessor
    implements PortableObject
{
    // ----- constructors -----

    /**
     * Default constructor (necessary for PortableObject implementation).
     */
    public OfficeUpdater()
    {
    }

    public OfficeUpdater(Address addrWork)
    {
        m_addrWork = addrWork;
    }

    // ----- InvocableMap.EntryProcessor interface -----

    public Object process(InvocableMap.Entry entry)
    {
        Contact contact = (Contact) entry.getValue();

        contact.setWorkAddress(m_addrWork);
        entry.setValue(contact);
        System.out.println("Work address was updated for " + contact.
getFirstName() + " " + contact.getLastName());
        return null;
    }

    // ----- PortableObject interface -----

    public void readExternal(PofReader reader)
        throws IOException
    {
        m_addrWork = (Address) reader.readObject(0);
    }

    public void writeExternal(PofWriter writer)
        throws IOException
    {
        writer.writeObject(0, m_addrWork);
    }

    // ----- data members -----

    private Address m_addrWork;
}
}

```

## Edit the POF Configuration File

Edit the `contacts-pof-config.xml` file to add a user type ID for the `OfficeUpdater` entries. In this case, add the type ID 106 for the `ProcessorExample$OfficeUpdater` class.

```

...
<user-type>
    <type-id>106</type-id>

```

```

        <class-name>com.oracle.handson.
        ProcessorExample$OfficeUpdater</class-name>
    </user-type>
    ...

```

## Run the Cache Update Example

To run the cache update example.

1. Create a run configuration for `ProcessorExample`. Right click `ProcessorExample` in the **Project Explorer** and select **Run As**. In the **Run Configurations** dialog box select **Oracle Coherence** and click the **New Configuration** icon
  - In the **Name** field, enter `ProcessorExample`.
  - In the **Project** field in the **Main** tab, enter `Loading`. In the **Main class** field, enter `com.oracle.handson.ProcessorExample`.
  - In the **General** tab of the **Coherence** tab, browse to the `c:\home\oracle\workspace\Contacts\appClientModule\contacts-cache-config.xml` file in the **Cache configuration descriptor** field. Select the **Disabled (cache client)** button. Enter `3155` in the **Cluster port** field. Click **Apply**.  
 In the **Other** tab, scroll down to the **tangosol.pof.config** field. Enter the absolute path to the POF configuration file `contacts-pof-config.xml`. Click **Apply**.
  - In the **Common** tab, select **Shared file** and browse for the **Loading** directory.
2. Perform the following steps to test the `ObserverExample` and `ProcessorExample` classes.
  - a. Stop any running cache servers. See "[Stopping Cache Servers](#)" on page 2-14 for more information.
  - b. Restart the `ContactsCacheServer`.
  - c. Run the `LoaderExample` class to load the cache.
  - d. Run the `ObserverExample` class.
  - e. Run the `ProcessorExample` to update records in the cache.

You should see messages in the cache server console window, that are similar to [Example 7-6](#), indicating that the work addresses for the specified employees were updated.

### Example 7-6 Output from the `ObserverExample` and `ProcessorExample` Classes

```

...
Started DefaultCacheServer...
...
2013-11-21 15:44:33.425/69.017 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): Member 4 joined Service
PartitionedPofCache with senior member 1
Work address was updated for John Lfyovf
Work address was updated for John Jptrpajked
Work address was updated for John Mtqdlm
Work address was updated for John Mstaugiw
Work address was updated for John Olfezqse
Work address was updated for John Qjefjgtgj
Work address was updated for John Kuhgkzn
Work address was updated for John Jpby

```

```
Work address was updated for John Cekuea
Work address was updated for John Guhkam
Work address was updated for John Ijwj
Work address was updated for John Trlb
Work address was updated for John Hnfcwxjq
Work address was updated for John Kizifh
Work address was updated for John Rqlhgboi
Work address was updated for John Ipphab
2013-11-21 15:44:33.547/69.139 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
TcpRing disconnected from Member(Id=4, Timestamp=2013-11-21 15:44:32.073, Address=130.35.99.
28:8092, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7872,
Role=OracleHandsonProcessorExample) due to a peer departure; removing the member.
...
```

---

---

# Using JPA with Coherence

In this exercise, you learn how to use Java Persistence API (JPA) to perform object-relational mapping. It highlights the use of the Coherence memory-based backing map and Oracle TopLink Grid.

This chapter contains the following sections:

- [Introduction](#)
- [Mapping Relational Data to Java Objects with JPA](#)

You must have a working version of the Oracle Database 12c installed on your system. If you do not have the database, you can download it from this URL:

<http://www.oracle.com/technetwork/database/enterprise-edition/overview/index.html>

## Introduction

JPA defines how relational data is mapped to Java objects (persistent entities), the way that these objects are stored in a relational database so that they can be accessed at a later time, and the continued existence of an entity's state even after the application that uses it ends. In addition to simplifying the entity persistence model, the JPA standardizes object-relational mapping.

To determine how data is stored within a Coherence cluster, a backing map is used. By default, Coherence uses a memory-based backing map. To persist data, there are several backing map implementations.

The JPA implementation used in this exercise provides Object Relational Mapping (ORM) from the Java world to the database world. It also allows you to use standard Coherence `get` and `put` methods, and have Coherence calls translated into database calls using JPA and Oracle TopLink (based on the open source EclipseLink project).

### What is Oracle TopLink?

Oracle TopLink is a persistence framework including development tools and run-time capabilities for providing transformation, mapping, and persistence operations in Java SE and Java EE environments. The core technology of TopLink is provided by EclipseLink, the open-source persistence framework from the Eclipse Foundation.

TopLink includes additional features beyond EclipseLink. One of the additional components provided by TopLink is TopLink Grid.

### What is Oracle TopLink Grid?

Oracle TopLink Grid is a feature used for integrating the TopLink JPA implementation, provided by EclipseLink, with Oracle Coherence.

You can use the Coherence API with caches backed by TopLink Grid to access relational data with special cache loader and cache store interfaces which have been implemented for JPA. In this traditional Coherence approach, TopLink Grid provides the `CacheLoader` and `CacheStore` implementations in the `oracle.eclipselink.coherence.standalone` package that are optimized for EclipseLink JPA.

You can build applications using JPA and transparently use the power of the data grid for improved scalability and performance. TopLink Grid provides the means for scaling JPA applications using Oracle Coherence. Applications can be scaled in a number of ways, ranging from using Coherence as a distributed shared (L2) cache up to directing JP QL queries to Coherence for parallel execution across the grid to reduce database load. In this *JPA on the Grid* approach, TopLink Grid provides a set of cache and query configuration options that allow you to control how EclipseLink JPA uses Coherence. These implementations reside in the `oracle.eclipselink.coherence.integrated` package.

The TopLink Grid cache store and cache loader implementations are shipped in the `toplink-grid.jar` file. TopLink Grid uses the standard JPA run-time configuration file `persistence.xml` and the JPA mapping file `orm.xml`. The Coherence cache configuration file `coherence-cache-config.xml` must be specified to override the default Coherence settings and to define the cache store caching scheme.

This project uses the TopLink Grid cache store and cache loader classes which are optimized for EclipseLink JPA and designed for use by Coherence applications. These classes can be found in the `oracle.eclipselink.coherence.standalone` package. For more information on TopLink Grid cache store and cache loader classes, see "Using Coherence Caches Backed by TopLink Grid" in *Oracle Fusion Middleware Integrating Oracle Coherence*.

The *JPA on the Grid* approach is not illustrated in this project. For information on this technique, see "Integrating TopLink Grid with Oracle Coherence" in *Oracle Fusion Middleware Integrating Oracle Coherence*.

## Mapping Relational Data to Java Objects with JPA

In this exercise, you use Eclipse to perform the following tasks:

1. [Unlock the Oracle Database](#)
2. [Configure the Project for JPA](#)
3. [Create the JPA Persistence Unit and Entities](#)
4. [Edit the persistence.xml File](#)
5. [Create the Cache Configuration File for JPA](#)
6. [Create a Cache Server Start-Up Configuration](#)
7. [Create a Class to Interact with the Data Object](#)
8. [Create a Run Configuration for RunEmployeeExample](#)
9. [Run the JPA Example](#)

### Unlock the Oracle Database

This chapter assumes that you have installed the Oracle 12c database and that the account associated with the HR user name was unlocked and assigned the password `hr` during the installation process.

If you did not unlock the account associated with the HR user name during installation, follow these steps:

1. Navigate to **Start**, then **All Programs**, then Oracle database home (such as **Oracle-OraDB12Home1**), then **Application Development**, then **SQL Plus** to open the SQL command line tool.

2. Login to the database as the `system` user, and enter the password, such as `Welcome1`. Connect to the database as `sysdba`:

```
connect system as sysdba
```

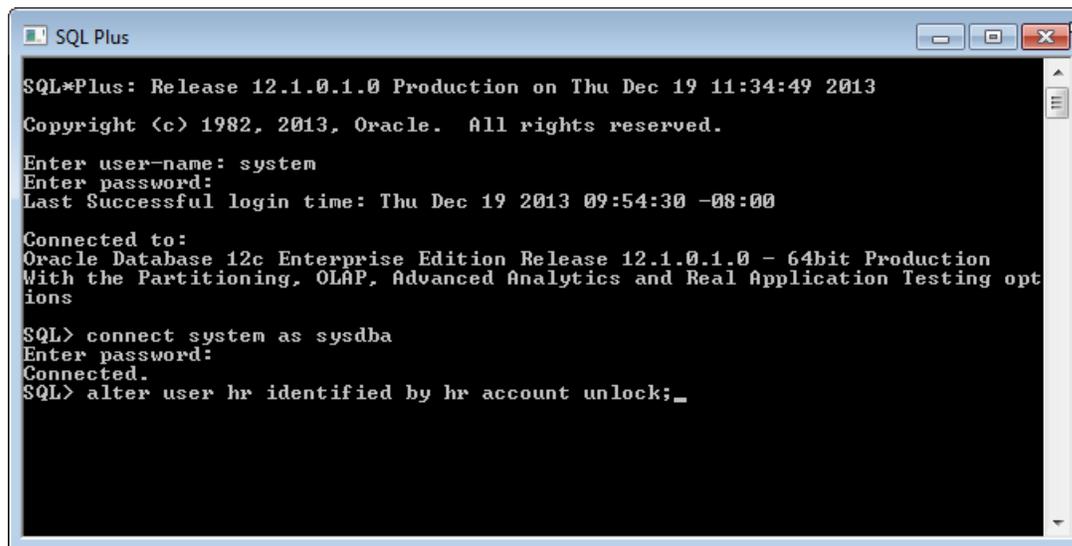
and then enter the `sysdba` password, such as `Welcome1`. (Note: your user name is `system` and your password is `Welcome1`.)

3. Enter the command to unlock the account associated with user `HR`. The password associated with the `HR` user is also `hr`.

```
alter user hr identified by hr account unlock;
```

These commands are illustrated in [Figure 8-1](#).

**Figure 8-1** Connecting to the Database



```

SQL*Plus: Release 12.1.0.1.0 Production on Thu Dec 19 11:34:49 2013
Copyright (c) 1982, 2013, Oracle. All rights reserved.

Enter user-name: system
Enter password:
Last Successful login time: Thu Dec 19 2013 09:54:30 -08:00

Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing opt
ions

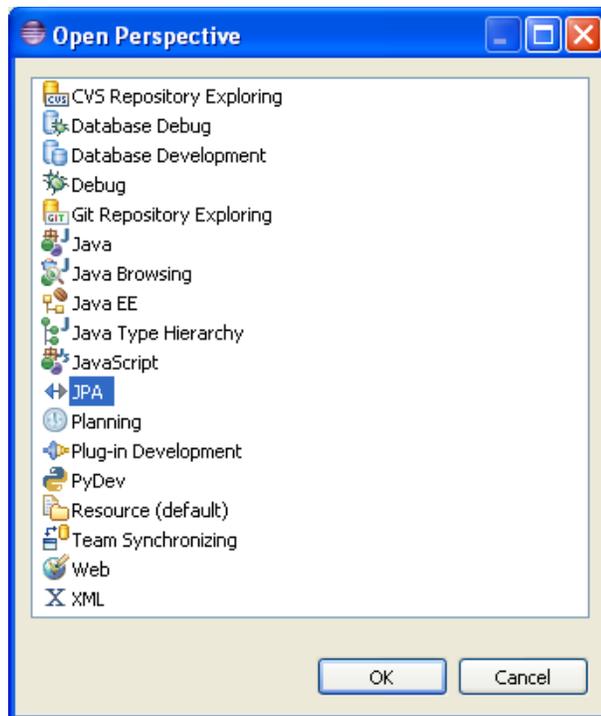
SQL> connect system as sysdba
Enter password:
Connected.
SQL> alter user hr identified by hr account unlock;_

```

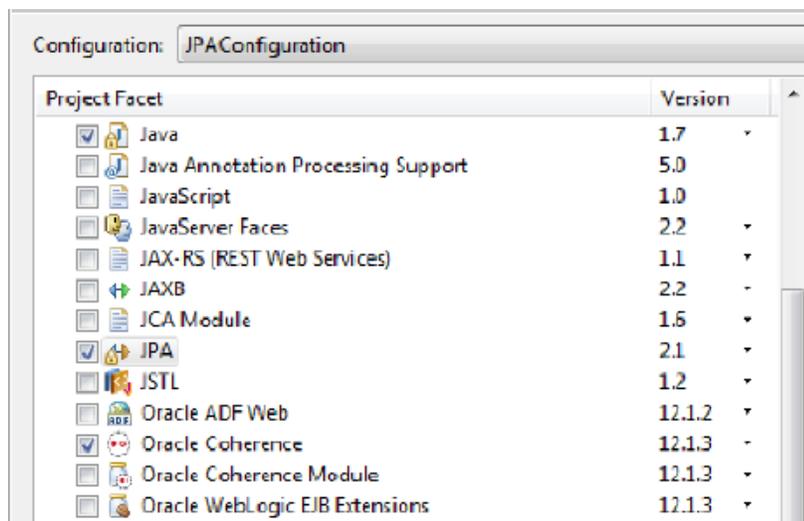
## Configure the Project for JPA

Create a new project in Eclipse IDE called `JPAProject`. See ["Creating a New Project in the Eclipse IDE"](#) on page 2-3 for detailed information.

1. In the Eclipse IDE, click **Window** then **Open Perspective** then **Other** to open the **Open Perspective** dialog box. Select the **JPA** perspective, as illustrated in [Figure 8-2](#).

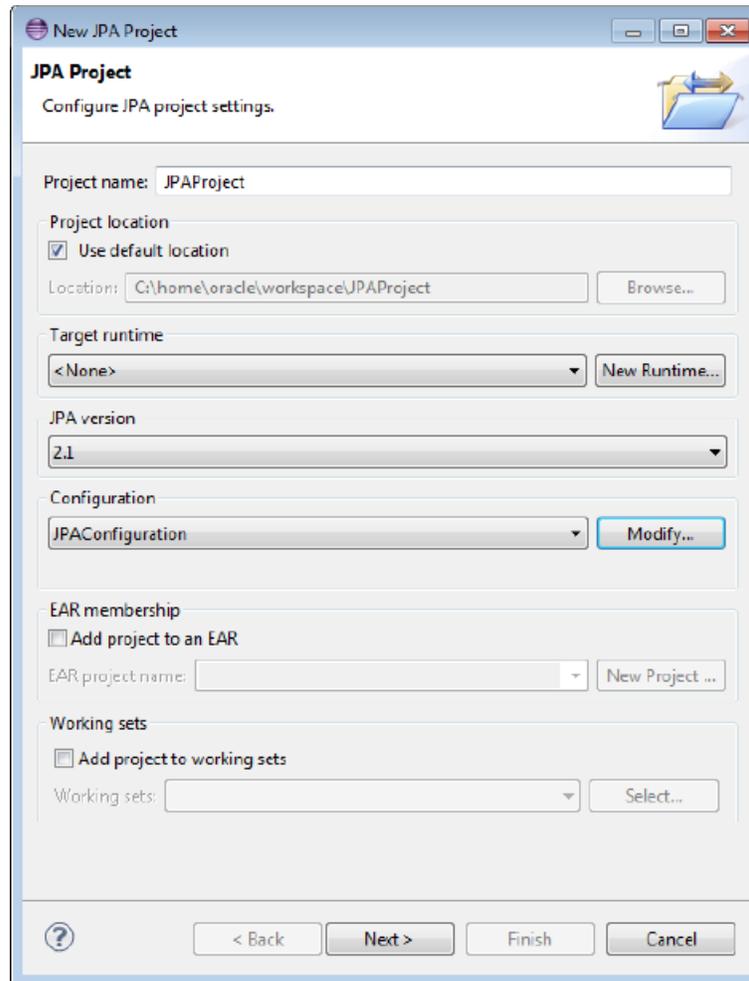
**Figure 8–2** Selecting the JPA Perspective in the Open Perspective Dialog Box

2. Select **File** then **New** then **JPA Project**. Enter `JPAProject` as the project name. Ensure that the default location points to `home\oracle\workspace\JPAProject`. Click **Modify** in the **Configuration** field to open the **Project Facets** dialog box.
3. Select the **Oracle Coherence** facet. The **JPA** and **Java** facets should already be selected.
4. Click **Save As** in the **Configuration** field. Enter `JPAConfiguration` in the **Save Preset** dialog box and click **OK**. The contents of the **Project Facets** dialog box should look similar to [Figure 8–3](#).

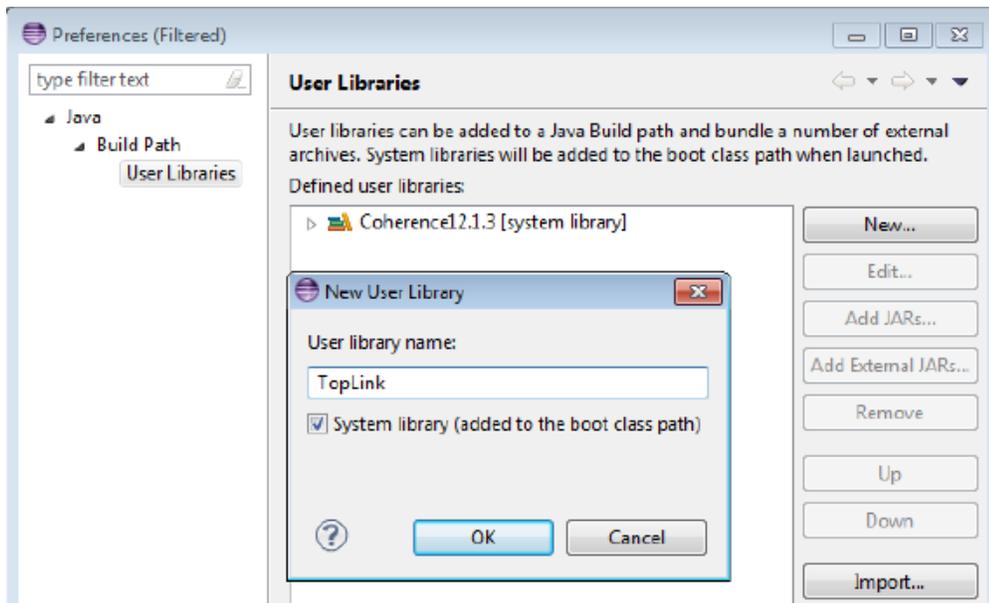
**Figure 8–3** Project Facets for a JPA Project

The contents of the **New JPA Project** dialog box should look similar to [Figure 8–4](#). Click **Next** to go to the **Java** page.

**Figure 8–4** Contents of the **New JPA Project** Dialog Box

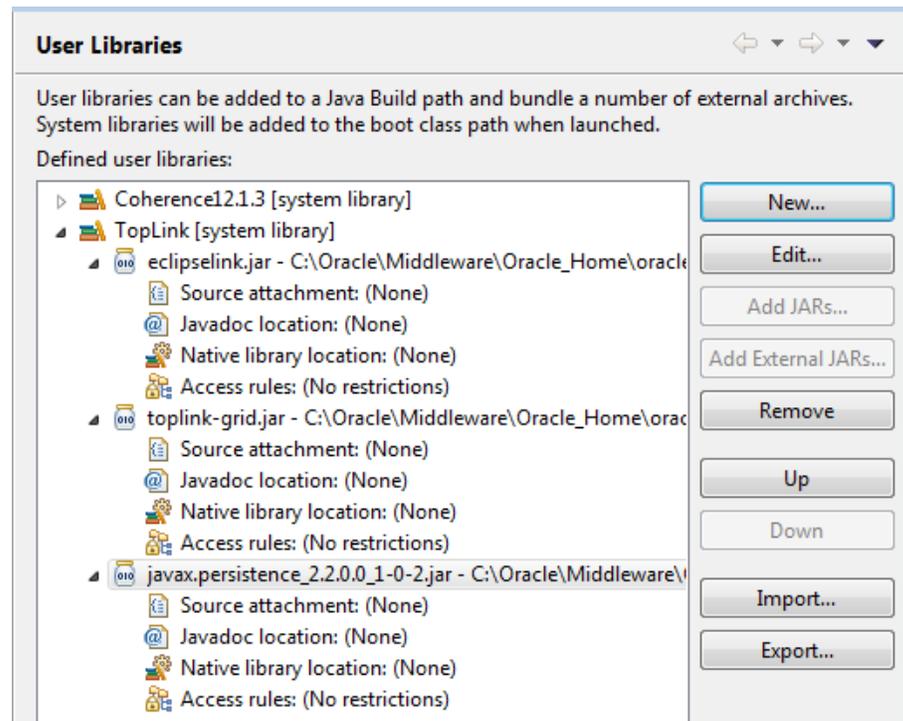


5. Click **Next** in the **Java** page to accept the default output folder location.
6. In the **JPA Facet** page, select **EclipseLink 2.5.x** in the **Platform** drop-down list. Click the **Manage Libraries** icon to add the **TopLink Grid**, **EclipseLink**, and **Java Persistence** files.
7. Click **New** in the Preferences dialog box. In the **New User Library** dialog box, enter **TopLink**. Select the **System library** checkbox, as illustrated in [Figure 8–5](#).

**Figure 8–5 Adding the EclipseLink JPA Library**

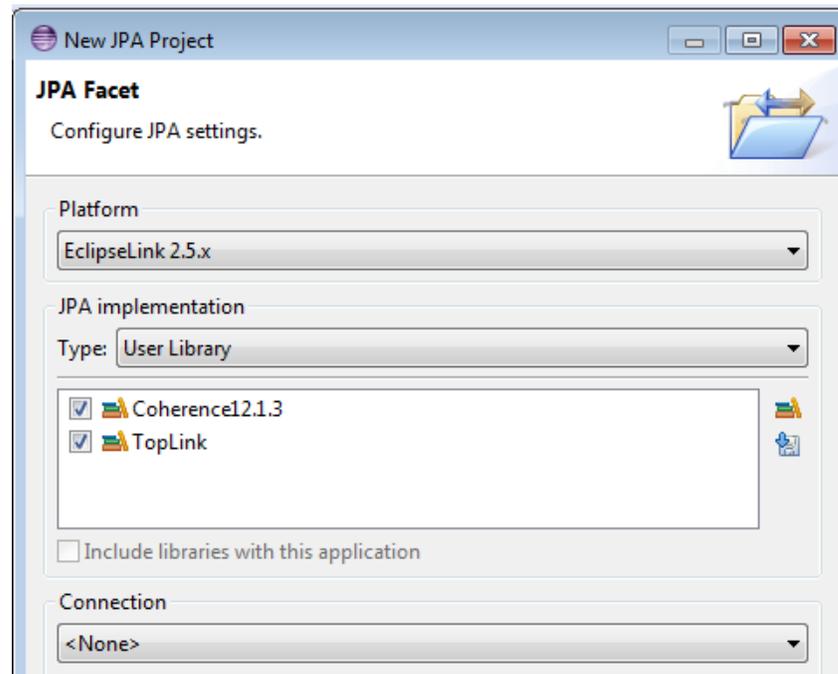
8. Click **OK** to dismiss the New User Library dialog box.
9. In the Preference dialog box, select **TopLink[system library]** and click **Add External JARS**.
10. Navigate to the **Oracle\_Home\oracle\_common\modules\oracle.toplink\_12.1.3** folder. Select the **toplink-grid.jar** and **eclipselink.jar** files and click **Open**. The **toplink-grid.jar** and **eclipselink.jar** files are added to the **TopLink** library. Click **Add External JARS** again and navigate to the **Oracle\_Home\oracle\_common\modules** folder. Select the **javax.persistence\_2.2.0.0\_1-0-2.jar** file and click **Open**. When you are finished, User Libraries should look similar to [Figure 8–6](#).

**Figure 8–6 Adding TopLink Grid, EclipseLink, and Java Persistence JAR Files to the TopLink User Library**



11. Click **OK** to dismiss the Preferences dialog box. Select the **Coherence12.1.3** and **TopLink** user libraries. The **JPA Facet** page should now look similar to [Figure 8–7](#).

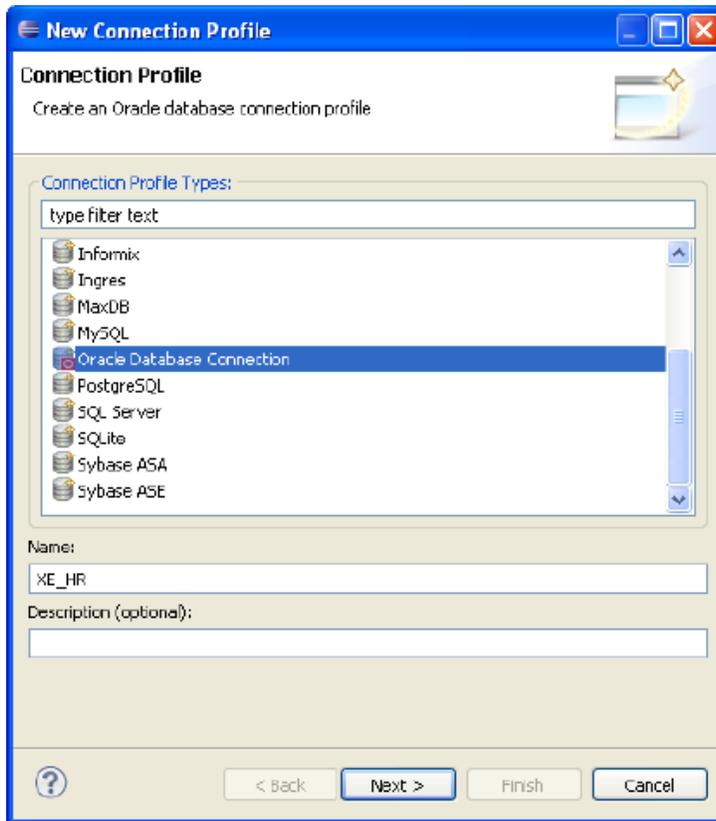
**Figure 8–7 TopLink Libraries Added to the JPA Facet Page**



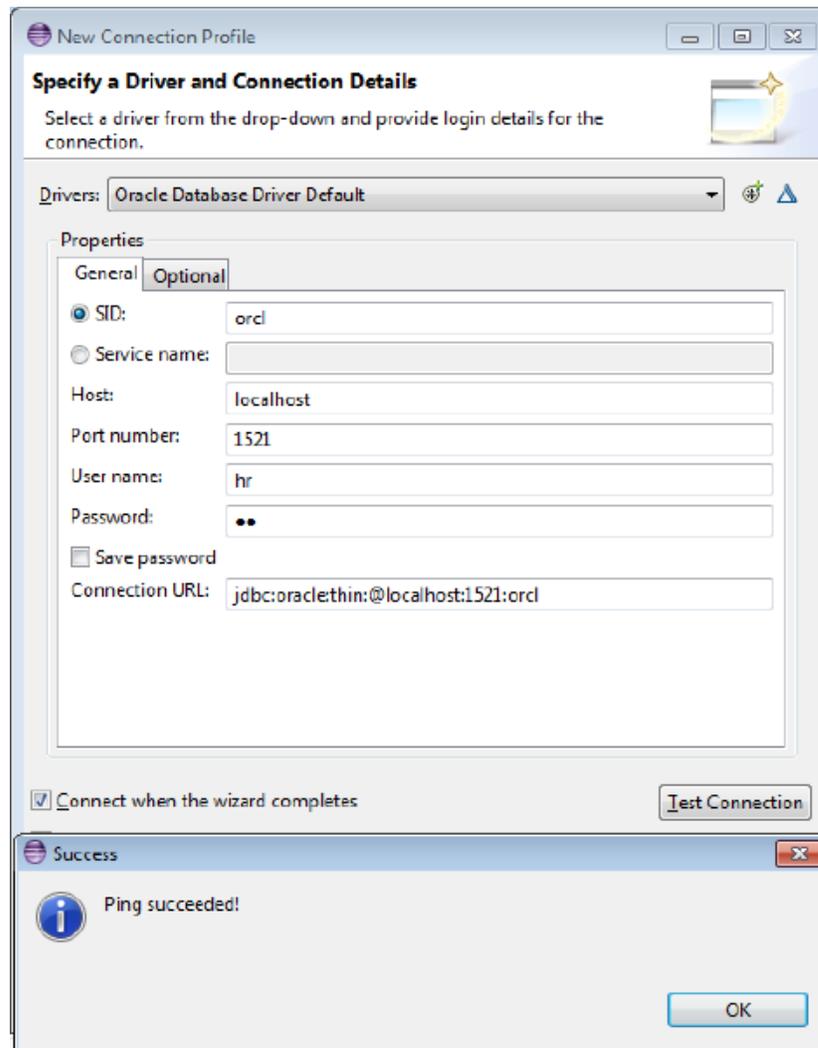
12. Create a connection to the Oracle Database. Click **Add connection**.

- a. In **Connection Profile** dialog box, select **Oracle Database Connection**, and enter `XE_HR` in the **Name** field, as illustrated in [Figure 8–8](#). Click **Next**.

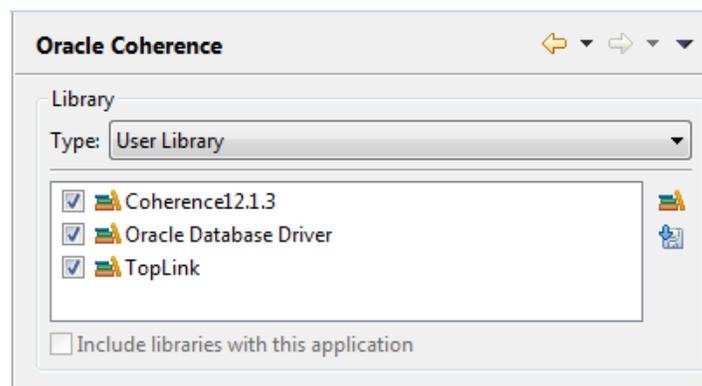
**Figure 8–8** *Connection Profile Page*



- b. Select **Oracle Database Driver Default** in the **Drivers** field. Edit the **General** tab. Enter `hr` in the **User name** field and `hr` in the **Password** field. Select the **Save Password** check box. Enter `localhost` in the **Host** field. Select the **Connect when the wizard completes** check box. Click **Test Connection** button. The test should return an alert box with a success message, as illustrated in [Figure 8–9](#). Click **OK**, then **Next** to see the Summary Page. Click **Finish**.

**Figure 8–9** The New Connection Profile Dialog Box and the Success Message

- c. In the **JPA Facet** page, click **Add driver library to build path**, then click **Next**.
13. In the **Coherence** page, ensure that the **Coherence 12.1.3**, **Oracle Database Driver**, and **TopLink** user libraries are present and selected. Click **Finish**.

**Figure 8–10** Coherence Page of the New JPA Project Wizard

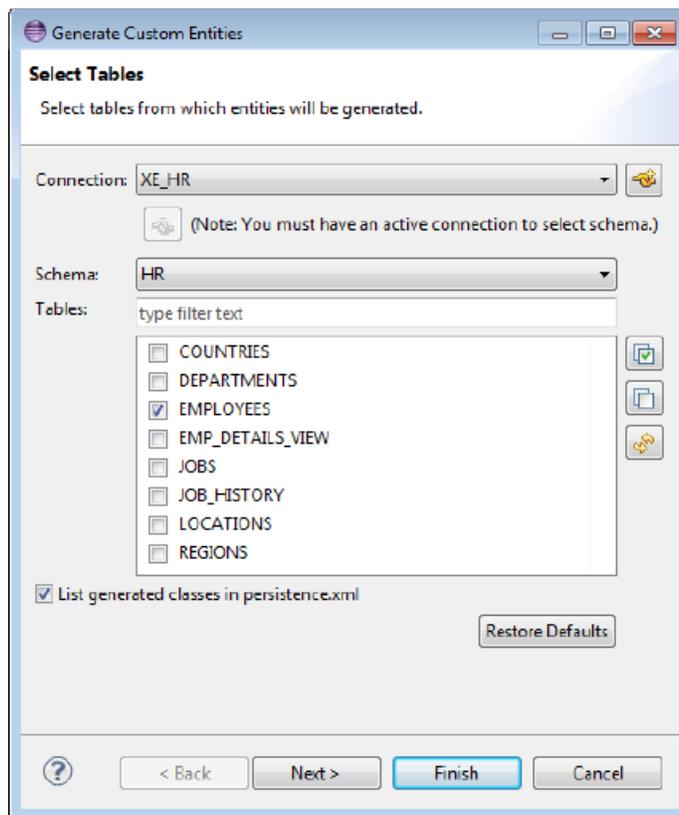
## Create the JPA Persistence Unit and Entities

A persistence unit provides an easy way to identify the set of metadata files, classes, and JARs that contain all classes that need to be persisted as a group. The name of the persistence unit is used to identify it. Entities contain persistent data that can be saved in persistent data stores. The JPA entity can belong to a persistence unit. The persistence unit is described by the `persistence.xml` file.

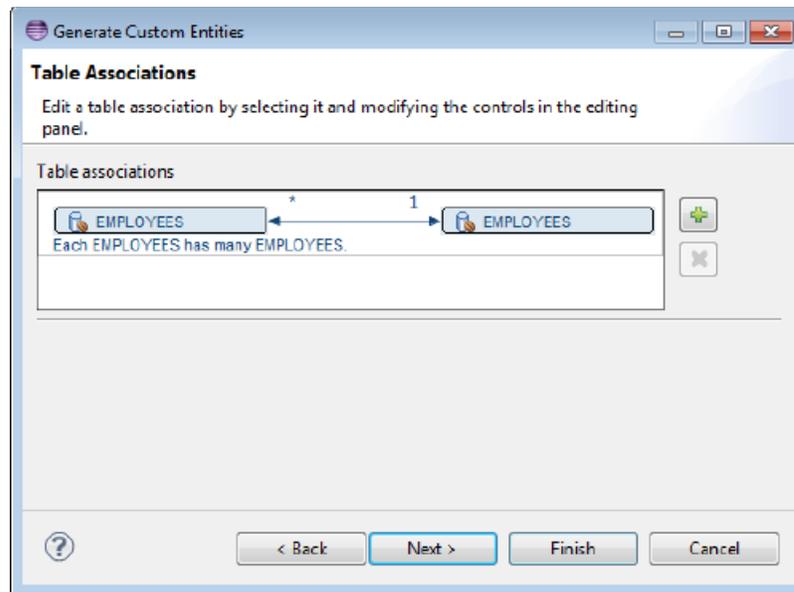
To create the JPA persistence unit and the JPA entities:

1. Right-click the `JPAProject` project and select **New** then select **JPA Entities from Tables**. The Select Tables dialog box opens.
2. Select the `EMPLOYEES` table, as illustrated in [Figure 8–11](#) and click **Next**.

**Figure 8–11** *Selecting the Database Tables*



3. In the **Table Associations** page, illustrated in [Figure 8–12](#), click **Next** to accept the defaults.

**Figure 8–12 Associations for the EMPLOYEES Table**

4. In the Customize Defaults page, select **Collection properties type** `java.util.List`, Enter `com.oracle.handson` in the **Package** field. Select **Lazy** in the **Associations fetch** field.

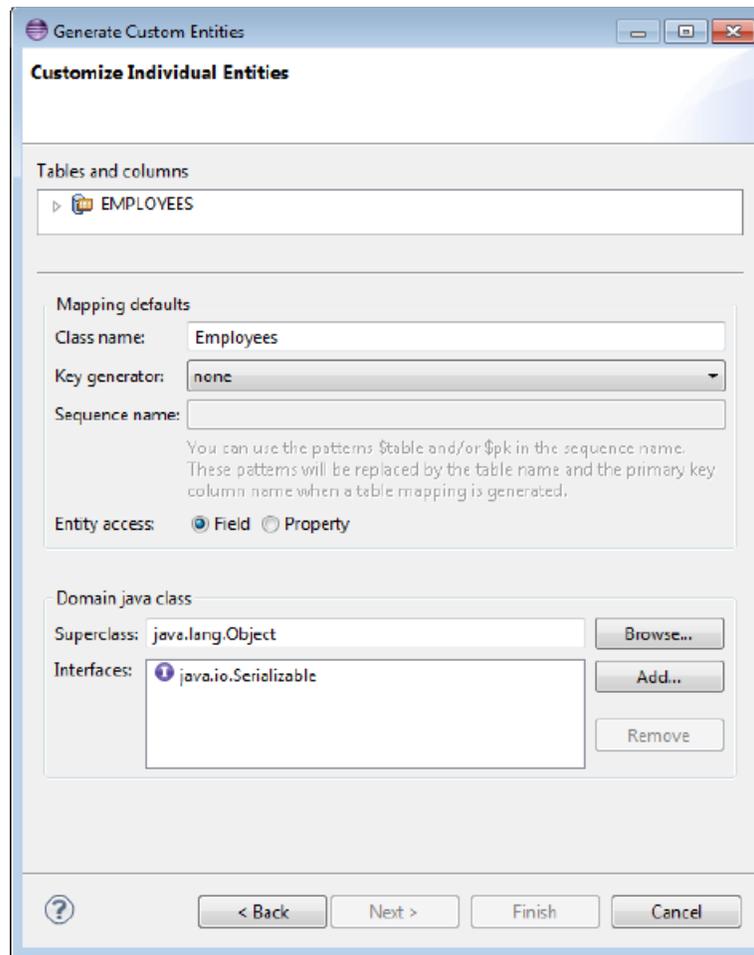
When you are finished, the **Customize Default Entity Generation** page should look similar to [Figure 8–13](#). Click **Next**.

**Figure 8–13 Customize Default Entity Generation Page**

The screenshot shows a dialog box titled "Generate Custom Entities" with a sub-tab "Customize Defaults". The main instruction reads: "Optionally customize aspects of entities that will be generated by default from database tables. A Java package should be specified." The "Mapping defaults" section includes a "Key generator" dropdown set to "none", an empty "Sequence name" field, and a note: "You can use the patterns \$table and/or \$pk in the sequence name. These patterns will be replaced by the table name and the primary key column name when a table mapping is generated." Below this are radio buttons for "Entity access" (Field selected, Property unselected) and "Associations: fetch" (Default unselected, Eager unselected, Lazy selected). The "Collection properties type" section has radio buttons for "java.util.Set" (unselected) and "java.util.List" (selected). A checkbox "Always generate optional JPA annotations and DDL parameters" is unchecked. The "Domain java class" section contains fields for "Source folder" (JPAProject/src), "Package" (com.oracle.handson), "Superclass" (empty), and "Interfaces" (java.io.Serializable). Each field has a corresponding "Browse..." or "Add..." button, and a "Remove" button is at the bottom right. At the bottom of the dialog are buttons for "< Back", "Next >", "Finish", and "Cancel".

5. In the **Customize Individual Entities** page, enter `java.lang.Object` in the **Superclass** field. Change the **Class name** from `Employee` to `Employees`, as illustrated in [Figure 8–14](#).

Figure 8–14 Customize Individual Entities Page

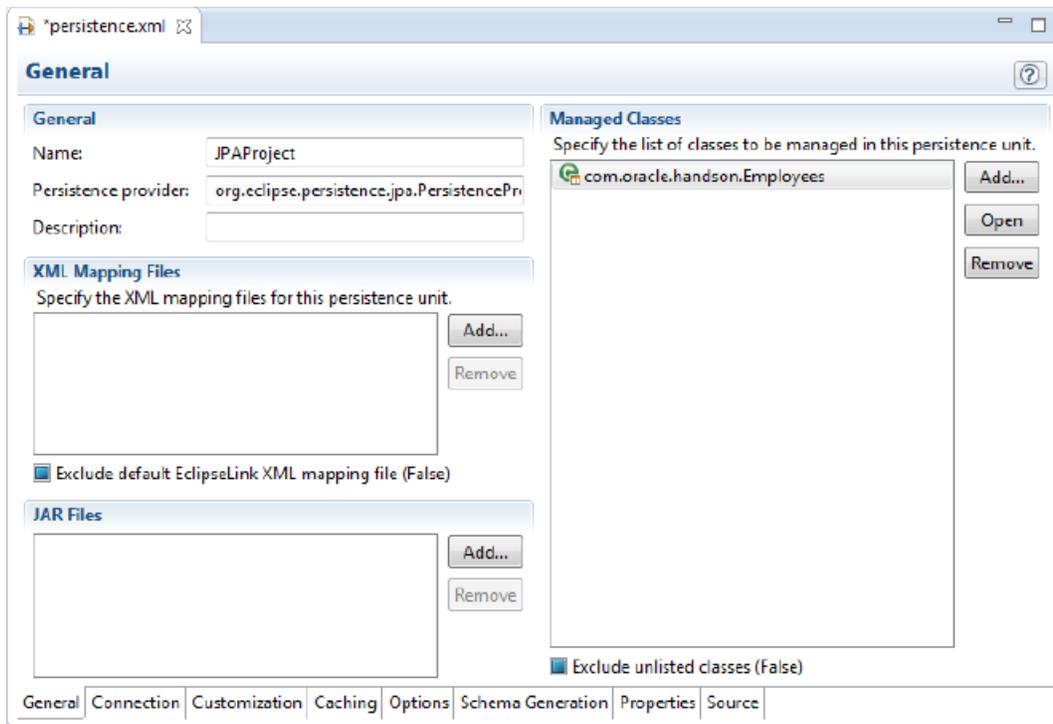


6. Click **Finish** to generate the JPA Entities. An `Employees.java` file is generated in the `JPAProject/src/com.oracle.handson` node in the Project Explorer.

## Edit the persistence.xml File

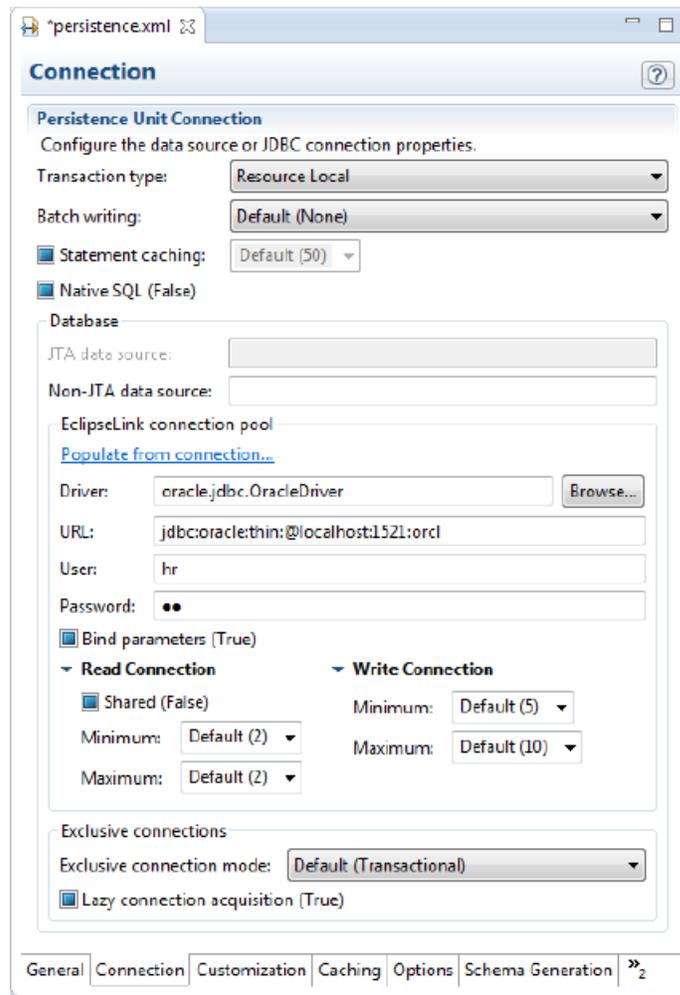
The `persistence.xml` file defines one or more persistence units. Use the **persistence.xml Editor** tool in the Eclipse IDE to edit the `persistence.xml` file.

1. Open the `persistence.xml` file in the Eclipse editor.
2. Click the **General** tab. Enter the name of the persistence provider if it is not already provided. It will be `org.eclipse.persistence.jpa.PersistenceProvider`.

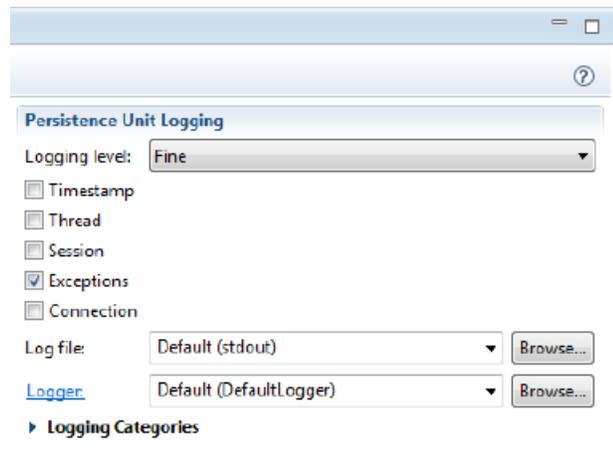
**Figure 8–15** General Tab in the persistence.xml Editor

3. In the **Connection** tab, select **Resource Local** from the **Transaction type** drop down list.

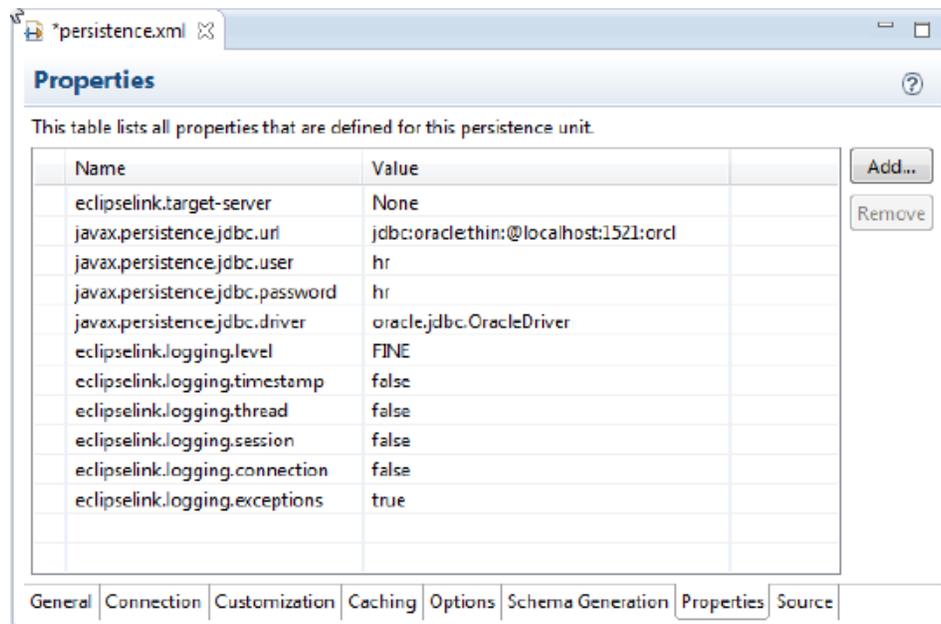
In **EclipseLink connection pool** section, click **Populate from connection** link, then click the XE\_HR item in the Connection Selection dialog box. The values in this field will be provided from the database connection you defined in "[Configure the Project for JPA](#)" on page 8-3. The Connection Properties tab should look similar to [Figure 8–16](#).

**Figure 8–16 Connection Properties Tab in the persistence.xml Editor**

4. Open the **Options** tab and select **Fine** from the **Logging level** drop down list. Select the **Exceptions** checkbox. Deselect the **Timestamp**, **Thread**, **Session**, and **Connection** checkboxes. When you are finished, the **Options** tab should look similar to [Figure 8–17](#).

**Figure 8–17** Logging Tab in the persistence.xml Editor

5. Inspect the **Properties** tab. The contents should look similar to [Figure 8–18](#).

**Figure 8–18** Properties Tab in the persistence.xml Editor

6. Open the **Source** tab. The persistence.xml file should look similar to [Example 8–1](#). Save the file.

**Example 8–1** Generated persistence.xml File

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.
org/xml/ns/persistence/persistence_2_1.xsd">
  <persistence-unit name="JPAProject" transaction-type="RESOURCE_LOCAL">
    <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
    <class>com.oracle.handson.Employees</class>
    <properties>
      <property name="eclipselink.target-server" value="None" />
    </properties>
  </persistence-unit>
</persistence>
```

```

        <property name="javax.persistence.jdbc.url"
value="jdbc:oracle:thin:@localhost:1521:orcl"/>
        <property name="javax.persistence.jdbc.user" value="hr"/>
        <property name="javax.persistence.jdbc.password" value="hr"/>
        <property name="javax.persistence.jdbc.driver" value="oracle.jdbc.
OracleDriver"/>
        <property name="eclipselink.logging.level" value="FINE"/>
        <property name="eclipselink.logging.timestamp" value="false"/>
        <property name="eclipselink.logging.thread" value="false"/>
        <property name="eclipselink.logging.session" value="false"/>
        <property name="eclipselink.logging.connection" value="false"/>
        <property name="eclipselink.logging.exceptions" value="true"/>
    </properties>
</persistence-unit>
</persistence>

```

## Create the Cache Configuration File for JPA

Create a cache configuration file for the JPA project.

1. Right click `coherence-cache-config.xml` file in the project and select **Open**.
2. Enter the code illustrated in [Example 8-2](#). Use **Rename** or **Save As** to save the file as `jpa-cache-config.xml`. The file will be saved to the `home\oracle\workspace\JPAProject\src` folder.
3. Ensure that the original `coherence-cache-config.xml` file no longer appears in **Project Explorer**.

In [Example 8-2](#), note the use of `distributed-eclipselink` as the value of the `scheme-name` element, `EclipseLinkJPA` as the value of the `service-name` element, and `oracle.eclipselink.coherence.standalone.EclipseLinkJPACacheStore` as the value of the `class-name` subelement of the `cachestore-scheme` element.

### Example 8-2 Cache Configuration for JPA

```

<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.oracle.com/coherence/cache-config"
xsi:schemaLocation="http://xmlns.oracle.
com/coherence/cache-config http://xmlns.oracle.
com/coherence/cache-config/1.2/coherence-cache-config.xsd">

    <defaults>
        <serializer system-property="tangosol.coherence.serializer"/>
        <socket-provider system-property="tangosol.coherence.socketprovider"/>
    </defaults>
    <caching-scheme-mapping>
        <cache-mapping>
            <!-- Set the name of the cache to be the entity name -->
            <cache-name>Employees</cache-name>
            <!-- Configure this cache to use the scheme defined below -->
            <scheme-name>distributed-eclipselink</scheme-name>
        </cache-mapping>
    </caching-scheme-mapping>
    <caching-schemes>
        <distributed-scheme>
            <scheme-name>distributed-eclipselink</scheme-name>
            <service-name>EclipseLinkJPA</service-name>

```

```

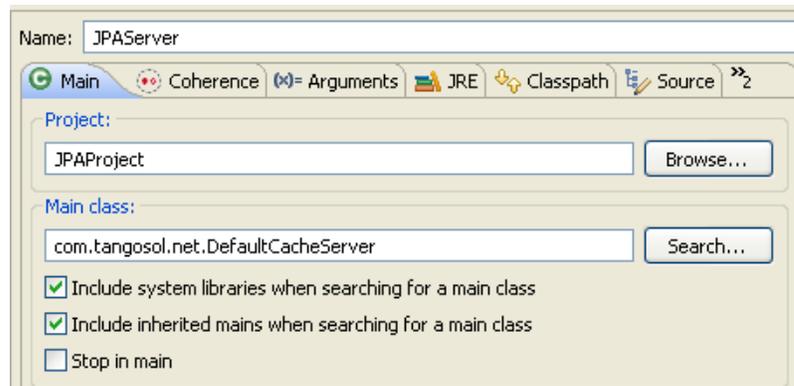
<backing-map-scheme>
  <read-write-backing-map-scheme>
    <!--
    Define the cache scheme
    -->
    <internal-cache-scheme>
      <local-scheme/>
    </internal-cache-scheme>
    <cachestore-scheme>
      <class-scheme>
        <class-name>oracle.eclipselink.coherence.standalone.
EclipseLinkJPACacheStore</class-name>
        <init-params>
          <!--
          This param is the entity name
          This param should match the value of the
          persistence unit name in persistence.xml
          -->
          <init-param>
            <param-type>java.lang.String</param-type>
            <param-value>{cache-name}</param-value>
          </init-param>
          <init-param>
            <param-type>java.lang.String</param-type>
            <param-value>JPAProject</param-value>
          </init-param>
        </init-params>
      </class-scheme>
    </cachestore-scheme>
  </read-write-backing-map-scheme>
</backing-map-scheme>
<autostart>true</autostart>
</distributed-scheme>
</caching-schemes>
</cache-config>

```

## Create a Cache Server Start-Up Configuration

Create a configuration to start the cache server for the JPA project.

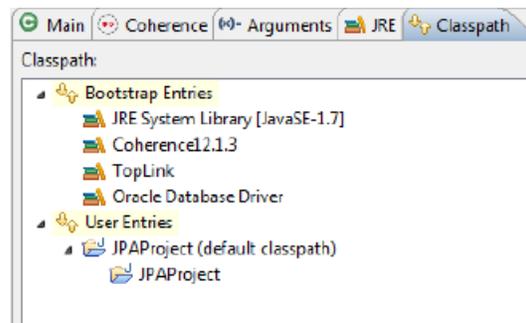
1. Right click the project and select **Run As** then **Run Configurations**. Double click the **Oracle Coherence** icon in the **Run Configurations** dialog box to create a new launch configuration.
2. In the **Main** tab, enter **JPAServer** in the **Name** field. Click **Browse** in the **Project** field and select the **JPAProject** project in the **Project Selection** dialog box. Select the **Include system libraries when searching for a main class** checkbox and click **Search**. Enter **DefaultCacheServer** in the **Select Type** field and select **com.tangosol.net.DefaultCacheServer**. Click **Apply**. The **Main** tab should look similar to [Figure 8–19](#).

**Figure 8–19 Main Tab for the JPA Cache Server**

3. In the **General** tab of the **Coherence** tab, identify the path to the cache configuration file under **Cache configuration descriptor**. Click the **Browse** button to navigate to the **Absolute file path** of the JPA cache configuration file `C:\home\oracle\workspace\JPAProject\src\jpa-cache-config.xml`. Select **Enabled (cache server)** under **Local storage**. Enter a unique value, such as 3155, for the **Cluster port**.

In the **Other** tab, ensure that the `tangosol.pof.config` item is set to the default `pof-config.xml`.

4. In the **Common** tab, select **Shared file** and browse to the `\JPAProject`.
5. The **Classpath** tab should look similar to [Figure 8–20](#).

**Figure 8–20 Classpath Tab for the JPA Cache Server Executable**

You will start the cache server in a later step: "[Run the JPA Example](#)" on page 8-21.

## Create a Class to Interact with the Data Object

Create a new class in the `JPAProject` project to interact with the `Employees` object. The objective of the class will be to change the value of an employee's salary.

1. Create a new class with a main method called `RunEmployeeExample`. See "[Creating a Java Class](#)" on page 2-11 for detailed information.
2. Create the code to perform the following:
  - a. Get an employee using the `EMPLOYEE_ID` attribute. `EMPLOYEE_ID` is a long data type.
  - b. Display the salary.

- c. Give the employee a 10% pay raise.
- d. Get the value again to confirm the pay raise.

[Example 8-3](#) illustrates a possible implementation of the `RunEmployeeExample` class.

### **Example 8-3 Sample Employee Class File**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import java.math.BigDecimal;

public class RunEmployeeExample
{
    public RunEmployeeExample()
    {
    }

    public static void main(String[] args)
    {
        long empId = 190L; // emp 190 - Timothy Gates

        NamedCache employees = CacheFactory.getCache("Employees");

        Employees emp = (Employees)employees.get(empId);

        System.out.println("Employee " + emp.getFirstName() + " " +
            emp.getLastName() + ", salary = $" + emp.getSalary() );

        // give them a 10% pay rise
        emp.setSalary(emp.getSalary().multiply(BigDecimal.valueOf(1.1)));

        employees.put(empId, emp);

        Employees emp2 = (Employees)employees.get(empId);

        System.out.println("New Employee details are " + emp2.getFirstName() + " "
            + emp2.getLastName() + ", salary = $" + emp2.getSalary() );
    }
}
```

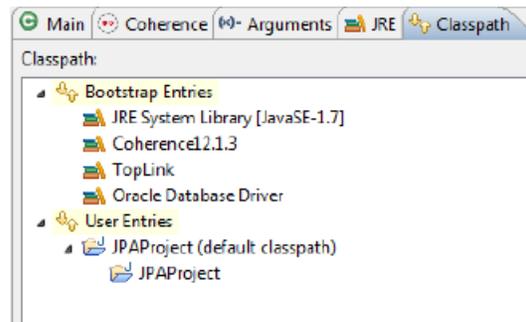
## **Create a Run Configuration for RunEmployeeExample**

To create a run configuration, modify the run-time properties and edit the class path in Eclipse.

1. Right click `RunEmployeeExample.java` in the **Project Explorer** and choose **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, double-click the **Oracle Coherence** icon to create a new launch configuration.
2. In the **Main** tab, enter `RunEmployee` in the **Name** field. Click **Browse** and select the `JPAProject` project. In the **Main class** field, click **Search** and choose `com.oracle.handson.RunEmployeeExample`. Click **Apply**.
3. In the **General** tab of the **Coherence** tab, browse to the `C:\home\oracle\workspace\JPAProject\src\jpa-cache-config.xml` file in the **Cache configuration descriptor** field. Select the **Disabled (cache client)** button. Enter 3155 in the **Cluster port** field. Click **Apply**.

4. In the **Other** tab, ensure that the default `pof-config.xml` appears in the `tangosol.pof.config` field.
5. In the **Common** tab, select **Shared file** and browse to the `\JPAProject` project. The **Classpath** tab should look similar to [Figure 8–21](#).

**Figure 8–21 Classpath Tab for the RunEmployee Executable**



## Run the JPA Example

Now that the `Employee` class has been annotated to persist to the database using JPA, and you have included the `persistence.xml` file to tell JPA where your database is, Coherence employs a `CacheStore` implementation that uses TopLink Grid to load and store objects in the database. When you use the `get(Object key)` method, the following happens:

- Coherence looks for the entry with the key.
- if the entry is not already in the cache, or if it is expired from the cache, then Coherence calls the backing map, which uses the TopLink Grid cache store, to retrieve the data.
- if the entry is in the cache, Coherence returns the entry directly to the application without going through the TopLink Grid cache store. When you use the `put(object key, object value)` method, Coherence uses EclipseLink JPA through TopLink Grid to persist any changes to the database.

In the **Run Configurations** dialog box, select the `JPAServer` launch configuration and click **Run** to start the cache server. After the cache server starts, select the `RunEmployee` configuration and click **Run**. The output from the executable should look similar to [Example 8–4](#).

### Example 8–4 Output from the RunEmployee Executable

```
...
2013-12-19 14:46:50.395/3.800 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=2): Service Management joined the cluster with senior service member 1
2013-12-19 14:46:50.415/3.820 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=2): Loaded
Reporter configuration from "jar:file:/C:/Oracle/coherence/lib/coherence.jar!/reports/report-group.
xml"
2013-12-19 14:46:50.745/4.150 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=2): TcpAcceptor now listening for connections on 130.35.99.
14:8090.3
2013-12-19 14:46:51.105/4.510 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:EclipseLinkJPA, member=2): Service EclipseLinkJPA joined the cluster with
senior service member 1
Employee Timothy Gates, salary = $2900
New Employee details are Timothy Gates, salary = $3190.0
```

The response from the cache server displays a successful login to retrieve information from the database, as illustrated in [Example 8-5](#). Note the logging information returned (in bold).

### Example 8-5 Cache Server Response to Logging In to the Database

```

...
Started DefaultCacheServer...

2013-12-19 14:46:50.143/42.045 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2013-12-19 14:46:49.953, Address=130.35.99.14:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3076, Role=OracleHandsonRunEmployeeExample) joined
Cluster with senior member 1
2013-12-19 14:46:50.395/42.297 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 joined Service Management with senior member 1
2013-12-19 14:46:51.125/43.027 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:EclipseLinkJPA, member=1): Member 2 joined Service EclipseLinkJPA with
senior member 1
[EL Config]: metadata: The access type for the persistent class [class com.oracle.handson.
Employees] is set to [FIELD].
[EL Config]: metadata: The target entity (reference) class for the many to one mapping element
[field employee] is being defaulted to: class com.oracle.handson.Employees.
[EL Config]: metadata: The target entity (reference) class for the one to many mapping element
[field employees] is being defaulted to: class com.oracle.handson.Employees.
[EL Config]: metadata: The alias name for the entity class [class com.oracle.handson.Employees] is
being defaulted to: Employees.
[EL Config]: metadata: The table name for entity [class com.oracle.handson.Employees] is being
defaulted to: EMPLOYEES.
[EL Config]: metadata: The column name for element [email] is being defaulted to: EMAIL.
[EL Config]: metadata: The column name for element [salary] is being defaulted to: SALARY.
[EL Config]: metadata: The primary key column name for the mapping element [field employee] is
being defaulted to: EMPLOYEE_ID.
[EL Info]: EclipseLink, version: Eclipse Persistence Services - 2.5.2.v20131130-72ace27
[EL Fine]: connection: Detected database platform: org.eclipse.persistence.platform.database.
oracle.Oracle11Platform
[EL Config]: connection: connecting(DatabaseLogin(
  platform=>Oracle11Platform
  user name=> "hr"
  datasource URL=> "jdbc:oracle:thin:@localhost:1521:orcl"
))
[EL Config]: connection: Connected: jdbc:oracle:thin:@localhost:1521:orcl
  User: HR
  Database: Oracle Version: Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit
Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options
  Driver: Oracle JDBC driver Version: 11.2.0.3.0
[EL Info]: connection: file:/C:/home/oracle/workspace/JPAProject/build/classes/_JPAProject login
successful
[EL Fine]: sql: SELECT EMPLOYEE_ID, COMMISSION_PCT, DEPARTMENT_ID, EMAIL, FIRST_NAME, HIRE_DATE,
JOB_ID, LAST_NAME, PHONE_NUMBER, SALARY, MANAGER_ID FROM EMPLOYEES WHERE (EMPLOYEE_ID = ?)
  bind => [190]
[EL Fine]: sql: SELECT EMPLOYEE_ID, COMMISSION_PCT, DEPARTMENT_ID, EMAIL, FIRST_NAME, HIRE_DATE,
JOB_ID, LAST_NAME, PHONE_NUMBER, SALARY, MANAGER_ID FROM EMPLOYEES WHERE (EMPLOYEE_ID = ?)
  bind => [122]
[EL Fine]: sql: SELECT EMPLOYEE_ID, COMMISSION_PCT, DEPARTMENT_ID, EMAIL, FIRST_NAME, HIRE_DATE,
JOB_ID, LAST_NAME, PHONE_NUMBER, SALARY, MANAGER_ID FROM EMPLOYEES WHERE (EMPLOYEE_ID = ?)
  bind => [100]
[EL Fine]: sql: SELECT EMPLOYEE_ID, COMMISSION_PCT, DEPARTMENT_ID, EMAIL, FIRST_NAME, HIRE_DATE,
JOB_ID, LAST_NAME, PHONE_NUMBER, SALARY, MANAGER_ID FROM EMPLOYEES WHERE (EMPLOYEE_ID = ?)

```

```
bind => [190]
[EL Fine]: sql: SELECT EMPLOYEE_ID, COMMISSION_PCT, DEPARTMENT_ID, EMAIL, FIRST_NAME, HIRE_DATE,
JOB_ID, LAST_NAME, PHONE_NUMBER, SALARY, MANAGER_ID FROM EMPLOYEES WHERE (EMPLOYEE_ID = ?)
bind => [122]
[EL Fine]: sql: SELECT EMPLOYEE_ID, COMMISSION_PCT, DEPARTMENT_ID, EMAIL, FIRST_NAME, HIRE_DATE,
JOB_ID, LAST_NAME, PHONE_NUMBER, SALARY, MANAGER_ID FROM EMPLOYEES WHERE (EMPLOYEE_ID = ?)
bind => [100]
[EL Fine]: sql: UPDATE EMPLOYEES SET SALARY = ? WHERE (EMPLOYEE_ID = ?)
bind => [3190.0, 190]
2013-12-19 14:46:53.397/45.299 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
TcpRing disconnected from Member(Id=2, Timestamp=2013-12-19 14:46:49.953, Address=130.35.99.
14:8090, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:3076,
Role=OracleHandsonRunEmployeeExample) due to a peer departure; removing the member.
2013-12-19 14:46:53.397/45.299 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2013-12-19 14:46:53.397, Address=130.35.99.14:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3076, Role=OracleHandsonRunEmployeeExample) left
Cluster with senior member 1
2013-12-19 14:46:53.397/45.300 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:EclipseLinkJPA, member=1): Member 2 left service EclipseLinkJPA with
senior member 1
2013-12-19 14:46:53.398/45.300 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 left service Management with senior member 1
```



---

---

## Interacting with the Cache and the Database

In this exercise, you will create and configure an Oracle Coherence cache in Eclipse. It highlights the use of the Coherence `CacheStore`, `ContinuousQueryCache`, `IdentityExtractor`, and `Filter` APIs.

In this exercise you will create these items:

- A Java class that creates a `NamedCache` instance and can put and get cache entries.
- A cache configuration file to define the mapping for cache names, cache types, and naming patterns.
- A Java class that creates a connection to Oracle Database and can retrieve and store table data.
- A database cache. This class will add cache entries, query the database cache, and retrieve entries.

This chapter contains the following sections:

- [Introduction](#)
- [Creating a Cache Application](#)
- [Creating a Database Cache](#)

### Introduction

A Coherence cache is a collection of data objects that acts as an intermediary between the database and the client applications. Database data can be loaded into a cache and made available to different applications. Thus, Coherence caches reduce load on the database and provide faster access to database data.

Coherence caches provide higher availability through database isolation and data replication. Modifications made to a cache can be synchronized with the database whenever the database is available. Even if the database or an application server node is not available, database updates are still reliable due to the lazy load and lazy write mechanism used by a Coherence cache and due to the failover and fail back provided by Coherence.

Coherence caches provide distributed processing not only across a cluster of application server nodes but also across the data objects in the cache, because data modification operations can be performed on the data objects.

Coherence also provides event-based processing. The state of data objects in a cache can be monitored and actions invoked on other processes such as the start of a business process execution language (BPEL) process.

Coherence supports different types of caches:

- Replicated caches, where data is replicated to each of the application server nodes in the cluster. This type of cache is recommended if faster read access is required but not suitable for write operations, because data must be written to each of the nodes. The drawback of replicated caches is that they require a large amount of memory because every node has a copy of every object.
- Distributed (or *partitioned*) caches, where data is distributed (load-balanced) across different nodes. Failover is implemented in a distributed cache using backups, which are also distributed across the cluster nodes.

Coherence is implemented by using services such as the cluster service, the distributed cache service, and the replicated cache service. Whichever type of cache is used, an application uses the same API to access and store data.

A cache configuration deployment descriptor is used to configure a cache. The root element of the cache configuration file is `cache-config`. Cache names and name patterns are mapped to cache types in the `caching-scheme-mapping` element using the subelement `cache-mapping`. Cache types are defined in the `caching-schemes` element. [Table 9-1](#) describes some of the cache types commonly used by Coherence.

**Table 9-1 Descriptions of Cache Types**

Cache Type	Description
distributed scheme	Defines a distributed cache in which data is stored across a cluster of nodes.
replicated scheme	Defines a cache in which cache entries are replicated across all the cluster nodes.
read-write-backing-map scheme	Defines a map, which provides a cache of a persistent store such as a relational database.
external scheme	Defines an external cache such as a disk.
class scheme	Defines a custom cache implementation, which is required to implement the <code>java.util.Map</code> interface.

## Creating a Cache Application

This section describes how to create and run an application that puts data into the cache and retrieves it.

1. [Create an Application that Constructs a Cache](#)
2. [Create a Cache Configuration File](#)
3. [Create a Run Configuration for the Application](#)
4. [Create the Cache Server Start-Up Configuration](#)
5. [Run the Cache Creation Application](#)

### Create an Application that Constructs a Cache

To create a Java class that constructs a Coherence cache:

1. Create a project in Eclipse.
  - a. Use the JPA perspective in Eclipse to create a JPA Project called `Interaction`. Select the `JPAConfiguration` you created in [Chapter 8, "Using JPA with Coherence."](#)

- b. In the JPA Facet page, ensure that **EclipseLink 2.5.x** appears in the **Platform** field. Ensure that **Coherence12.1.3**, **Oracle Database Driver**, and **TopLink** are selected under User Libraries. In the **Connection** field, select the connection you created in "[Configure the Project for JPA](#)" on page 8-3 (XE\_HR), from the **Connection** drop down list. Click the **Connect** link to connect to the Oracle Database if necessary. (Note, this assumes that the database connection that you created is still running.) Select **Add Driver Library to Build Path** and select **Oracle Database Driver Default** from the drop-down list. Click **Next**.
  - c. In the Coherence page, ensure that **Coherence12.1.3**, **Oracle Database Driver**, and **TopLink** are selected. Click **Finish**.
2. Create a Java class, `CoherenceCache`, that will be used to create a Coherence cache. Include a main method in the class. See "[Creating a Java Class](#)" on page 2-11 for detailed information on creating a class.

- a. Create a cache in the `CoherenceCache` Java class. Import the `CacheFactory` class and the `NamedCache` interface.

```
import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
```

- b. Create the cache (`NamedCache`) instance by using the `CacheFactory`. `getCache` method. Use the cache name `VirtualCache`, which is mapped to a distributed caching scheme.

```
NamedCache cache = CacheFactory.getCache ( "VirtualCache");
```

- c. A `NamedCache` is a `java.util.Map` instance that holds resources that are shared across nodes in a cluster. Add a cache entry by using the `put` method.

```
cache.put (key, "Hello Cache");
```

- d. Retrieve a cache entry by using the `get` method.

```
System.out.println((String)cache.get("hello"));
```

[Example 9-1](#) illustrates a possible implementation of `CoherenceCache` class. You can copy the code to the `CoherenceCache` file in Eclipse.

#### **Example 9-1 Implementation of a Coherence Cache**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

public class CoherenceCache
{
    NamedCache cache;
    public CoherenceCache()
    {
    }
    public void putCache()
    {
        cache = CacheFactory.getCache ( "VirtualCache");
        String key = "hello";
        cache.put (key, "Hello Cache");
    }

    public void retrieveCache()
    {
```

```

        System.out.println((String)cache.get("hello"));
    }

    public static void main (String [] args)
    {
        CoherenceCache cache = new CoherenceCache();
        cache.putCache();
        cache.retrieveCache();
    }
}

```

## Create a Cache Configuration File

In the Project Explorer, rename the `coherence-cache-config.xml` cache configuration file to be `cache-config.xml`. Open the file in the Eclipse Editor.

In the cache configuration file:

- Define mappings for cache names and naming patterns with the `cache-mapping` elements in the `caching-scheme-mapping` element.
- Map the cache name `VirtualCache` to cache type `distributed-eclipselink`.
- Define the distributed caching scheme with the `distributed-scheme` element using the `EclipseLinkJPA` service.

The cache configuration file is illustrated in [Example 9–2](#). Copy the contents of this example to the `cache-config.xml`.

### Example 9–2 Cache Configuration File

```

<?xml version="1.0"?>
<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
    xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-
cache-config http://xmlns.oracle.com/coherence/coherence-cache-config/1.2/
coherence-cache-config.xsd">

    <defaults>
        <serializer system-property="tangosol.coherence.serializer"/>
        <socket-provider system-property="tangosol.coherence.socketprovider"/>
    </defaults>
    <caching-scheme-mapping>

        <cache-mapping>
            <cache-name>VirtualCache</cache-name>
            <scheme-name>distributed-eclipselink</scheme-name>
        </cache-mapping>
    </caching-scheme-mapping>
    <caching-schemes>
        <!--
        Default Distributed caching scheme.
        -->
        <distributed-scheme>
            <scheme-name>distributed-eclipselink</scheme-name>
            <service-name>EclipseLinkJPA</service-name>
            <backing-map-scheme>
                <class-scheme>
                    <scheme-ref>default-backing-map</scheme-ref>
                </class-scheme>
            </backing-map-scheme>
        </distributed-scheme>
    </caching-schemes>

```

```

        </backing-map-scheme>
    <autostart>true</autostart>
</distributed-scheme>
<class-scheme>
    <scheme-name>default-backing-map</scheme-name>
    <class-name>com.tangosol.util.SafeHashMap</class-name>
</class-scheme>
</caching-schemes>
</cache-config>

```

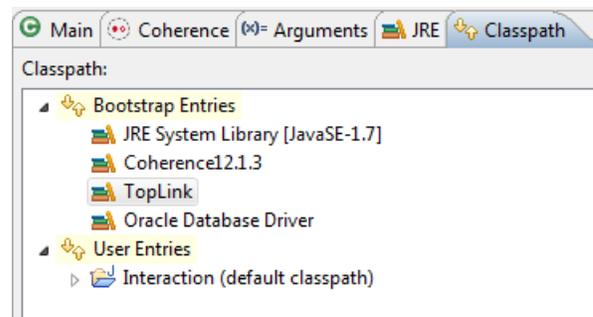
## Create a Run Configuration for the Application

Create a run configuration for the application to add the cache configuration file as a run-time Java option.

1. Right click `CoherenceCache.java` in the **Project Explorer** and choose **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, click the **New launch configuration** icon.
2. Enter `CoherenceCacheServer` in the **Name** field. Ensure that `Interaction` is in the **Project** field and `com.oracle.handson.CoherenceCache` is in the **Main class** field.
3. In the **Coherence** tab, enter the path to the cache configuration file, `C:\home\oracle\workspace\Interaction\src\cache-config.xml`. Select the **Enabled (cache server)** button. Enter a unique value, such as `3155`, in the **Cluster port** field. Click **Apply**.

The Classpath tab should look similar to [Figure 9-1](#).

**Figure 9-1** Classpath for the `CoherenceCacheServer` Executable

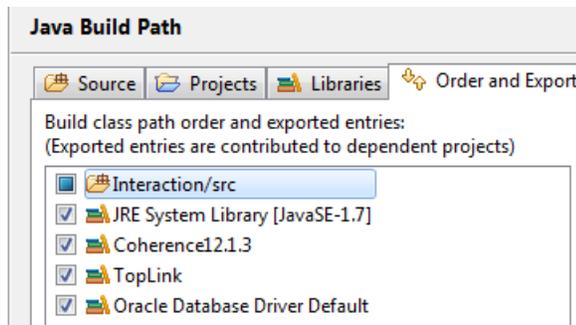


## Create the Cache Server Start-Up Configuration

To create a cache sever start-up configuration for the `Interaction` project:

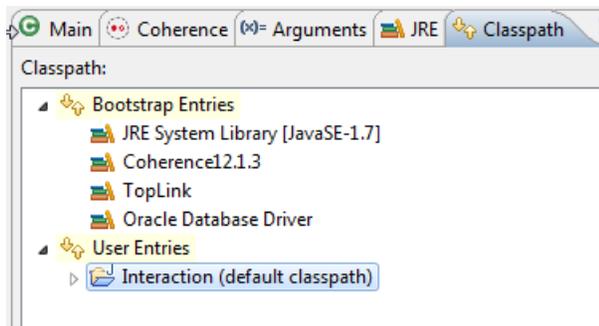
1. Right click the **Interaction** project and select **Properties**. In the **Properties for Interact** dialog box, select **Java Build Path**. In the **Order and Export** tab, the **Interaction** project, the **JRE**, **Coherence12.1.3**, **TopLink**, and the **Oracle Database Driver Default** libraries should be present. Ensure that they are all selected. The **Order and Export** tab should look similar to [Figure 9-2](#).

**Figure 9–2 Order and Export Tab for Libraries for the Java Build Path**



2. Edit the JPA cache server start-up configuration (JPAServer) that you created in [Chapter 8, "Using JPA with Coherence"](#).
  - a. Right click the **Interaction** project and select **Run As** then **Run Configurations**. In the **Main** tab click **Browse** and select the **Interaction** project from the **Project Selection** dialog box.
  - b. In the **General** tab of the **Coherence** tab, replace the name and path of the configuration file with  
`C:\home\oracle\workspace\Interaction\src\cache-config.xml`.
  - c. In the **Classpath** tab, remove the **JPA (default classpath)** folder if it appears. Click **Add Project** to add the **Interaction** project. The **Classpath** tab should look similar to [Figure 9–3](#).

**Figure 9–3 Classpath for the Interaction Project Cache Server**



- d. In the **Shared file** field of the **Common** tab, click **Browse** to select the **Interaction** project. Click **Apply**, then **Close**.

## Run the Cache Creation Application

To run the cache creation application `CoherenceCache.java`:

1. Stop any running cache servers. See ["Stopping Cache Servers"](#) on page 2-14 for more information.
2. Run the `JPAServer` to start the cache server.
3. Right-click the Coherence application `CoherenceCache.java` and click **Run As** then **Run Configurations**. Select the `CoherenceCacheServer` configuration and click **Run**. The Eclipse console window displays the output:

- The operational configuration is loaded from the `tangosol-coherence.xml` file. This file specifies the operational and run-time settings used by Coherence for its clustering, communication, and data management services.
- The cache configuration is loaded from the `cache-config.xml` file.
- A new cluster is created and the `DistributedCache` service joins the cluster.
- The output of the `CoherenceCache.java` program, `Hello Cache` is displayed.

### Example 9-3 Output of the Coherence Cache Application

```

...
2013-12-20 11:41:53.821/0.930 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "file:/C:/home/oracle/workspace/Interaction/src/cache-config.xml"
2013-12-20 11:41:54.451/1.560 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2013-12-20 11:41:55.672/2.781 Oracle Coherence GE 12.1.3.0.0 <D4> (thread=main, member=n/a): TCMP
bound to /10.159.165.75:8090 using SystemDatagramSocketProvider
2013-12-20 11:41:56.333/3.442 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=Cluster, member=n/a):
Failed to satisfy the variance: allowed=16, actual=20
...
2013-12-20 11:41:57.384/4.493 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=2): TcpAcceptor now listening for connections on 10.159.
165.75:8090.3
2013-12-20 11:41:57.744/4.853 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:EclipseLinkJPA, member=2): Service EclipseLinkJPA joined the cluster with
senior service member 1
Hello Cache

```

## Creating a Database Cache

In this section, create a cache backed by Oracle Database. This is also referred to as an Oracle Database cache.

1. [Create an Oracle Database Cache](#)
2. [Create a Class to Define a Custom Cache Store](#)
3. [Modify the Cache Configuration File](#)
4. [Create a Class to Construct the Database Cache](#)
5. [Run the Database Cache Application](#)

### Create an Oracle Database Cache

To use SQL\*Plus and the Oracle Database to create an Oracle Database cache follow these steps. You must have the database installed on your system.

1. Invoke SQL\*Plus.

Navigate to **Start** then **All Programs** then `<Oracle Database Home>` (for example, `Oracle-OraDB12Home1`) then **Application Development**, and then **SQL Plus**.

2. Connect as `hr` user with `hr` as the password.

```
connect hr/hr;
```

3. Create an Oracle Database table.

Open a text editor and copy the following SQL code. Save it as a file named `dbscript.sql` in the `/home/oracle/workspace/` folder.

**Example 9–4 SQL Script for Creating a Database Table**

```
CREATE TABLE HR.CATALOG(id VARCHAR(25) PRIMARY KEY, value VARCHAR(96));
INSERT INTO HR.CATALOG VALUES('catalog1', 'Tuning Undo Tablespace');
INSERT INTO HR.CATALOG VALUES('catalog2', 'Tuning Your View Objects');
```

4. Run the SQL script.

[Example 9–5](#) illustrates the output from the script.

**Example 9–5 Running the SQL Script for Creating a Database Table**

Copyright (c) 1982, 2013, Oracle. All rights reserved.

```
Enter user-name: system
Enter password:
Last Successful login time: Thu Dec 19 2013 11:35:04 -08:00

Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit Production With
the Partitioning, OLAP, Advanced Analytics and Real Application Testing options

SQL> connect hr/hr;
Connected.
SQL> @/home/oracle/workspace/dbscript.sql

Table created

1 row created

1 row created
```

## Create a Class to Define a Custom Cache Store

To create a Java class that connects to the database and retrieves table data:

1. Create a Java class `DBCacheStore` in the **Interaction** project in Eclipse. See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Create the code to connect to the database and get table data.

[Example 9–6](#) illustrates a possible implementation of the `DBCacheStore` class. Copy the code to the `DBCacheStore` application in the Eclipse IDE. The `DBCacheStore` application uses Java Database Connectivity (JDBC) to access Oracle Database, but you could use another mechanism, such as Hibernate or Java Data Objects (JDO), instead.

**Example 9–6 Database Cache Store Implementation**

```
package com.oracle.handson;

import com.tangosol.net.cache.CacheStore;
import com.tangosol.util.Base;

import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```

import java.sql.SQLException;
import java.util.Collection;
import java.util.Iterator;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;

public class DBCacheStore

    extends Base
    implements CacheStore {

    protected Connection m_con;
    protected String m_sTableName;
    private static final String DB_DRIVER    = "oracle.jdbc.OracleDriver";

        private static final String DB_URL    =
"jdbc:oracle:thin:@localhost:1521:orcl";
        private static final String DB_USERNAME = "hr";
        private static final String DB_PASSWORD = "hr";

    public DBCacheStore(String sTableName)
    {
        m_sTableName = sTableName;

        configureConnection();
    }
    protected void configureConnection()
    {
        try
        {
            Class.forName("oracle.jdbc.OracleDriver");
            m_con = DriverManager.getConnection(DB_URL, DB_USERNAME, DB_PASSWORD);
            m_con.setAutoCommit(true);
        }
        catch (Exception e)
        {
            throw ensureRuntimeException(e, "Connection failed");
        }
    }

    public String getTableName()
    {
        return m_sTableName;
    }

    public Connection getConnection()
    {
        return m_con;
    }

    public Object load(Object oKey)
    {
        Object    oValue = null;
        Connection con    = getConnection();
        String    sSQL    = "SELECT id, value FROM " + getTableName()
            + " WHERE id = ?";
        try
        {

```

```
        PreparedStatement stmt = con.prepareStatement(sSQL);

        stmt.setString(1, String.valueOf(oKey));
        ResultSet rslt = stmt.executeQuery();
        if (rslt.next())
        {
            oValue = rslt.getString(2);
            if (rslt.next())
            {
                throw new SQLException("Not a unique key: " + oKey);
            }
        }
        stmt.close();
    }
    catch (SQLException e)
    {
        throw ensureRuntimeException(e, "Load failed: key=" + oKey);
    }
    return oValue;
}

public void store(Object oKey, Object oValue)
{
    Connection con    = getConnection();
    String      sTable = getTableName();
    String      sSQL;

    if (load(oKey) != null)
    {
        sSQL = "UPDATE " + sTable + " SET value = ? where id = ?";
    }
    else
    {
        sSQL = "INSERT INTO " + sTable + " (value, id) VALUES (?,?)";
    }
    try
    {
        PreparedStatement stmt = con.prepareStatement(sSQL);
        int i = 0;
        stmt.setString(++i, String.valueOf(oValue));
        stmt.setString(++i, String.valueOf(oKey));
        stmt.executeUpdate();
        stmt.close();
    }
    catch (SQLException e)
    {
        throw ensureRuntimeException(e, "Store failed: key=" + oKey);
    }
}

public void erase(Object oKey)
{
    Connection con = getConnection();
    String      sSQL = "DELETE FROM " + getTableName() + " WHERE id=?";
    try
    {
        PreparedStatement stmt = con.prepareStatement(sSQL);

        stmt.setString(1, String.valueOf(oKey));
    }
}
```

```

        stmt.executeUpdate();
        stmt.close();
    }
    catch (SQLException e)
    {
        throw ensureRuntimeException(e, "Erase failed: key=" + oKey);
    }
}

public void eraseAll(Collection colKeys)
{
    throw new UnsupportedOperationException();
}

public Map loadAll(Collection colKeys)
{
    throw new UnsupportedOperationException();
}

public void storeAll(Map mapEntries)
{
    throw new UnsupportedOperationException();
}

public Iterator keys()
{
    Connection con = getConnection();
    String      sSQL = "SELECT id FROM " + getTableName();
    List        list = new LinkedList();

    try
    {
        PreparedStatement stmt = con.prepareStatement(sSQL);
        ResultSet        rslt = stmt.executeQuery();
        while (rslt.next())
        {
            Object oKey = rslt.getString(1);
            list.add(oKey);
        }
        stmt.close();
    }
    catch (SQLException e)
    {
        throw ensureRuntimeException(e, "Iterator failed");
    }

    return list.iterator();
}
}

```

## Modify the Cache Configuration File

Modify the cache configuration file (`cache-config.xml`) that you created earlier for the database cache. To connect a cache to a database, you must configure the `cachestore-scheme` element with a custom class that implements either the `com.tangosol.net.cache.CacheLoader` or `com.tangosol.net.cache.CacheStore` interface.

Replace the code in the existing `cache-config.xml` file in Eclipse with the cache configuration code for the database cache in [Example 9-7](#).

**Example 9-7 Database Cache Configuration File**

```

<?xml version="1.0" encoding="UTF-8" ?>
<cache-config>
  <caching-scheme-mapping>

    <!--
      Caches with names that start with 'DBBacked' will be created
      as distributed-db-backed.
    -->
    <cache-mapping>
      <cache-name>DBBacked* </cache-name>
      <scheme-name>distributed-db-backed </scheme-name>
    </cache-mapping>
  </caching-scheme-mapping>
</caching-schemes>
<!--
  DB Backed Distributed caching scheme.
-->
<distributed-scheme>
  <scheme-name>distributed-db-backed </scheme-name>
  <service-name>DistributedCache </service-name>
  <backing-map-scheme>
    <read-write-backing-map-scheme>
      <internal-cache-scheme>
        <class-scheme>
          <class-name>com.tangosol.util.ObservableHashMap </class-name>
        </class-scheme>
      </internal-cache-scheme>
      <cachestore-scheme>
        <class-scheme>
          <class-name>com.oracle.handson.DBCacheStore </class-name>
          <init-params>
            <init-param>
              <param-type>java.lang.String </param-type>
              <param-value>CATALOG </param-value>
            </init-param>
          </init-params>
        </class-scheme>
      </cachestore-scheme>
      <read-only>>false</read-only>
    <!--
      To make this a write-through cache just change the value below to 0 (zero)
    -->
    <write-delay-seconds>0 </write-delay-seconds>
  </read-write-backing-map-scheme>
</backing-map-scheme>
<autostart>true </autostart>
</distributed-scheme>
</caching-schemes>
</cache-config>

```

In the cache configuration file, you have done the following:

- Defined a cache name pattern `DBBacked*`, which is mapped to a distributed caching scheme `distributed-db-backed`.
- Specified the `CacheStore` scheme in the distributed scheme using the class `coherence.DBCacheStore`, which implements the `CacheStore` interface.
- Specified for the `DBCacheStore` class an `init` parameter for the database table that is at the back end of the cache. The table name is specified in the `init-param`

element. The `DBCachedStore` class performs database operations such as reading and writing cache entries.

- Specified a `read-write-backing-map-scheme` as the backing map. This scheme defines a backing map, which provides a size-limited cache of a persistent store. Here, by setting the `write-delay-seconds` parameter to 0, you specify the write-through mechanism.

[Table 9-2](#) describes the types of read/write caching that you can use with Coherence.

**Table 9-2 Types of Read/Write Caching Supported by Coherence**

Types of Read/Write Caching	Action
read-through	A cache entry is read into a cache from the database when required and made available to an application.
write-through	Updates to cache entries are synchronized with the database without a delay.
refresh-ahead	Cache entries are refreshed periodically.
write-behind	Updates to cache entries are asynchronously written to a database after a delay specified in the <code>write-delay-seconds</code> element in the cache configuration file.

## Create a Class to Construct the Database Cache

Create a Java class `DatabaseCache` for the database cache in the **Interaction** project in Eclipse. The class must contain a `main` method. See ["Creating a Java Class"](#) on page 2-11 for detailed information.

In the class file, provide the code to add a cache entry, query a database cache, and retrieve a cache entry. Add the following methods: `createCache()`, `addEntry()`, `retrieveEntry()`, `eraseEntry()`, and `queryCache()`. [Example 9-8](#) illustrates a possible implementation.

**Example 9-8 Implementation for the DatabaseCache Class File**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.net.cache.ContinuousQueryCache;
import com.tangosol.util.Filter;
import com.tangosol.util.extractor.IdentityExtractor;
import com.tangosol.util.filter.LikeFilter;

import java.util.HashSet;

import java.util.Iterator;
import java.util.Map;
import java.util.Set;

public class DatabaseCache {
    NamedCache cache;
    public DatabaseCache() {
    }

    public void createCache()
    {
        cache = CacheFactory.getCache("DBBackedCache");
    }
}
```

```

    }

    public void addEntry()
    {
        cache.put(new String("catalog3"), new String("Tuning Grid Management"));
        cache.put(new String("catalog4"), new String("Tuning Coherence"));
        cache.put(new String("catalog5"), new String("Tuning Database"));
    }

    public void retrieveEntry()
    {
        System.out.println((String) cache.get( "catalog3"));
    }

    public void eraseEntry()
    {
        cache.remove(new String("catalog3"));
    }

    public void queryCache()
    {
        Filter filter = new LikeFilter(IdentityExtractor.INSTANCE, "Tuning%", '\\\\',
true);
        HashSet hashSet=new HashSet();
        hashSet.add(new String("catalog3"));
        hashSet.add(new String("catalog4"));
        hashSet.add(new String("catalog5"));

        Map map=cache.getAll(hashSet);
        Set results = cache.entrySet(filter);

        for (Iterator i = results.iterator(); i.hasNext();)
        {
            Map.Entry e = (Map.Entry) i.next();
            System.out.println("Catalog ID: "+e.getKey() + ", Title: "+e.
getValue());
        }
    }

    public static void main(String[] args)
    {
        DatabaseCache databaseCache = new DatabaseCache();
        databaseCache.createCache();
        databaseCache.addEntry();
        databaseCache.queryCache();
    }
}

```

Note the following features of the database cache class file:

- A `NamedCache` object is created using the `getCache` method of the `CacheFactory` class in the `createCache` method.

```
NamedCache cache = CacheFactory.getCache("DBBackedCache");
```
- The `DBBackedCache` matches the cache pattern `DBBacked*` and is, therefore, mapped to a distributed caching scheme `distributed-db-backed` in the `cache-config.xml` file. Add a cache entry using the `put()` method of the `NamedCache` object.

```
cache.put(new String("catalog3"), new String("Tuning Grid Management"));
```

- Because the write-through mechanism is used, the new cache entry gets synchronized with the database; a new row is added to the CATALOG table. Comment out all the methods except the `createCache` and `addEntry` methods.
- When the `put` method is invoked, the `store` method, which maps the new cache entry to the database table CATALOG using JDBC, gets invoked in the `DBCachedStore` class. The output from the Coherence application is displayed in the Log window and a new cache entry is added. The output shows that the operational configuration deployment descriptor is loaded, the cache configuration is loaded, a new cluster is created, and the `DistributedCache` service has joined the cluster.
- The new cache entry can be removed with the `remove` method of the `NamedCache` object.

```
cache.remove(new String("catalog3"));
```

- Bulk uploading of cache entries is performed using the `putAll` method.
- A cache entry is retrieved using the `get` method of the `NamedCache` object. For example, retrieving the cache entry for ID `catalog1`:

```
System.out.println((String) cache.get("catalog1"));
```

- When the `get()` method is invoked, the `load` method, which retrieves database table data using JDBC, gets invoked in the `DBCachedStore` class.
- Bulk retrieval is performed using the `getAll` method of the `NamedCache` object.
- Coherence supports searching for cache entries based on a search criteria using filters. Coherence filters are available in the `com.tangosol.util.filter` package. In Oracle Coherence Enterprise Edition and Grid Edition, indexes can be added to the Coherence cache to improve performance. You query the database cache using a `LikeFilter` filter, which matches cache entries with a specified pattern. To query a database cache, the cache entries must be created before querying, and the cache entries must be retrieved into the cache using the `get()` or `getAll()` method before a query using a filter can be performed. Therefore, you can retrieve database data and create a collection of cache entries using the `getAll()` method.

```
HashSet hashSet=new HashSet();
hashSet.add(new String("catalog1"));
hashSet.add(new String("catalog2"));
hashSet.add(new String("catalog3"));
Map map=cache.getAll(hashSet);
```

- Create a `LikeFilter` filter to search for cache entries starting with `Tuning`.

```
Filter filter = new LikeFilter(IdentityExtractor.INSTANCE, "Tuning%", '\\',
true);
```

- Query the database cache using the `entrySet()` method with the `LikeFilter` filter.

```
Set results = cache.entrySet(filter);
```

- Iterate over the results of the query. Display the key and value of the cache entries that are retrieved.

```
for (Iterator i = results.iterator(); i.hasNext(); )
{
```

```
Map.Entry e = (Map.Entry) i.next();
System.out.println("Catalog ID: "+e.getKey() + ", Title: "+e.getValue());
}
```

- Coherence supports continuous query using the `com.tangosol.net.cache.ContinuousQueryCache` class. A continuous query is a query that is kept up-to-date using a continuous query cache. In a `ContinuousQueryCache`, the results of a query are updated using event listeners on events that could change the results of the query. Create a `ContinuousQueryCache` object using the `NamedCache` object and the `LikeFilter` object.

```
ContinuousQueryCache queryCache = new ContinuousQueryCache(cache, filter);
```

- Create a result set by using the `entrySet()` method.

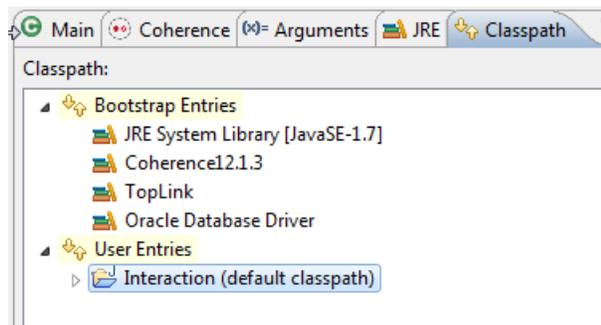
```
Set results = queryCache.entrySet(filter);
```

## Run the Database Cache Application

To run the Oracle Database cache application:

- Stop any running cache servers. See ["Stopping Cache Servers"](#) on page 2-14 for more information.
- Start the cache server (`JPAServer`). Right click the project and select **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, select `JPACacheServer` to display its configuration. Click **Run**.
- Create a run configuration for the `DatabaseCache` program.
  - In the **Run Configurations** dialog box, click the **New launch configuration** icon.
  - Enter `DatabaseCacheClient` in the **Name** field. Ensure that `Interaction` is in the **Project** field and `com.oracle.handson.DatabaseCache` is in the **Main class** field.
  - In the **Coherence** tab, enter the path to the cache configuration file in the **Cache configuration descriptor** field:  
`C:\home\oracle\workspace\Interaction\src\cache-config.xml`.  
 Select **Local storage: Disabled (cache client)**. Enter a unique value, such as 3155, in the **Cluster port** field.
  - Open the **Classpath** tab. The **Classpath** tab should look similar to [Figure 9-4](#).

**Figure 9-4 Classpath for the DatabaseCache Program**



- Click **Apply** then **Run**.

[Example 9-9](#) illustrates the expected results.

**Example 9–9 Output of the DatabaseCache Program**

```

...
Group{Address=224.12.1.0, Port=3155, TTL=4}

MasterMemberSet (
  ThisMember=Member(Id=2, Timestamp=2013-12-20 12:28:41.013, Address=10.159.165.75:8090,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:1976, Role=OracleHandsonDatabaseCache)
  OldestMember=Member(Id=1, Timestamp=2013-12-20 12:27:53.503, Address=10.159.165.75:8088,
MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:7320, Role=CoherenceServer)
  ActualMemberSet=MemberSet(Size=2
    Member(Id=1, Timestamp=2013-12-20 12:27:53.503, Address=10.159.165.75:8088, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:7320, Role=CoherenceServer)
    Member(Id=2, Timestamp=2013-12-20 12:28:41.013, Address=10.159.165.75:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:1976, Role=OracleHandsonDatabaseCache)
  )
  MemberId|ServiceVersion|ServiceJoined|MemberState
  1|12.1.3|2013-12-20 12:27:53.503|JOINED,
  2|12.1.3|2013-12-20 12:28:41.013|JOINED
  RecycleMillis=1200000
  RecycleSet=MemberSet(Size=0
  )
)

TcpRing{Connections=[1]}
IpMonitor{Addresses=0}

2013-12-20 12:28:41.265/3.800 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=2): Service Management joined the cluster with senior service member 1
2013-12-20 12:28:41.295/3.830 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=2): Loaded
Reporter configuration from "jar:file:/C:/Oracle/coherence/lib/coherence.jar!/reports/report-group.
xml"
2013-12-20 12:28:41.715/4.250 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=NameService:TcpAcceptor, member=2): TcpAcceptor now listening for connections on 10.159.
165.75:8090.3
2013-12-20 12:28:42.086/4.621 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=2): Service DistributedCache joined the cluster with senior service member 1
Catalog ID: catalog5, Title: Tuning Database
Catalog ID: catalog4, Title: Tuning Coherence
Catalog ID: catalog3, Title: Tuning Grid Management

```

[Example 9–10](#) illustrates the cache server response to the DatabaseCache program.

**Example 9–10 Cache Server Response to the DatabaseCache Program**

```

...
Started DefaultCacheServer...

2013-12-20 12:28:41.208/50.536 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2013-12-20 12:28:41.013, Address=10.159.165.75:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:1976, Role=OracleHandsonDatabaseCache) joined Cluster
with senior member 1
2013-12-20 12:28:41.275/50.603 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 joined Service Management with senior member 1
2013-12-20 12:28:42.296/51.624 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): Member 2 joined Service DistributedCache with senior member 1
2013-12-20 12:28:42.377/51.705 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
TcpRing disconnected from Member(Id=2, Timestamp=2013-12-20 12:28:41.013, Address=10.159.165.
75:8090, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:1976,
Role=OracleHandsonDatabaseCache) due to a peer departure; removing the member.

```

```
2013-12-20 12:28:42.377/51.705 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2013-12-20 12:28:42.377, Address=10.159.165.75:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:1976, Role=OracleHandsonDatabaseCache) left Cluster
with senior member 1
2013-12-20 12:28:42.378/51.706 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=DistributedCache,
member=1): Member 2 left service DistributedCache with senior member 1
2013-12-20 12:28:42.378/51.706 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 left service Management with senior member 1
```

If you receive any exceptions, such as the following:

```
java.lang.IllegalArgumentException: No scheme for cache: "cachename,
```

You may be able to remove them by editing the `cache-config.xml` file and replacing `DBBacked*` in the `<cache-name>` element with `*`. Re-run the `DatabaseCache` application in Eclipse. You should not see any exceptions.

4. Note that because you are using a write-through cache, the database table is also updated. From the SQL prompt, enter the following code:

```
select * from hr.catalog;
```

[Example 9-11](#) illustrates the results.

**Example 9-11 Output from the SELECT Statement**

```
...
SQL> select * from hr.catalog;

ID
-----
VALUE
-----

catalog1
Tuning Undo Tablespace

catalog2
Tuning Your View Objects

catalog3
Tuning Grid Management

ID
-----
VALUE
-----

catalog4
Tuning Coherence

catalog5
Tuning Database

SQL>
```

---

---

## Working with Security

This exercise describes how to apply security to a Oracle Coherence\*Extend client and highlights the use of the Coherence `SecurityHelper`, `PofPrincipal`, `IdentityTransformer`, `IdentityAsserter`, and `WrapperCacheService` APIs.

Coherence\*Extend allows a wide range of access to Coherence caches. These include desktop applications, remote servers, and machines located across wide area network (WAN) connections.

This chapter contains the following sections:

- [Introduction](#)
- [Enabling Token-Based Security](#)
- [Including Role-Based Access Control to the Cluster](#)
- [Including Role-Based Access Control to an Invocable Object](#)

### Introduction

Coherence\*Extend consists of an extend client running outside the cluster and a proxy service running inside the cluster, hosted by one or more cache servers. The client APIs route all requests to the proxy. The proxy responds to client requests by delegating to Coherence clustered services, such as a partitioned or replicated cache service or an invocation service.

Because the extend client exists outside of the cluster, the issue of securing access to the cluster takes on greater importance. This chapter describes three techniques that you can use to secure access between the client and the cluster. The techniques include using identity token-based passwords, an entitled cache service, and an invocation service.

A detailed discussion of these security techniques and extend clients is beyond the scope of this tutorial. For more information on these topics, see *Oracle Fusion Middleware Securing Oracle Coherence*.

### Enabling Token-Based Security

You can implement token-based security to enable access between an extend client and an extend proxy in the cluster. To enable access between the extend client and an extend proxy the following files are typically required:

- Client application files, which describe the functionality that will access the cluster
- Cache configuration files, where the extend client and the extend proxy each have their own cache configuration

- Operational override file, which overrides the operational and run-time settings in the default operational deployment descriptor
- Server start-up files, where there is a start-up file for the extend proxy and for the cache server in the cluster
- POF configuration deployment descriptor, which specifies custom data types when using POF to serialize objects

To add token-based security, you must also provide identity transformer and asserter implementations. The transformer generates the token on the client side and the asserter validates it on the cluster side.

The following steps describe how to create and run an application for an extend client that uses token-based security to access the cluster.

1. [Use a Security Helper File](#)
2. [Create an Identity Transformer](#)
3. [Create an Identity Asserter](#)
4. [Create the Password File](#)
5. [Enable the Identity Transformer and Asserter](#)
6. [Create a Cache Configuration File for the Extend Client](#)
7. [Create a Cache Configuration File for the Extend Proxy](#)
8. [Create a Start-Up Configuration for a Cache Server with a Proxy Service](#)
9. [Create a Start-Up Configuration for a Cache Server](#)
10. [Run the Password Example](#)

## Use a Security Helper File

The examples in this chapter reference a security helper file that defines role-based security policies and access control to the cache. For the purpose of these examples, a file with simplified mappings is provided for you.

Cache access is determined by a user's role. The security helper file defines several roles: `role_reader`, `role_writer`, and `role_admin`. It defines the mappings of various users to the roles, such as `BuckarooBanzai` to `ROLE_ADMIN`. It defines the mappings of roles to integer IDs, such as `ROLE_ADMIN` to `9`. The helper file also defines the cache name and the invocation service name used in the examples.

The key features of this file are the `login` and `checkAccess` methods. The `login` method takes a user name and constructs a simplified distinguished name (DN). It then associates a role with the name. `PofPrincipal` provides the `Principal` implementation.

The `checkAccess` method shows where the authorization code is placed. It determines whether the user can access the cache based on a provided user role.

To create a new project and the security helper file:

1. In Eclipse IDE, select the Java EE perspective and create a new Application Client Project called `Security`. Select **CoherenceConfig** from the **Configuration** drop-down list. Ensure that the **Create a default main** is *not* selected on the Application Client module page.

In the Coherence page, select *only* **Coherence12.1.3**.

See ["Creating a New Project in the Eclipse IDE"](#) on page 2-3 for detailed information on creating a project.

2. Create a new Java file named `SecurityExampleHelper.java`. Ensure that the package path is `com.oracle.handson`.

See ["Creating a Java Class"](#) on page 2-11 for detailed information on creating a Java class.

3. Copy the code illustrated in [Example 10–1](#) into the file.

**Example 10–1 A Security Helper File**

```
package com.oracle.handson;

import com.tangosol.io.pof.PofPrincipal;

import com.tangosol.net.security.SecurityHelper;

import java.util.HashMap;
import java.util.HashSet;
import java.util.Iterator;
import java.util.Map;
import java.util.Set;

import java.security.Principal;

import javax.security.auth.Subject;

/**
 * This class provides extremely simplified role based policies and access control.
 *
 */
public class SecurityExampleHelper
{
    // ----- static methods -----

    /**
     * Login the user.
     *
     * @param sName the user name
     *
     * @return the authenticated user
     */
    public static Subject login(String sName)
    {
        // For simplicity, just create a Subject. Normally, this would be
        // done using JAAS.
        String sUserDN = "CN=" + sName + ",OU=Yoyodyne";
        Set setPrincipalUser = new HashSet();

        setPrincipalUser.add(new PofPrincipal(sUserDN));
        // Map the user to a role
        setPrincipalUser.add(new PofPrincipal((String) s_mapUserToRole.
get(sName)));

        return new Subject(true, setPrincipalUser, new HashSet(), new HashSet());
    }
}
```

```

/**
 * Assert that a Subject is associated with the calling thread with a
 * Principal representing the required role.
 *
 * @param sRoleRequired the role required for the operation
 *
 * @throws SecurityException if a Subject is not associated with the
 *         calling thread or does not have the specified role Principal
 */
public static void checkAccess(String sRoleRequired)
    {
        checkAccess(sRoleRequired, SecurityHelper.getCurrentSubject());
    }

/**
 * Assert that a Subject contains a Principal representing the required
 * role.
 *
 * @param sRoleRequired the role required for the operation
 *
 * @param subject the Subject requesting access
 *
 * @throws SecurityException if a Subject is null or does not have the
 *         specified role Principal
 */
public static void checkAccess(String sRoleRequired, Subject subject)
    {
        if (subject == null)
            {
                throw new SecurityException("Access denied, authentication
required");
            }

        Map mapRoleToId = s_mapRoleToId;
        Integer nRoleRequired = (Integer) mapRoleToId.get(sRoleRequired);

        for (Iterator iter = subject.getPrincipals().iterator(); iter.hasNext();)
            {
                Principal principal = (Principal) iter.next();
                String sName = principal.getName();

                if (sName.startsWith("role_"))
                    {
                        Integer nRolePrincipal = (Integer) mapRoleToId.get(sName);

                        if (nRolePrincipal == null)
                            {
                                // invalid role
                                break;
                            }

                        if (nRolePrincipal.intValue() >= nRoleRequired.intValue())
                            {
                                return;
                            }
                    }
            }

        throw new SecurityException("Access denied, insufficient privileges");
    }

```

```

// ----- constants -----
public static final String ROLE_READER = "role_reader";

public static final String ROLE_WRITER = "role_writer";

public static final String ROLE_ADMIN = "role_admin";

/**
 * The cache name for security examples
 */
public static final String SECURITY_CACHE_NAME = "security";

/**
 * The name of the InvocationService used by security examples.
 */
public static String INVOCATION_SERVICE_NAME = "ExtendTcpInvocationService";

// ----- static data -----

/**
 * The map keyed by user name with the value being the user's role.
 * Represents which user is in which role.
 */
private static Map s_mapUserToRole = new HashMap();

/**
 * The map keyed by role name with the value the role id.
 * Represents the numeric role identifier.
 */
private static Map s_mapRoleToId = new HashMap();

// ----- static initializer -----

static
{
    // User to role mapping
    s_mapUserToRole.put("BuckarooBanzai", ROLE_ADMIN);
    s_mapUserToRole.put("JohnWhorfin", ROLE_WRITER);
    s_mapUserToRole.put("JohnBigboote", ROLE_READER);

    // Role to Id mapping
    s_mapRoleToId.put(ROLE_ADMIN, Integer.valueOf(9));
    s_mapRoleToId.put(ROLE_WRITER, Integer.valueOf(2));
    s_mapRoleToId.put(ROLE_READER, Integer.valueOf(1));
}
}

```

## Create an Identity Transformer

An identity transformer (`com.tangosol.net.security.IdentityTransformer`) is a client-side component that converts a subject or principal into an identity token. The token must be a type that Coherence can serialize. Coherence automatically serializes the token at run time and sends it as part of the connection request to the proxy.

To create an identity transformer implementation:

1. Create a new Java class in the `Security` project named `PasswordIdentityTransformer`.  
See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Import the `IdentityTransformer` interface. Ensure that the `PasswordIdentityTransformer` class implements the `IdentityTransformer` interface.
3. Implement the `transformIdentity` method so that it performs the following tasks:
  - Tests whether the subject exists and is complete
  - Gets the principal names from the subject and saves them in a `String` array
  - Constructs the token, as a combination of the password plus the principal names, that can be serialized as POF types

[Example 10-2](#) illustrates a possible implementation of the `PasswordIdentityTransformer` class.

**Example 10-2 Sample Identity Transformer Implementation**

```
package com.oracle.handson;

import com.tangosol.net.security.IdentityTransformer;

import java.security.Principal;
import java.util.Iterator;
import java.util.Set;

import javax.security.auth.Subject;
import com.tangosol.net.Service;

/**
 * PasswordIdentityTransformer creates a security token that contains the
 * required password and then adds a list of Principal names.
 */
public class PasswordIdentityTransformer
    implements IdentityTransformer
{
    // ----- IdentityTransformer interface -----

    /**
     * Transform a Subject to a token that asserts an identity.
     *
     * @param subject the Subject representing a user.
     *
     * @return the token that asserts identity.
     *
     * @throws SecurityException if the identity transformation fails.
     */
    public Object transformIdentity(Subject subject, Service service)
        throws SecurityException
    {
        // The service is not needed so the service argument is being ignored.
        // It could be used, for example, if there were different token types
    }
}
```

```

// required per service.

    if (subject == null)
        {
            throw new SecurityException("Incomplete Subject");
        }

    Set setPrincipals = subject.getPrincipals();

    if (setPrincipals.isEmpty())
        {
            throw new SecurityException("Incomplete Subject");
        }

    String[] asPrincipalName = new String[setPrincipals.size() + 1];
    int      i                = 0;

    asPrincipalName[i++] = System.getProperty("coherence.password",
        "secret-password");

    for (Iterator iter = setPrincipals.iterator(); iter.hasNext();)
        {
            asPrincipalName[i++] = ((Principal) iter.next()).getName();
        }

    // The token consists of the password plus the principal names as an
    // array of pof-able types, in this case strings.
    return asPrincipalName;
}
}

```

## Create an Identity Asserter

An identity asserter (`com.tangosol.net.security.IdentityAsserter`) is a cluster-side component on the cache server that hosts an extend proxy service. The asserter validates that the token created by the identity transformer on the extend client contains the required credentials to access the cluster.

To create an identity asserter implementation:

1. Create a new Java class in the `Security` project named `PasswordIdentityAsserter`.  
See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Import the `IdentityAsserter` interface. Ensure that the `PasswordIdentityAsserter` class implements the `IdentityAsserter` interface.
3. Implement the `assertIdentity` method such that it:
  - Validates that the token contains the correct password and that the password is the first name in the token

[Example 10-3](#) illustrates a possible implementation of the `PasswordIdentityAsserter` class.

### **Example 10-3** Sample Identity Asserter Implementation

```

package com.oracle.handson;

import com.tangosol.io.pof.PofPrincipal;

```

```
import com.tangosol.net.security.IdentityAsserter;

import java.util.HashSet;
import java.util.Set;

import javax.security.auth.Subject;
import com.tangosol.net.Service;

/**
 * PasswordIdentityAsserter asserts that the security token contains the
 * required password and then constructs a Subject based on a list of
 * Principal names.
 */
public class PasswordIdentityAsserter
    implements IdentityAsserter
{
    // ----- IdentityAsserter interface -----

    /**
     * Asserts an identity based on a token-based identity assertion.
     *
     * @param oToken the token that asserts identity.
     *
     * @return a Subject representing the identity.
     *
     * @throws SecurityException if the identity assertion fails.
     */
    public Subject assertIdentity(Object oToken, Service service)
        throws SecurityException
    {
        // The service is not needed so the service argument is being ignored.
        // It could be used, for example, if there were different token types
        // required per service.
        if (oToken instanceof Object[])
        {
            String sPassword = System.getProperty(
                "coherence.password", "secret-password");
            Set setPrincipalUser = new HashSet();
            Object[] asName = (Object[]) oToken;

            // first name must be password
            if (((String) asName[0]).equals(sPassword))
            {
                // prints the user name to server shell to ensure we are
                // communicating with it and to ensure user is validated
                System.out.println("Password validated for user: " + asName[1]);
                for (int i = 1, len = asName.length; i < len; i++)
                {
                    setPrincipalUser.add(new PofPrincipal((String)asName[i]));
                }

                return new Subject(true, setPrincipalUser, new HashSet(),
                    new HashSet());
            }
            throw new SecurityException("Access denied");
        }
    }
}
```

```
}

```

## Create the Password File

Create a Java file that requires a password to get a reference to a cache. Use the `SecurityExampleHelper.login("BuckarooBanzai")` to call the `login` method in the `SecurityExampleHelper` file to generate a token. At run time, the user name is associated with its subject defined in the `SecurityExampleHelper` class. A token is generated from this subject by the `PasswordIdentityTransformer` class and validated by the `PasswordIdentityAsserter` class as part of the connection request. If the validation succeeds, then a connection to the proxy and a reference to the cache is granted. Use the `Subject.doAs` method to make the subject available in the security context.

1. Create a new Java class with a main method in the `Security` project named `PasswordExample`.  
See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Implement the main method to get a reference to the cache.
3. Use the `SecurityExampleHelper.login` method to get a subject for user `BuckarooBanzai`.
4. Implement the `doAs` method to make the subject part of the Java security context. The subject will be available to any subsequent code. In this case, the `doAs` method is implemented to validate whether the user can access the cache based on its defined role.

[Example 10-4](#) illustrates a possible implementation of `PasswordExample`.

### **Example 10-4 Sample Implementation to Run the Password Example**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import java.io.IOException;

import java.security.PrivilegedExceptionAction;

import javax.security.auth.Subject;
import com.tangosol.net.Service;

/**
 * This class shows how a Coherence Proxy can require a password to get a
 * reference to a cache.
 * <p>
 * The PasswordIdentityTransformer will generate a security token that
 * contains the password. The PasswordIdentityAsserter will validate the
 * security token to enforce the password. The token generation and
 * validation occurs automatically when a connection to the proxy is made.
 *
 */
public class PasswordExample
{
    // ----- static methods -----
}
```

```

/**
 * Get a reference to the cache. Password will be required.
 */
public static void main (String[] args){
    getCache();
}

public static void getCache()
{
    System.out.println("-----password example begins-----");

    Subject subject = SecurityExampleHelper.login("BuckarooBanzai");

    try
    {
        NamedCache cache = (NamedCache) Subject.doAs(
            subject, new PrivilegedExceptionAction()
            {
                public Object run()
                    throws Exception
                {
                    NamedCache cache;

                    cache = CacheFactory.getCache(
                        SecurityExampleHelper.SECURITY_CACHE_NAME);
                    System.out.println("-----password example succeeded-----");
                    return cache;
                }
            });
    }
    catch (Exception e)
    {
        // get exception if the password is invalid
        System.out.println("Unable to connect to proxy");
        e.printStackTrace();
    }
    System.out.println("-----password example completed-----");
}
}

```

## Enable the Identity Transformer and Asserter

Configure an operational override file (`tangosol-coherence-override.xml`) to identify the classes that define the identity transformer (the class that transforms a `Subject` to a token on the extend client) and the identity asserter (the class that validates the token on the cluster).

1. Open the `tangosol-coherence-override.xml` file from the Project Explorer window. You can find the file under `Security/appClientModule`.
2. Use the `identity-transformer` and `identity-asserter` elements within the `security-config` stanza to identify the full path to the `PasswordIdentityTransformer` and `PasswordIdentityAsserter` implementation classes, respectively. Set the `subject-scope` parameter to `true` to associate the identity from the current security context with the cache and remote invocation service references that are returned to the client.

[Example 10-5](#) illustrates a possible implementation of the `tangosol-coherence-override.xml` file.

**Example 10–5 Specifying an Identity Transformer and an Asserter**

```
<?xml version="1.0" encoding="UTF-8"?>

<?xml version="1.0" encoding="UTF-8"?>
<coherence xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-operational-config
http://xmlns.oracle.com/coherence/coherence-operational-config/1.2/coherence-
operational-config.xsd">
  <!--coherence-version:12.1.3-->
  <security-config>
    <identity-asserter>
      <class-name>com.oracle.handson.PasswordIdentityAsserter</class-name>
    </identity-asserter>
    <identity-transformer>
      <class-name>com.oracle.handson.PasswordIdentityTransformer</class-name>
    </identity-transformer>
    <subject-scope>true</subject-scope>
  </security-config>
</coherence>
```

**Create a Cache Configuration File for the Extend Client**

The cache configuration file for the extend client routes cache operations to an extend proxy in the cluster. At run time, cache operations are not executed locally; instead, they are sent to the extend proxy service.

To create a cache configuration file for an extend client:

1. Open the `coherence-cache-config.xml` file from the Project explorer window. You can find the file under `Security/appClientModule`.
2. Save the file as `client-cache-config.xml`.
3. Write the extend client cache configuration. The following list highlights some key elements:
  - Use the `cache-name` element to define `security` as the name of the cache. Note that there must be a cache defined in the cluster-side cache configuration that is also named `security`.
  - Use the `remote-cache-scheme` stanza to define the details about the remote cache.
  - Use the `address` and `port` elements in the `tcp-initiator` stanza to identify the extend proxy service that is listening on the `localhost` address at port `9099`.
  - Use `defaults` and `serializer` with a value of `pof` to call the serializer for the custom POF configuration file (which you will create later in this chapter).

[Example 10–6](#) illustrates a possible implementation for the `client-cache-config.xml` file.

**Example 10–6 Sample Extend Client Cache Configuration File**

```
<?xml version="1.0"?>
<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-
cache-config coherence-cache-config.xsd">
  <defaults>
```

```
<serializer>pof</serializer>
</defaults>

<caching-scheme-mapping>
  <cache-mapping>
    <cache-name>security</cache-name>
    <scheme-name>examples-remote</scheme-name>
  </cache-mapping>
</caching-scheme-mapping>

<caching-schemes>
  <remote-cache-scheme>
    <scheme-name>examples-remote</scheme-name>
    <service-name>ExtendTcpCacheService</service-name>
    <initiator-config>
      <tcp-initiator>
        <remote-addresses>
          <socket-address>
            <address system-property="tangosol.coherence.proxy.
address">localhost</address>
            <port system-property="tangosol.coherence.proxy.port">9099</port>
          </socket-address>
        </remote-addresses>
      </tcp-initiator>
    </initiator-config>
  </remote-cache-scheme>

  <remote-invocation-scheme>
    <scheme-name>remote-invocation-scheme</scheme-name>
    <service-name>ExtendTcpInvocationService</service-name>
    <initiator-config>
      <tcp-initiator>
        <remote-addresses>
          <socket-address>
            <address system-property="tangosol.coherence.proxy.
address">localhost</address>
            <port system-property="tangosol.coherence.proxy.port">9099</port>
          </socket-address>
        </remote-addresses>
        <connect-timeout>2s</connect-timeout>
      </tcp-initiator>
      <outgoing-message-handler>
        <request-timeout>5s</request-timeout>
      </outgoing-message-handler>
    </initiator-config>
  </remote-invocation-scheme>
</caching-schemes>
</cache-config>
```

## Create a Cache Configuration File for the Extend Proxy

To create a cache configuration file for the extend proxy service:

1. Open the `coherence-cache-config.xml` file from the Project explorer window. You can find the file under `Security/appClientModule`.
2. Save the file as `examples-cache-config.xml`.
3. Configure the extend proxy cache configuration file. The following list highlights some key elements:

- Use the `cache-name` element to define `security` as the name of the cache. Note that there must be a cache defined in the extend client cache configuration that is also named `security`.
- Use the `address` and `port` elements in the `acceptor-config` stanza to identify the extend proxy service that is listening on the `localhost` address at port `9099`
- Use the `autostart` element with the `tangosol.coherence.extend.enabled` system property to prevent the cache server from running a proxy service.
- Use `defaults` and `serializer` with a value of `pof` to call the serializer for the custom POF configuration file (which you will create later in this chapter)

[Example 10-7](#) illustrates a possible implementation for the `examples-cache-config.xml` file.

#### **Example 10-7 Sample Cache Configuration File for the Proxy Server**

```
<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
              xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-
cache-config coherence-cache-config.xsd">
  <defaults>
    <serializer>pof</serializer>
  </defaults>

  <caching-scheme-mapping>
    <cache-mapping>
      <cache-name>security</cache-name>
      <scheme-name>ExamplesPartitionedPofScheme</scheme-name>
    </cache-mapping>
  </caching-scheme-mapping>

  <caching-schemes>
    <distributed-scheme>
      <scheme-name>ExamplesPartitionedPofScheme</scheme-name>
      <service-name>PartitionedPofCache</service-name>
      <backing-map-scheme>
        <local-scheme>
          <!-- each node will be limited to 32MB -->
          <high-units>32M</high-units>
          <unit-calculator>binary</unit-calculator>
        </local-scheme>
      </backing-map-scheme>
      <autostart>true</autostart>
    </distributed-scheme>

    <!--
    Proxy Service scheme that allows remote clients to connect to the
    cluster over TCP/IP.
    -->
    <proxy-scheme>
      <scheme-name>secure-proxy</scheme-name>
      <service-name>ProxyService</service-name>

      <thread-count system-property="tangosol.coherence.extend.threads">2</thread-
count>
```

```

    <acceptor-config>
      <tcp-acceptor>
        <local-address>
          <address system-property="tangosol.coherence.extend.
address">localhost</address>
          <port system-property="tangosol.coherence.extend.port">9099</port>
        </local-address>
      </tcp-acceptor>
    </acceptor-config>

    <autostart>true</autostart>
  </proxy-scheme>
</caching-schemes>
</cache-config>

```

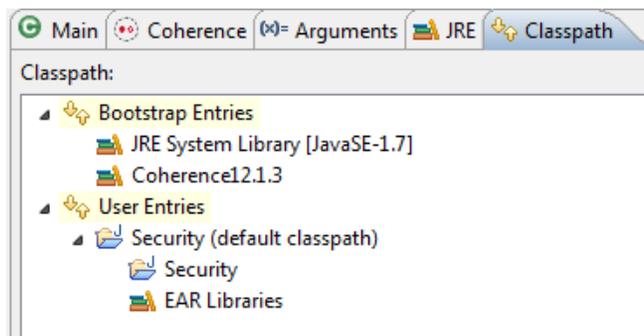
## Create a Start-Up Configuration for a Cache Server

Create a start-up configuration for a cache server cluster node. The configuration must include the system properties to designate a proxy service and the cluster-side cache configuration file. You must also include the application class files and the XML configuration files on the class path.

To create a start-up file for a cache server:

1. Create a start-up configuration in Eclipse. Right-click the `Security` project and select **Run As** then **Run Configurations**. In the Name field, enter `SecurityCacheServer`.
2. In the **Main** tab, click **Browse** in the **Project** field to select the `Security` project. Select the **Include system libraries when searching for a main class** and click the **Search** button in the **Main class** field. Enter `DefaultCacheServer` in the **Select type** field of the **Select Main Type** dialog box. Select **DefaultCacheServer - com.tangosol.net** and click **OK**. Click **Apply**.
3. In the **Coherence** tab, enter the name and absolute path to the cluster-side cache configuration file (in this case, `C:\home\oracle\workspace\Security\appClientModule\examples-cache-config.xml`). Select **Enabled (cache server)** in the **Local storage** field. Enter a unique value, such as `3155`, in the **Cluster port** field. Click **Apply**.
4. The **Classpath** tab should look similar to [Figure 10–1](#).

**Figure 10–1** Class Path for the Security Cache Server



5. In the **Common** tab, click **Browse** in the **Shared file** field to select the `Security` project. Click **Apply**.

## Create a Start-Up Configuration for a Cache Server with a Proxy Service

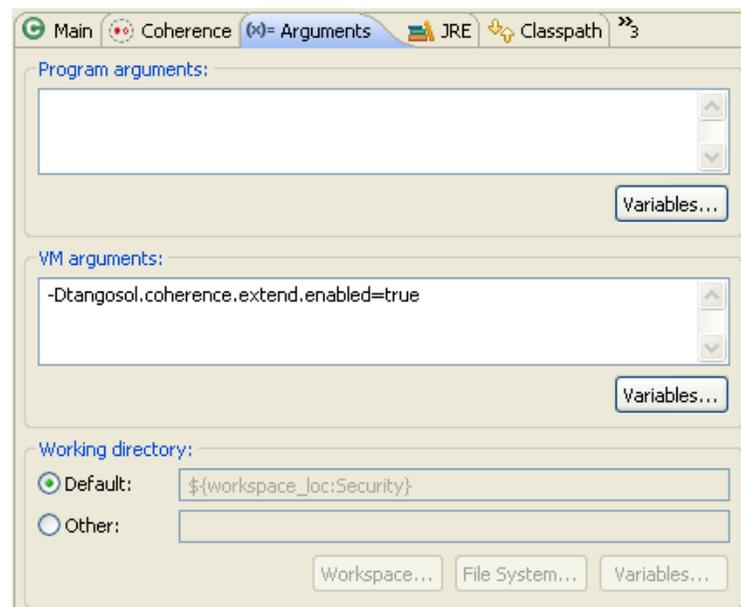
Create a configuration to start an extend proxy service on a cache server in the cluster. The extend client connects to this service. The configuration must include the system properties to designate a proxy service and the cluster-side cache configuration file. You must also include the application class files and the XML configuration files on the class path.

For these examples, the cache server with proxy service start-up configuration will have the same configuration as the cache server start-up configuration, but it will include the system property to enable the extend proxy.

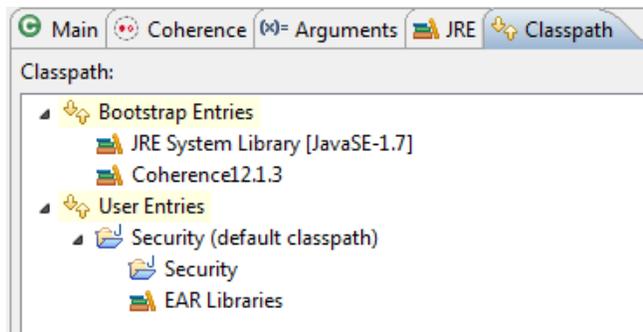
To create a start-up file for a cache server with a proxy service:

1. Create a start-up configuration in Eclipse. Right-click the `Security` project and select **Run As then Run Configurations**. In the Name field, enter `SecurityRunProxy`.
2. In the **Main** tab, click **Browse** in the **Project** field to select the `Security` project. Select the **Include system libraries when searching for a main class** and click the **Search** button in the **Main class** field. Enter `DefaultCacheServer` in the **Select type** field of the **Select Main Type** dialog box. Select **DefaultCacheServer - com.tangosol.net** and click **OK**. Click **Apply**.
3. In the **Coherence** tab, enter the name and absolute path to the cluster-side cache configuration file (in this case, `C:\home\oracle\workspace\Security\appClientModule\examples-cache-config.xml`). Select **Enabled (cache server)** in the **Local storage** field. Enter a unique value (such as `3155`) in the **Cluster port** field. Click **Apply**.
4. In the **Arguments** tab, enter the system property `-Dtangosol.coherence.extend.enabled=true` to designate a proxy service in the **VM arguments** field.

**Figure 10–2 Arguments Tab for the Security Proxy Server**



5. The **Classpath** tab should look similar to [Figure 10–3](#).

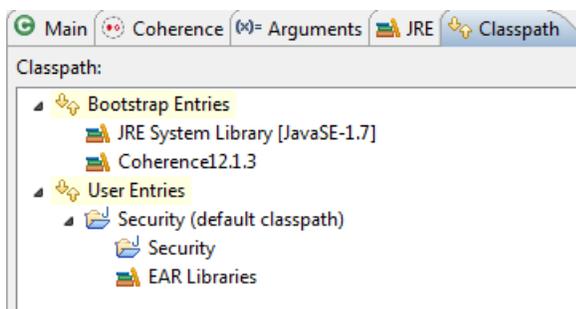
**Figure 10–3 Class Path for the Proxy Server**

6. In the **Common** tab, click **Browse** in the **Shared file** field to select the **Security** project. Click **Apply**.

## Run the Password Example

Run the password example to generate and validate the token, and pass it to the proxy service.

1. Create a run configuration for the `PasswordExample.java` file.
  - a. Right click `PasswordExample.java` in the **Project Explorer**, and select **Run As** then **Run Configurations**.
  - b. Click the **New launch configuration** icon. Ensure that `PasswordExample` appears in the **Name** field, `Security` appears in the **Project** field, and `com.oracle.handson.PasswordExample` appears in the **Main class** field. Click **Apply**.
  - c. In the **Coherence** tab, enter the path to the client cache configuration file, `C:\home\oracle\workspace\Security\appClientModule\client-cache-config.xml` in the **Cache configuration descriptor** field. Select **Disabled (cache client)** in the **Local cache** field. Enter a unique value, such as 3155, in the **Cluster port** field.
  - d. The **Classpath** tab should look similar to [Figure 10–4](#).

**Figure 10–4 Classpath Tab for the PasswordExample Program**

- e. In the **Common** tab, click **Browse** in the **Shared file** field to select the **Security** project. Click **Apply**.
2. Stop any running cache servers. See "[Stopping Cache Servers](#)" on page 2-14 for more information.

3. Run the security proxy server, the security cache server then the `PasswordExample.java` program.
  - a. Right click the project and select **Run As** then **Run Configurations**. Run the `SecurityRunProxy` configuration from the **Run Configurations** dialog box.
  - b. Right click the project and select **Run As** then **Run Configurations**. Run the `SecurityCacheServer` configuration from the **Run Configurations** dialog box.
  - c. Right click the `PasswordExample.java` file in the **Project Explorer** and select **Run As** then **Run Configurations**. Select `PasswordExample` in the **Run Configurations** dialog box and click **Run**.

The output of the `PasswordExample` program should be similar to [Example 10–8](#) in the Eclipse console. It indicates the start of the password example, the opening of the socket to the proxy server, and the completion of the example.

### **Example 10–8 Password Example Output in the Eclipse Console**

```
-----password example begins-----
2014-01-09 11:05:04.690/0.380 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/OracleCoherence/coherence/lib/coherence.jar!/tangosol-
coherence.xml"
2014-01-09 11:05:04.800/0.490 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/OracleCoherence/coherence/lib/coherence.jar!/tangosol-
coherence-override-dev.xml"
...
Oracle Coherence Version 12.1.3.0.0 Build 49331
  Grid Edition: Development mode
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

...
2014-01-09 11:05:06.472/2.162 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=main, member=n/a):
Connecting Socket to 10.159.156.144:9099
2014-01-09 11:05:06.473/2.163 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Connected Socket to 10.159.156.144:9099
-----password example succeeded-----
-----password example completed-----
```

The response in the proxy server shell should be similar to [Example 10–9](#). It lists the CN and OU values from the distinguished name and whether the password was validated.

### **Example 10–9 Response from the Cache Server Running the Proxy Service Shell**

```
...
Started DefaultCacheServer...

2014-01-09 11:04:35.313/34.291 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2014-01-09 11:04:35.113, Address=10.159.156.144:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:3896, Role=CoherenceServer) joined Cluster with senior
member 1
2014-01-09 11:04:35.405/34.383 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 joined Service Management with senior member 1
2014-01-09 11:04:36.175/35.153 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): Member 2 joined Service
PartitionedPofCache with senior member 1
2014-01-09 11:04:37.243/36.221 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): Transferring 129B of backup[1] for
PartitionSet{128..256} to member 2
```

```
2014-01-09 11:04:37.409/36.387 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=1): Transferring primary PartitionSet{0..127}
to member 2 requesting 128
Password validated for user: CN=BuckarooBanzai,OU=Yoyodyne
Password validated for user: CN=BuckarooBanzai,OU=Yoyodyne
Password validated for user: CN=BuckarooBanzai,OU=Yoyodyne
```

## Including Role-Based Access Control to the Cluster

This section describes how to create an example that uses role-based policies to access the cluster. The code logs in to get a subject with a user ID assigned to a particular role. It gets a cache reference running in the context of the subject and then attempt various cache operations. Depending on the role granted to the user, the cache operation is allowed or denied. Note that the role mapping and role-based authorization in the example is simplified and not intended for real security use.

For example, a user with a writer role can use the `put` and `get` methods. A user with a reader role can use the `get` method, but not the `put` method. A user with a writer role cannot destroy a cache; however, a user with an admin role can.

Note that when the cache reference is created in the context of a subject that identity is permanently associated with that reference. Any use of that cache reference is on behalf of that identity.

The example will use the `PasswordIdentityTransformer` and `PasswordIdentityAsserter` classes that you created in the previous section. The `PasswordIdentityTransformer` class generates a security token that contains the password, the user ID, and the roles. The `PasswordIdentityAsserter` class (running in the proxy) validates the security token to enforce the password and construct a subject with the proper user ID and roles. The production and assertion of the security token happens automatically.

To create the example:

1. [Define Which User Roles Are Entitled to Access Cache Methods](#)
2. [Apply the Entitlements to the Cache Service](#)
3. [Create the Access Control Example Program](#)
4. [Edit the Cluster-Side Cache Configuration File](#)
5. [Run the Access Control Example](#)

### Define Which User Roles Are Entitled to Access Cache Methods

Create a Java file that enables access to cache methods on the basis of a user's role. To do this, you can apply access permissions to a wrapped `NamedCache` using the `Subject` object passed from the client by using `Coherence*Extend`. The implementation allows only clients with a specified role to access the wrapped `NamedCache`.

The class that you create in this section extends the `com.tangosol.net.cache.WrapperNamedCache` class. This class is a convenience function that enables you to secure the methods on the `NamedCache` interface.

To determine which user role can access a cache method, include a call to the `SecurityExampleHelper.checkAccess` method in each cache method's implementation. As the argument to `checkAccess`, provide the user role that is permitted to access the method. Close the implementation with a call to `super`.

For example, the following code indicates that users with the `admin` role can destroy the cache.

```
public void destroy()
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_ADMIN);
    super.destroy();
}
```

In this example, users with the `reader` role can call the `aggregate` method.

```
public Object aggregate(Filter filter, EntryAggregator agent)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.aggregate(filter, agent);
}
```

To create a file that determines which user role can call cache methods:

1. Create a new Java class named `EntitledNamedCache` in the Security project. See ["Creating a Java Class"](#) on page 2-11 if you need detailed information.
2. Ensure that the class imports and extends `WrapperNamedCache`.
3. Import the `Filter`, `MapListener`, `ValueExtractor`, `Collection`, `Comparator`, `Map`, `Service`, and `Set` classes. The methods in `WrapperNamedCache` (and by extension, `EntitledNamedCache`) use arguments with these types.
4. Implement the methods in `EntitledNamedCache` such that only a user with a specific role can call the method.

[Example 10–10](#) illustrates a possible implementation of `EntitledNamedCache`.

#### **Example 10–10 Entitled Named Cache**

```
package com.oracle.handson;

import com.tangosol.net.NamedCache;
import com.tangosol.net.security.SecurityHelper;

import com.tangosol.net.Service;
import com.tangosol.net.cache.WrapperNamedCache;

import com.tangosol.util.Filter;
import com.tangosol.util.MapEvent;
import com.tangosol.util.MapListener;
import com.tangosol.util.ValueExtractor;

import java.util.Collection;
import java.util.Comparator;
import java.util.Map;
import java.util.Set;

import javax.security.auth.Subject;

/**
 * Example WrapperNamedCache that demonstrates how entitlements can be applied
 * to a wrapped NamedCache using the Subject passed from the client through
 * Coherence*Extend. This implementation only allows clients with a specified
 * role to access the wrapped NamedCache.
 */
```

```
*
*/
public class EntitledNamedCache
    extends WrapperNamedCache
    {
    /**
    * Create a new EntitledNamedCache.
    *
    * @param cache the wrapped NamedCache
    */
    public EntitledNamedCache(NamedCache cache)
        {
        super(cache, cache.getCacheName());
        }

    // ----- NamedCache interface -----

    /**
    * {@inheritDoc}
    */
    public void release()
        {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        super.release();
        }

    /**
    * {@inheritDoc}
    */
    public void destroy()
        {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_ADMIN);
        super.destroy();
        }

    /**
    * {@inheritDoc}
    */
    public Object put(Object oKey, Object oValue, long cMillis)
        {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        return super.put(oKey, oValue, cMillis);
        }

    /**
    * {@inheritDoc}
    */
    public void addMapListener(MapListener listener)
        {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        super.addMapListener(new EntitledMapListener(listener));
        }

    /**
    * {@inheritDoc}
    */
    public void removeMapListener(MapListener listener)
        {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        }
    }
}
```

```
        super.removeMapListener(listener);
    }

    /**
     * {@inheritDoc}
     */
    public void addMapListener(MapListener listener, Object oKey, boolean fLite)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        super.addMapListener(new EntitledMapListener(listener), oKey, fLite);
    }

    /**
     * {@inheritDoc}
     */
    public void removeMapListener(MapListener listener, Object oKey)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        super.removeMapListener(listener, oKey);
    }

    /**
     * {@inheritDoc}
     */
    public void addMapListener(MapListener listener, Filter filter, boolean fLite)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        super.addMapListener(new EntitledMapListener(listener), filter, fLite);
    }

    /**
     * {@inheritDoc}
     */
    public void removeMapListener(MapListener listener, Filter filter)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        super.removeMapListener(listener, filter);
    }

    /**
     * {@inheritDoc}
     */
    public int size()
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        return super.size();
    }

    /**
     * {@inheritDoc}
     */
    public void clear()
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        super.clear();
    }

    /**
     * {@inheritDoc}
     */

```

```
public boolean isEmpty()
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.isEmpty();
}

/**
 * {@inheritDoc}
 */
public boolean containsKey(Object oKey)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.containsKey(oKey);
}

/**
 * {@inheritDoc}
 */
public boolean containsValue(Object oValue)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.containsValue(oValue);
}

/**
 * {@inheritDoc}
 */
public Collection values()
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.values();
}

/**
 * {@inheritDoc}
 */
public void putAll(Map map)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    super.putAll(map);
}

/**
 * {@inheritDoc}
 */
public Set entrySet()
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.entrySet();
}

/**
 * {@inheritDoc}
 */
public Set keySet()
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.keySet();
}
```

```
/**
 * {@inheritDoc}
 */
public Object get(Object oKey)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.get(oKey);
}

/**
 * {@inheritDoc}
 */
public Object remove(Object oKey)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    return super.remove(oKey);
}

/**
 * {@inheritDoc}
 */
public Object put(Object oKey, Object oValue)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    return super.put(oKey, oValue);
}

/**
 * {@inheritDoc}
 */
public Map getAll(Collection colKeys)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.getAll(colKeys);
}

/**
 * {@inheritDoc}
 */
public boolean lock(Object oKey, long cWait)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    return super.lock(oKey, cWait);
}

/**
 * {@inheritDoc}
 */
public boolean lock(Object oKey)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    return super.lock(oKey);
}

/**
 * {@inheritDoc}
 */
public boolean unlock(Object oKey)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
```

```
        return super.unlock(oKey);
    }

    /**
     * {@inheritDoc}
     */
    public Set keySet(Filter filter)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        return super.keySet(filter);
    }

    /**
     * {@inheritDoc}
     */
    public Set entrySet(Filter filter)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        return super.entrySet(filter);
    }

    /**
     * {@inheritDoc}
     */
    public Set entrySet(Filter filter, Comparator comparator)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        return super.entrySet(filter, comparator);
    }

    /**
     * {@inheritDoc}
     */
    public void addIndex(ValueExtractor extractor, boolean fOrdered, Comparator
comparator)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        super.addIndex(extractor, fOrdered, comparator);
    }

    /**
     * {@inheritDoc}
     */
    public void removeIndex(ValueExtractor extractor)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        super.removeIndex(extractor);
    }

    /**
     * {@inheritDoc}
     */
    public Object invoke(Object oKey, EntryProcessor agent)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        return super.invoke(oKey, agent);
    }

    /**
     * {@inheritDoc}
     */
```

```

*/
public Map invokeAll(Collection collKeys, EntryProcessor agent)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    return super.invokeAll(collKeys, agent);
}

/**
 * {@inheritDoc}
 */
public Map invokeAll(Filter filter, EntryProcessor agent)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    return super.invokeAll(filter, agent);
}

/**
 * {@inheritDoc}
 */
public Object aggregate(Collection collKeys, EntryAggregator agent)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.aggregate(collKeys, agent);
}

/**
 * {@inheritDoc}
 */
public Object aggregate(Filter filter, EntryAggregator agent)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
    return super.aggregate(filter, agent);
}

// ----- inner class -----

/**
 * Example MapListener that adds authorization to map events.
 */
public class EntitledMapListener
    implements MapListener
{
    // ----- constructors -----

    /**
     * Construct an EntitledMapListener with the current subject.
     * The subject will not be available in the security context
     * when events are received by the proxy at runtime.
     *
     * @param listener the MapListener
     */
    public EntitledMapListener(MapListener listener)
    {
        m_listener = listener;
        m_subject = SecurityHelper.getCurrentSubject();
    }

    // ----- MapListener interface -----

```

```
/**
 * {@inheritDoc}
 */
public void entryInserted(MapEvent mapEvent)
{
    try
    {
        SecurityExampleHelper.checkAccess(
            SecurityExampleHelper.ROLE_WRITER, m_subject);
    }
    catch (SecurityException e)
    {
        System.out.println("Access denied for entryInserted");
        return;
    }
    m_listener.entryInserted(mapEvent);
}

/**
 * {@inheritDoc}
 */
public void entryUpdated(MapEvent mapEvent)
{
    try
    {
        SecurityExampleHelper.checkAccess(
            SecurityExampleHelper.ROLE_WRITER, m_subject);
    }
    catch (SecurityException e)
    {
        System.out.println("Access denied for entryUpdated");
        return;
    }
    m_listener.entryUpdated(mapEvent);
}

/**
 * {@inheritDoc}
 */
public void entryDeleted(MapEvent mapEvent)
{
    try
    {
        SecurityExampleHelper.checkAccess(
            SecurityExampleHelper.ROLE_WRITER, m_subject);
    }
    catch (SecurityException e)
    {
        System.out.println("Access denied for entryDeleted");
        return;
    }
    m_listener.entryDeleted(mapEvent);
}

// ----- data members -----
```

```

    /**
     * Subject from security context when the MapListener was registered
     */
    private Subject m_subject;

    /**
     * Registered listener
     */
    private MapListener m_listener;
}

// ----- helper methods -----

/**
 * Return the wrapped NamedCache.
 *
 * @return the wrapped CacheService
 */
public NamedCache getNamedCache()
{
    return (NamedCache) getMap();
}
}

```

## Apply the Entitlements to the Cache Service

Create a file that demonstrates how access entitlements can be applied to a wrapped `CacheService` using the `Subject` passed from the client through `Coherence*Extend`. The implementation delegates access control for cache operations to the `EntitledNamedCache` you created in the previous section.

The class that you create extends the `com.tangosol.net WrapperCacheService` class. This is a convenience function that allows you to secure the methods on the `CacheService`. It also provides a mechanism to delegate between the cache service on the proxy and the client request.

Implement the methods `ensureCache`, `releaseCache`, and `destroyCache` to ensure that only users with specific roles can use them. In the implementations, include a call to the `SecurityExampleHelper.checkAccess` method with a specific user role as its argument. For example, the following code ensures that only users with the role `admin` can destroy the cache.

```

public void destroyCache(NamedCache map)
{
    if (map instanceof EntitledNamedCache)
    {
        EntitledNamedCache cache = (EntitledNamedCache) map;
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_ADMIN);
        map = cache.getNamedCache();
    }
    super.destroyCache(map);
}

```

To create a file that applies entitlements to access the cache service:

1. Create a new Java class named `EntitledCacheService` in the `Security` project.
2. Ensure that the class imports and extends the `WrapperCacheService` class.

3. Implement the `ensureCache`, `releaseCache`, and `destroyCache` methods to ensure that only users with specific roles can use them.

[Example 10–11](#) illustrates a possible implementation of `EntitledCacheService`.

**Example 10–11 Entitled Cache Service**

```
package com.oracle.handson;

import com.tangosol.net.CacheService;
import com.tangosol.net.NamedCache;
import com.tangosol.net WrapperCacheService;

/**
 * Example WrapperCacheService that demonstrates how entitlements can be
 * applied to a wrapped CacheService using the Subject passed from the
 * client through Coherence*Extend. This implementation delegates access control
 * for cache operations to the EntitledNamedCache.
 */
public class EntitledCacheService
    extends WrapperCacheService
    {
    /**
     * Create a new EntitledCacheService.
     *
     * @param service    the wrapped CacheService
     */
    public EntitledCacheService(CacheService service)
    {
        super(service);
    }

    // ----- CacheService interface -----

    /**
     * {@inheritDoc}
     */
    public NamedCache ensureCache(String sName, ClassLoader loader)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
        return new EntitledNamedCache(super.ensureCache(sName, loader));
    }

    /**
     * {@inheritDoc}
     */
    public void releaseCache(NamedCache map)
    {
        if (map instanceof EntitledNamedCache)
        {
            EntitledNamedCache cache = (EntitledNamedCache) map;
            SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_READER);
            map = cache.getNamedCache();
        }
        super.releaseCache(map);
    }

    /**
     * {@inheritDoc}
     */
}
```

```

*/
public void destroyCache(NamedCache map)
{
    if (map instanceof EntitledNamedCache)
    {
        EntitledNamedCache cache = (EntitledNamedCache) map;
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_ADMIN);
        map = cache.getNamedCache();
    }
    super.destroyCache(map);
}
}

```

## Create the Access Control Example Program

Create a file to run the access control example. The role policies are defined in the `SecurityExampleHelper` class. The `EntitledCacheService` and `EntitledNamedCache` classes enforce the policies.

The program should specify various users as arguments to the `SecurityHelperFile.login` method, and then attempt to perform cache read, write, and destroy operations. Based on the entitlement policies defined in the `EntitledCacheService` and `EntitledNamedCache` classes, the operations succeed or fail.

1. Create a new Java class with a main method in the `Security` project named `AccessControlExample`.  
See "[Creating a Java Class](#)" on page 2-11 for detailed information.
2. Implement the main method to access the cache.
3. Specify users defined in the `SecurityExampleHelper` file as arguments to its `login` method.
4. For each user, execute read (`get`), write (`put`), and destroy operations on the cache and provide success or failure messages in response.

[Example 10–12](#) illustrates a possible implementation of the `AccessControlExample.java` class.

### **Example 10–12 Sample Program to Run the Access Control Example**

```

package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.InvocationService;
import com.tangosol.net.NamedCache;
import com.tangosol.net.Service;

import com.tangosol.util.MapEvent;
import com.tangosol.util.MapListener;

import java.security.PrivilegedExceptionAction;

import javax.security.auth.Subject;

/**
 * This class demonstrates simplified role based access control.
 * <p>
 * The role policies are defined in SecurityExampleHelper. Enforcement
 * is done by EntitledCacheService and EntitledNamedCache.

```

```

*
*/
public class AccessControlExample
{
    // ----- static methods -----

    public static void main (String[] args){
        accessCache();
    }

    /**
    * Demonstrate role based access to the cache.
    */
    public static void accessCache()
    {
        System.out.println("-----cache access control example begins-----");

        Subject subject = SecurityExampleHelper.login("JohnWhorfin");

        // Someone with writer role can write and read
        try
        {
            NamedCache cache = (NamedCache) Subject.doAs(
                subject, new PrivilegedExceptionAction()
                {
                    public Object run()
                        throws Exception
                    {
                        return CacheFactory.getCache(SecurityExampleHelper.SECURITY_
CACHE_NAME);
                    }
                });
            cache.put("myKey", "myValue");
            cache.get("myKey");
            System.out.println("    Success: read and write allowed");
        }
        catch (Exception e)
        {
            // get exception if not allowed to perform the operation
            e.printStackTrace();
        }

        // Someone with reader role can read but not write
        subject = SecurityExampleHelper.login("JohnBigboote");
        try
        {
            NamedCache cache = (NamedCache) Subject.doAs(
                subject, new PrivilegedExceptionAction()
                {
                    public Object run()
                        throws Exception
                    {
                        return CacheFactory.getCache(SecurityExampleHelper.SECURITY_
CACHE_NAME);
                    }
                });
            cache.get("myKey");
            System.out.println("    Success: read allowed");
            cache.put("anotherKey", "anotherValue");
        }
        catch (Exception e)

```

```

        {
            // get exception if not allowed to perform the operation
            System.out.println("    Success: Correctly cannot write");
        }

// Someone with writer role cannot call destroy
subject = SecurityExampleHelper.login("JohnWhorfin");
try
{
    {
        NamedCache cache = (NamedCache) Subject.doAs(
            subject, new PrivilegedExceptionAction()
            {
                public Object run()
                    throws Exception
                {
                    return CacheFactory.getCache(SecurityExampleHelper.SECURITY_
CACHE_NAME);
                }
            });
        cache.destroy();
    }
    catch (Exception e)
    {
        // get exception if not allowed to perform the operation
        System.out.println("    Success: Correctly cannot " +
            "destroy the cache");
    }

// Someone with admin role can call destroy
subject = SecurityExampleHelper.login("BuckarooBanzai");
try
{
    {
        NamedCache cache = (NamedCache) Subject.doAs(
            subject, new PrivilegedExceptionAction()
            {
                public Object run()
                    throws Exception
                {
                    return CacheFactory.getCache(SecurityExampleHelper.SECURITY_
CACHE_NAME);
                }
            });
        cache.destroy();
        System.out.println("    Success: Correctly allowed to " +
            "destroy the cache");
    }
    catch (Exception e)
    {
        // get exception if not allowed to perform the operation
        e.printStackTrace();
    }
    System.out.println("-----cache access control example completed-----");
}
}
}

```

## Edit the Cluster-Side Cache Configuration File

Edit the cluster-side cache configuration file `examples-cache-config.xml`. Specify the full path of the class name of the cache service in the `cache-service-proxy`

stanza under `proxy-config`. The `cache-service-proxy` stanza contains the configuration information for a cache service proxy managed by a proxy service.

In this case, the cache service class name is `com.oracle.handson.EntitledCacheService` proxy and the `param-type` is `com.tangosol.net.CacheService`.

[Example 10-13](#) illustrates the XML code to add to the configuration.

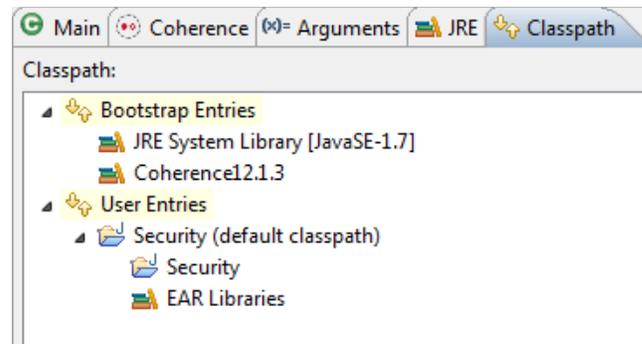
**Example 10-13 Cache Service Proxy Configuration for a Cluster-Side Cache Configuration**

```
...
<proxy-config>
  <cache-service-proxy>
    <class-name>com.oracle.handson.EntitledCacheService</class-name>
    <init-params>
      <init-param>
        <param-type>com.tangosol.net.CacheService</param-type>
        <param-value>{service}</param-value>
      </init-param>
    </init-params>
  </cache-service-proxy>
</proxy-config>
...
```

## Run the Access Control Example

Run the access control example to demonstrate how access to the cache can be granted or denied, based on a user's role.

1. Create a run configuration for the `AccessControlExample.java`.
  - a. Right click `AccessControlExample` in the **Project Explorer**. Select **Run As** then **Run Configurations**.
  - b. Click **Oracle Coherence** then **New launch configuration** icon. Ensure that `AccessControlExample` appears in the **Name** field, `Security` appears in the **Project** field, and `com.oracle.handson.AccessControlExample` appears in the **Main class** field. Click **Apply**.
  - c. In the **Coherence** tab, enter the path to the `client-cache-config.xml` file in the **Cache configuration descriptor** field. Select **Disabled (cache client)** in the **Local storage** field. Enter `3155` in the **Cluster port** field.
  - d. In the **Classpath** tab, ensure that **User Entries** looks similar to [Figure 10-5](#). Click **Apply** then **Close**.

**Figure 10–5 Classpath Tab for the AccessControlExample Program**

- e. In the **Common** tab, click **Browse** in the **Shared file** field to select the **Security** project. Click **Apply**.
2. Stop any running cache servers. See "[Stopping Cache Servers](#)" on page 2-14 for more information.
3. Run the security proxy server, the security cache server then the `AccessControlExample.java` program.
  - a. Right click the project and select **Run As** then **Run Configurations**. Run the `SecurityRunProxy` configuration from the **Run Configurations** dialog box.
  - b. Right click the project and select **Run As** then **Run Configurations**. Run the `SecurityCacheServer` configuration from the **Run Configurations** dialog box.
  - c. Right click the `AccessControlExample.java` file in the **Project Explorer** and select **Run As** then **Run Configurations**. Run the `AccessControlExample` configuration from the **Run Configurations** dialog box.

The output is similar to [Example 10–14](#) in the Eclipse console. The messages correspond to the various users specified in `AccessControlExample` executing read, write, and destroy operations on the cache.

- The `Success: read and write allowed` message corresponds to the user with role `writer` attempting to read from and write to the cache
- The `Success: read allowed` message corresponds to the user with role `reader` attempting to read from the cache
- The `Success: Correctly cannot write` message corresponds to the user with role `reader` attempting to write to the cache
- The `Success: Correctly cannot destroy the cache` message corresponds to the user with role `writer` attempting to destroy the cache
- The `Success: Correctly allowed to destroy the cache` message corresponds to the user with role `admin` attempting to destroy the cache

#### **Example 10–14 Access Control Example Output in the Eclipse Console**

-----cache access control example begins-----

```
2014-01-16 13:15:02.802/0.413 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/OracleCoherence/coherence/lib/coherence.jar!/tangosol-
coherence.xml"
```

```

2014-01-16 13:15:02.901/0.512 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/OracleCoherence/coherence/lib/coherence.jar!/tangosol-
coherence-override-dev.xml"
...
2014-01-16 13:15:04.593/2.204 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Connected Socket to 130.35.99.90:9099
    Success: read and write allowed
2014-01-16 13:15:05.695/3.306 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=main, member=n/a):
Connecting Socket to 130.35.99.90:9099
2014-01-16 13:15:05.696/3.307 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Connected Socket to 130.35.99.90:9099
    Success: read allowed
    Success: Correctly cannot write
    Success: Correctly cannot destroy the cache
2014-01-16 13:15:06.178/3.789 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=main, member=n/a):
Connecting Socket to 130.35.99.90:9099
2014-01-16 13:15:06.179/3.790 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Connected Socket to 130.35.99.90:9099
    Success: Correctly allowed to destroy the cache
-----cache access control example completed-----

```

[Example 10–15](#) lists the output in the shell where the cache server is running the proxy service. Notice that the security exceptions in the output correspond to the **Success: Correctly cannot write** and **Success: Correctly cannot destroy the cache** messages in the Eclipse console.

#### **Example 10–15** Output for the Cache Server Running the Proxy Service

```

Started DefaultCacheServer...

2014-01-16 13:14:46.473/25.406 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2014-01-16 13:14:46.303, Address=130.35.99.90:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:6516, Role=CoherenceServer) joined Cluster with senior
member 1
2014-01-16 13:14:46.564/25.497 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 joined Service Management with senior member 1
2014-01-16 13:14:48.361/27.294 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
TcpRing disconnected from Member(Id=2, Timestamp=2014-01-16 13:14:46.303, Address=130.35.99.
90:8090, MachineId=47251, Location=site:,machine:TPFAEFFL-LAP,process:6516, Role=CoherenceServer)
due to a peer departure; removing the member.
2014-01-16 13:14:48.361/27.294 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster, member=1):
Member(Id=2, Timestamp=2014-01-16 13:14:48.361, Address=130.35.99.90:8090, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:6516, Role=CoherenceServer) left Cluster with senior
member 1
2014-01-16 13:14:48.362/27.295 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 left service Management with senior member 1
Password validated for user: role_writer
Password validated for user: role_writer
Password validated for user: role_writer
Password validated for user: role_reader
Password validated for user: role_reader
Password validated for user: role_reader
2014-01-16 13:15:05.706/44.639 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Proxy:ProxyService:TcpAcceptorWorker:0, member=1): An exception occurred while processing a
PutRequest for Service=Proxy:ProxyService:TcpAcceptor: java.lang.SecurityException: Access denied,
insufficient privileges
    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:105)
    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:59)
    at com.oracle.handson.EntitledNamedCache.put(EntitledNamedCache.java:68)

```

```

    at com.tangosol.coherence.component.net.extend.proxy.NamedCacheProxy.
put$Router(NamedCacheProxy.CDB:1)
    at com.tangosol.coherence.component.net.extend.proxy.NamedCacheProxy.put(NamedCacheProxy.CDB:2)
    at com.tangosol.coherence.component.net.extend.messageFactory.NamedCacheFactory$PutRequest.
onRun(NamedCacheFactory.CDB:6)
    at com.tangosol.coherence.component.net.extend.message.Request.run(Request.CDB:4)
    at com.tangosol.coherence.component.net.extend.proxy.NamedCacheProxy.onMessage(NamedCacheProxy.
CDB:11)
    at com.tangosol.coherence.component.net.extend.Channel$MessageAction.run(Channel.CDB:13)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:356)
    at com.tangosol.coherence.component.net.extend.Channel.execute(Channel.CDB:48)
    at com.tangosol.coherence.component.net.extend.Channel.receive(Channel.CDB:26)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.
Peer$DaemonPool$WrapperTask.run(Peer.CDB:9)
    at com.tangosol.coherence.component.util.DaemonPool$WrapperTask.run(DaemonPool.CDB:32)
    at com.tangosol.coherence.component.util.DaemonPool$Daemon.onNotify(DaemonPool.CDB:65)
    at com.tangosol.coherence.component.util.Daemon.run(Daemon.CDB:51)
    at java.lang.Thread.run(Thread.java:724)

```

2014-01-16 13:15:05.742/44.675 Oracle Coherence GE 12.1.3.0.0 <D5>

(thread=Proxy:ProxyService:TcpAcceptorWorker:1, member=1): An exception occurred while processing a DestroyCacheRequest for Service=Proxy:ProxyService:TcpAcceptor: java.lang.SecurityException: Access denied, insufficient privileges

```

    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:105)
    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:59)
    at com.oracle.handson.EntitledCacheService.destroyCache(EntitledCacheService.java:60)
    at com.tangosol.coherence.component.net.extend.messageFactory.
CacheServiceFactory$DestroyCacheRequest.onRun(CacheServiceFactory.CDB:6)
    at com.tangosol.coherence.component.net.extend.message.Request.run(Request.CDB:4)
    at com.tangosol.coherence.component.net.extend.proxy.serviceProxy.CacheServiceProxy.
onMessage(CacheServiceProxy.CDB:12)
    at com.tangosol.coherence.component.net.extend.Channel$MessageAction.run(Channel.CDB:13)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:356)
    at com.tangosol.coherence.component.net.extend.Channel.execute(Channel.CDB:48)
    at com.tangosol.coherence.component.net.extend.Channel.receive(Channel.CDB:26)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.
Peer$DaemonPool$WrapperTask.run(Peer.CDB:9)
    at com.tangosol.coherence.component.util.DaemonPool$WrapperTask.run(DaemonPool.CDB:32)
    at com.tangosol.coherence.component.util.DaemonPool$Daemon.onNotify(DaemonPool.CDB:65)
    at com.tangosol.coherence.component.util.Daemon.run(Daemon.CDB:51)
    at java.lang.Thread.run(Thread.java:724)

```

**Password validated for user: CN=BuckarooBanzai,OU=Yoyodyne**

**Password validated for user: CN=BuckarooBanzai,OU=Yoyodyne**

**Password validated for user: CN=BuckarooBanzai,OU=Yoyodyne**

2014-01-16 13:15:06.196/45.129 Oracle Coherence GE 12.1.3.0.0 <Error>

(thread=Proxy:ProxyService:TcpAcceptor, member=1): Error unregistering channel from receiver: NamedCacheProxy(NamedCache=security): java.lang.SecurityException: Access denied, insufficient privileges

```

    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:105)
    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:59)
    at com.oracle.handson.EntitledNamedCache.removeMapListener(EntitledNamedCache.java:86)
    at com.tangosol.coherence.component.net.extend.proxy.NamedCacheProxy.
unregisterChannel(NamedCacheProxy.CDB:21)

```

...

## Including Role-Based Access Control to an Invocable Object

An invocation service cluster service enables extend clients to execute invocable objects on the cluster. This example demonstrates how you can use role-based policies to determine which users can run the invocable objects.

For example, a user with a writer role can run an invocable object. A user with a reader role cannot.

In this example, you create a simple invocable object that can be called from a client program. Since the invocable object will be serializable, you must also list it in a POF configuration file. You also create an invocation service program that tests whether a user can execute methods on the service based on the user's role.

As in the previous example, this example uses the `PasswordIdentityTransformer` class to generate a security token that contains the password, the user ID, and the roles. The `PasswordIdentityAsserter` (running in the proxy) will be used to validate the security token to enforce the password and construct a subject with the proper user ID and roles. The production and assertion of the security token happens automatically.

To create the example:

1. [Create an Invocable Object](#)
2. [Create an Entitled Invocation Service](#)
3. [Create the Access Invocation Service Example Program](#)
4. [Edit the Cluster-Side Cache Configuration File](#)
5. [Create a POF Configuration File](#)
6. [Edit the Run Configurations for the Servers](#)
7. [Run the Access Invocation Service Example](#)

### Create an Invocable Object

Create an implementation of a simple invocable object that will be used by an entitled invocation service. For example, the invocable object can be written to increment and return an integer.

To create an invocable object:

1. Create a new Java class named `ExampleInvocable` in the `Security` project. See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Import the `Invocable` and `InvocationService` interfaces. Because this class will be working with serializable objects, import the `PortableObject`, `PofReader` and `PofWriter` classes.
3. Ensure that the `ExampleInvocable` class implements `Invocable` and `PortableObject`.
4. Implement the `ExampleInvocable` class to increment an integer and return the result.
5. Implement the `PofReader.readExternal` and `PofWriter.writeExternal` methods.

[Example 10-16](#) illustrates a possible implementation of `ExampleInvocable.java`.

**Example 10–16 A Sample Invocable Object**

```

package com.oracle.handson;

import com.tangosol.io.pof.PortableObject;
import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PofWriter;

import com.tangosol.net.Invocable;
import com.tangosol.net.InvocationService;

import java.io.IOException;

/**
 * Invocable implementation that increments and returns a given integer.
 */
public class ExampleInvocable
    implements Invocable, PortableObject
    {
    // ----- constructors -----

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    /**
     * Default constructor.
     */
    public ExampleInvocable()
        {
        }

    // ----- Invocable interface -----

    /**
     * {@inheritDoc}
     */
    public void init(InvocationService service)
        {
        m_service = service;
        }

    /**
     * {@inheritDoc}
     */
    public void run()
        {
        if (m_service != null)
            {
            m_nValue++;
            }
        }

    /**
     * {@inheritDoc}
     */
    public Object getResult()

```

```
        {
            return new Integer(m_nValue);
        }

// ----- PortableObject interface -----

/**
 * {@inheritDoc}
 */
public void readExternal(PofReader in)
    throws IOException
    {
        m_nValue = in.readInt(0);
    }

/**
 * {@inheritDoc}
 */
public void writeExternal(PofWriter out)
    throws IOException
    {
        out.writeInt(0, m_nValue);
    }

// ----- data members -----

/**
 * The integer value to increment.
 */
private int m_nValue;

/**
 * The InvocationService that is executing this Invocable.
 */
private transient InvocationService m_service;
}
```

## Create an Entitled Invocation Service

This example shows how a remote invocation service can be wrapped to provide access control. Access entitlements can be applied to a wrapped `InvocationService` using the `Subject` passed from the client by using `Coherence*Extend`. This implementation enables only clients with a specified role to access the wrapped invocation service.

The class that you create should extend the `com.tangosol.net.WrapperInvocationService` class. This is a convenience function that enables you to secure the methods on `InvocationService`. It also provides a mechanism to delegate between the invocation service on the proxy and the client request.

To create an entitled invocation service:

1. Create a new Java class named `EntitledInvocationService` in the Security project.
2. Import the `Invocable`, `InvocationObserver`, `InvocationService`, `WrapperInvocationService`, `Map`, and `Set` interfaces. Ensure that the

EntitledInvocationService class extends the WrapperInvocationService class.

3. Implement the query and execute methods. In the implementations, include a call to the SecurityExampleHelper.checkAccess method to determine whether a specified user role, in this case, ROLE\_WRITER, can access these operations.

Example 10-17 illustrates a possible implementation of EntitledInvocationService.java.

**Example 10-17 A Sample Entitled Invocation Service**

```
package com.oracle.handson;

import com.tangosol.net.Invocable;
import com.tangosol.net.InvocationObserver;
import com.tangosol.net.InvocationService;
import com.tangosol.net.WrapperInvocationService;

import java.util.Map;
import java.util.Set;

/**
 * Example WrapperInvocationService that demonstrates how entitlements can be
 * applied to a wrapped InvocationService using the Subject passed from the
 * client through Coherence*Extend. This implementation only allows clients with a
 * specified role to access the wrapped InvocationService.
 */
public class EntitledInvocationService
    extends WrapperInvocationService
    {
    /**
     * Create a new EntitledInvocationService.
     *
     * @param service the wrapped InvocationService
     */
    public EntitledInvocationService(InvocationService service)
    {
        super(service);
    }

    // ----- InvocationService interface -----

    /**
     * {@inheritDoc}
     */
    public void execute(Invocable task, Set setMembers, InvocationObserver
observer)
    {
        SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
        super.execute(task, setMembers, observer);
    }

    /**
     * {@inheritDoc}
     */
}
```

```
public Map query(Invocable task, Set setMembers)
{
    SecurityExampleHelper.checkAccess(SecurityExampleHelper.ROLE_WRITER);
    return super.query(task, setMembers);
}
}
```

## Create the Access Invocation Service Example Program

Create a program to run the access invocation service example. The objective of the program is to test whether various users defined in the `SecurityExampleHelper` class are able to access and run an invocable object. The enforcement of the role-based policies is provided by the `EntitledInvocationService` class.

To create a program to run the Access Invocation Service example:

1. Create a Java class with a main method in the `Security` project named `AccessInvocationServiceExample.java`.  
See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Among other classes, import `ExampleInvocable`, `CacheFactory`, and `InvocationService`.
3. Implement the main method to invoke the `accessInvocationService` method.
4. Implement the `accessInvocationService` class so that various users defined in the `SecurityExampleHelper` class attempt to log in to the service and run the object defined in `ExampleInvocable`. Use the `SecurityExampleHelper.login` method to test whether various users can access the invocable service.

[Example 10-18](#) illustrates a possible implementation of `AccessInvocationServiceExample.java`.

### **Example 10-18 Sample Program to Run the Access Invocation Service Example**

```
package com.oracle.handson;

import com.oracle.handson.ExampleInvocable;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.InvocationService;

import java.security.PrivilegedExceptionAction;

import javax.security.auth.Subject;

/**
 * This class demonstrates simplified role based access control for the
 * invocation service.
 * <p>
 * The role policies are defined in SecurityExampleHelper. Enforcement
 * is done by EntitledInvocationService.
 *
 */
public class AccessInvocationServiceExample
{
    /**
     * Invoke the example
     */
}
```

```

* @param asArg  command line arguments (ignored in this example)
*/
public static void main(String[] asArg)
{
    accessInvocationService();
}

/**
 * Access the invocation service
 */
public static void accessInvocationService()
{
    System.out.println("-----InvocationService access control example " +
        "begins-----");

    // Someone with writer role can run invocables
    Subject subject = SecurityExampleHelper.login("JohnWhorfin");

    try
    {
        InvocationService service = (InvocationService) Subject.doAs(
            subject, new PrivilegedExceptionAction()
            {
                public Object run()
                {
                    return CacheFactory.getService(
                        SecurityExampleHelper.INVOCATION_SERVICE_NAME);
                }
            });
        service.query(new ExampleInvocable(), null);
        System.out.println("    Success: Correctly allowed to " +
            "use the invocation service");
    }
    catch (Exception e)
    {
        // get exception if not allowed to perform the operation
        e.printStackTrace();
    }

    // Someone with reader role cannot cannot run invocables
    subject = SecurityExampleHelper.login("JohnBigboote");
    try
    {
        InvocationService service = (InvocationService) Subject.doAs(
            subject, new PrivilegedExceptionAction()
            {
                public Object run()
                {
                    return CacheFactory.getService(
                        SecurityExampleHelper.INVOCATION_SERVICE_NAME);
                }
            });
        service.query(new ExampleInvocable(), null);
    }
    catch (Exception ee)
    {
        System.out.println("    Success: Correctly unable to " +
            "use the invocation service");
    }
    System.out.println("-----InvocationService access control example " +

```

```

        "completed-----");
    }
}

```

## Edit the Cluster-Side Cache Configuration File

Edit the `examples-cache-config.xml` file to add the full path to the invocation service to the `invocation-service-proxy` stanza under the `proxy-config`. The `invocation-service-proxy` stanza contains the configuration information for an invocation service proxy managed by a proxy service.

In this case, the invocation service class name is `com.oracle.handson.EntitledInvocationService` and its `param-type` is `com.tangosol.net.InvocationService`.

### Example 10–19 Invocation Service Proxy Configuration for a Cluster-Side Cache

```

...
<proxy-config>
...
  <invocation-service-proxy>
    <class-name>com.oracle.handson.EntitledInvocationService</class-name>
    <init-params>
      <init-param>
        <param-type>com.tangosol.net.InvocationService</param-type>
        <param-value>{service}</param-value>
      </init-param>
    </init-params>
  </invocation-service-proxy>
</proxy-config>
...

```

## Create a POF Configuration File

Create a POF configuration file to declare `ExampleInvocable` as a user type.

1. Locate the `pof-config.xml` file under `Security\appClientModule` in the **Project Explorer** and open it in the Eclipse IDE.
2. Enter the code to declare `ExampleInvocable` as a user type and save the file.

The contents of the file should look similar to [Example 10–20](#). The file will be saved to the `C:\home\oracle\workspace\Security\appClientModule` folder.

### Example 10–20 POF Configuration File with ExampleInvocable User Type

```

<?xml version="1.0"?>
<pof-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://xmlns.oracle.com/coherence/coherence-pof-config"
  xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-pof-config
http://xmlns.oracle.com/coherence/coherence-pof-config/1.2/coherence-pof-config.
xsd">
  <user-type-list>
    <!-- include all "standard" Coherence POF user types -->
    <include>coherence-pof-config.xml</include>

    <!-- com.tangosol.examples package -->
    <user-type>
      <type-id>1007</type-id>
      <class-name>com.oracle.handson.ExampleInvocable</class-name>
    </user-type>
  </user-type-list>
</pof-config>

```

```
</user-type>

</user-type-list>

<allow-interfaces>true</allow-interfaces>
<allow-subclasses>true</allow-subclasses>
</pof-config>
```

## Edit the Run Configurations for the Servers

The classloader must encounter the custom POF configuration file (which must be named `pof-config.xml`) before it references the one in the `coherence.jar` file. If it does not, then the custom POF configuration file will be ignored and the default file in the `coherence.jar` file will be used instead.

To make the application use the XML configuration files in the `C:\home\oracle\workspace\Security\appModule`, ensure that the `Coherence12.1.3` library does not appear in the **Bootstrap Entries** section of the servers' class path. Also, position the `coherence.jar` file after the `Security` folder in the **User Entries** section.

If the `Coherence12.1.3` library does appear in the **Bootstrap Entries** section of the servers' class path, follow these steps to remove it:

1. Right click the project in the **Project Explorer** and select **Run As** then **Run Configurations**.
2. Select **SecurityRunProxy**. In the **Classpath** tab, *remove* **Coherence12.1.3** from the list of **Bootstrap Entries** if it appears. Click **Add External Jars** and select the `coherence.jar` file from the `Oracle_Home\coherence\lib` folder. Click **Apply**.
3. Select **SecurityCacheServer**. In the **Classpath** tab, *remove* **Coherence12.1.3** from the list of **Bootstrap Entries** if it appears. Click **Add External Jars** and select the `coherence.jar` file from the `Oracle_Home\coherence\lib` folder. Click **Apply**.

---

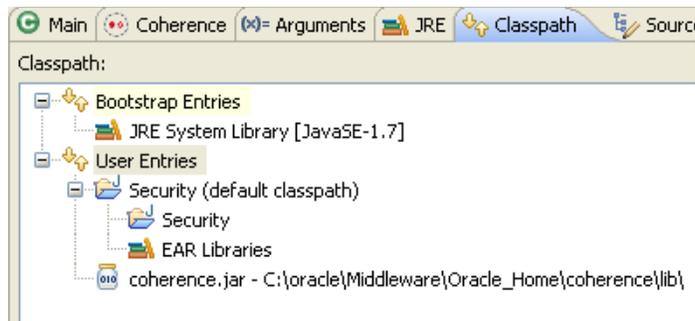
---

**Note:** Ensure that the XML configuration files (in the `C:\home\oracle\workspace\Security` and the `Security\build\classes` folders) appear before the `coherence.jar` file on the class path. The classloader must encounter the custom POF configuration file (which must be named `pof-config.xml`) before it references the one in the `coherence.jar` file.

---

---

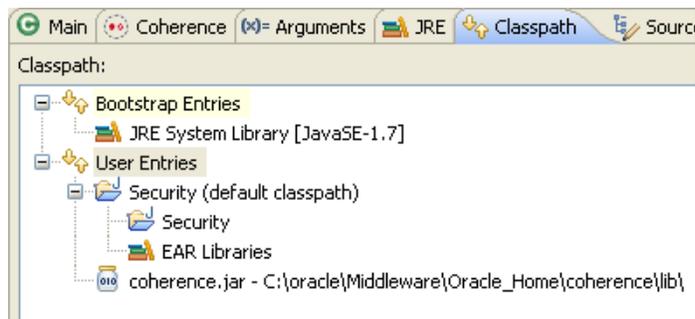
When you are finished, the **Classpath** tab for `SecurityRunProxy` and `SecurityCacheServer` should look similar to [Figure 10-6](#).

**Figure 10–6 Class Path for the Cache Server and the Server Running the Proxy Service**

## Run the Access Invocation Service Example

Run the access invocation service example to demonstrate how access to the invocable object can be granted or denied, based on a user's role.

1. Create a run configuration for the `AccessInvocationServiceExample.java` file.
  - a. Right click `AccessInvocationServiceExample.java` in the **Project Explorer** and select **Run As** then **Run Configurations**.
  - b. Click **Oracle Coherence**, then the **New launch configuration** icon. Ensure that `AccessInvocationServiceExample` appears in the **Name** field, `Security` appears in the **Project** field, and `com.oracle.handson.AccessInvocationServiceExample.java` appears in the **Main class** field. Click **Apply**.
  - c. In the **Coherence** tab, enter the path to the `client-cache-config.xml` file in the **Cache configuration descriptor** field. Select **Disabled (cache client)** in the **Local storage** field.
  - d. In the **Classpath** tab, *remove* **Coherence12.1.3** from the **Bootstrap Entries** list if it appears there. Click **Add External Jars** to add the `coherence.jar` file to **User Entries**. Move the `Security` folder to the top of **User Entries** followed by the `coherence.jar` file if it does not already appear there. When you are finished, the **Classpath** tab should look similar to [Figure 10–7](#). Click **Apply** then **Close**.

**Figure 10–7 Classpath Tab for the AccessInvocationServiceExample Program**

2. Stop any running cache servers. See "[Stopping Cache Servers](#)" on page 2-14 for more information.

3. Run the security proxy server, the security cache server then the `AccessInvocationServiceExample.java` program.
  - a. Right click the project and select **Run As** then **Run Configurations**. Run the `SecurityRunProxy` configuration from the **Run Configurations** dialog box.
  - b. Right click the project and select **Run As** then **Run Configurations**. Run the `SecurityCacheServer` configuration from the **Run Configurations** dialog box.
  - c. Right click the `AccessInvocationServiceExample.java` file in the **Project Explorer** and select **Run As** then **Run Configurations**. Select `AccessControlExample` in the **Run Configurations** dialog box and click **Run**.

The output is similar to [Example 10–21](#) in the Eclipse console. The messages correspond to the users specified in the `AccessInvocationServiceExample` class that are trying to run the invocable object `ExampleInvocable`.

- The message `Success: Correctly allowed to use the invocation service` corresponds to the user with role `writer` attempting to run `ExampleInvocable`.
- The message `Success: Correctly unable to use the invocation service` corresponds to the user with role `reader` attempting to run `ExampleInvocable`.

#### **Example 10–21 Client Program Response in the Eclipse Console**

```
-----InvocationService access control example begins-----
2014-01-16 14:20:34.509/0.370 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational configuration from "jar:file:/C:/OracleCoherence/coherence/lib/coherence.jar!/tangosol-
coherence.xml"
2014-01-16 14:20:34.609/0.470 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "jar:file:/C:/OracleCoherence/coherence/lib/coherence.jar!/tangosol-
coherence-override-dev.xml"
2014-01-16 14:20:34.689/0.550 Oracle Coherence 12.1.3.0.0 <Info> (thread=main, member=n/a): Loaded
operational overrides from "file:/C:/home/oracle/workspace_jan_
2014/Security/build/classes/tangosol-coherence-override.xml"
2014-01-16 14:20:34.699/0.560 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-config.xml" is not specified
2014-01-16 14:20:34.699/0.560 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "cache-factory-builder-config.xml" is not specified
2014-01-16 14:20:34.699/0.560 Oracle Coherence 12.1.3.0.0 <D5> (thread=main, member=n/a): Optional
configuration override "/custom-mbeans.xml" is not specified

Oracle Coherence Version 12.1.3.0.0 Build 49721
Grid Edition: Development mode
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

2014-01-16 14:20:35.039/0.900 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Loaded cache configuration from "file:/C:/home/oracle/workspace_jan_
2014/Security/appClientModule/client-cache-config.xml"
2014-01-16 14:20:35.699/1.560 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Created cache factory com.tangosol.net.ExtensibleConfigurableCacheFactory
2014-01-16 14:20:36.187/2.048 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=main, member=n/a):
Connecting Socket to 130.35.99.90:9099
2014-01-16 14:20:36.190/2.051 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Connected Socket to 130.35.99.90:9099
Success: Correctly allowed to use the invocation service
```

```

2014-01-16 14:20:37.094/2.955 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=main, member=n/a):
Connecting Socket to 130.35.99.90:9099
2014-01-16 14:20:37.094/2.955 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main, member=n/a):
Connected Socket to 130.35.99.90:9099
Success: Correctly unable to use the invocation service
-----InvocationService access control example completed-----

```

[Example 10–22](#) lists the output in the shell where the cache server is running the proxy service. Notice that the security exception in the output corresponds to the **Success: Correctly unable to use the invocation service** message in the Eclipse console, where the user with role reader attempts to run ExampleInvocable.

### **Example 10–22 Proxy Service Response in the Eclipse Console**

```

Started DefaultCacheServer...
...
2014-01-16 14:18:33.043/27.491 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Invocation:Management,
member=1): Member 2 left service Management with senior member 1
Password validated for user: role_writer
Password validated for user: role_writer
Password validated for user: role_reader
Password validated for user: role_reader
2014-01-16 14:20:37.094/151.542 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Proxy:ProxyService:TcpAcceptorWorker:1, member=1): An exception occurred while processing a
InvocationRequest for Service=Proxy:ProxyService:TcpAcceptor: java.lang.SecurityException: Access
denied, insufficient privileges
    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:105)
    at com.oracle.handson.SecurityExampleHelper.checkAccess(SecurityExampleHelper.java:59)
    at com.oracle.handson.EntitledInvocationService.query(EntitledInvocationService.java:50)
    at com.tangosol.coherence.component.net.extend.messageFactory.
InvocationServiceFactory$InvocationRequest.onRun(InvocationServiceFactory.CDB:12)
    at com.tangosol.coherence.component.net.extend.message.Request.run(Request.CDB:4)
    at com.tangosol.coherence.component.net.extend.proxy.serviceProxy.InvocationServiceProxy.
onMessage(InvocationServiceProxy.CDB:9)
    at com.tangosol.coherence.component.net.extend.Channel$MessageAction.run(Channel.CDB:13)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:356)
    at com.tangosol.coherence.component.net.extend.Channel.execute(Channel.CDB:48)
    at com.tangosol.coherence.component.net.extend.Channel.receive(Channel.CDB:26)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.
Peer$DaemonPool$WrapperTask.run(Peer.CDB:9)
    at com.tangosol.coherence.component.util.DaemonPool$WrapperTask.run(DaemonPool.CDB:32)
    at com.tangosol.coherence.component.util.DaemonPool$Daemon.onNotify(DaemonPool.CDB:65)
    at com.tangosol.coherence.component.util.Daemon.run(Daemon.CDB:51)
    at java.lang.Thread.run(Thread.java:724)

```

---

---

## Working with Live Events

In this exercise you learn how to work with entry and `EntryProcessor` events, and how to be notified of events using event interceptors. This exercise provides instructions for creating a project where you create event interceptors and cause events to exercise the features of the Live Events framework.

A brief overview of the Live Events framework is also provided. For a more detailed description of the framework and the API discussed in this chapter, see "Using Live Events" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence* and *Oracle Fusion Middleware Java API Reference for Oracle Coherence*.

This chapter contains the following sections:

- [Introduction](#)
- [Creating, Registering, and Executing an Event Interceptor](#)
- [Vetoing Pre- and Post-commit Events Using an Event Interceptor](#)
- [Logging Partition Activity Using an Event Interceptor](#)

### 11.1 Introduction

Coherence provides an event framework that allows your applications to react to operations performed in the data grid. The framework uses an event-based model where events represent observable occurrences of cluster operations. The supported events include partitioned service, cache, and application events. These events can be consumed by registering event interceptors (classes that implement `EventInterceptor`) either programmatically or by using the cache configuration.

#### 11.1.1 About Event Interceptors

Applications can react to Live Events by registering event interceptors (`EventInterceptor`). The interceptors explicitly define which events to receive and what action, if any, to take. Any number of event interceptors can be created and registered for a specific cache or for all caches managed by a specific partitioned service. Multiple interceptors that are registered for the same event type are automatically chained together and executed in the context of a single event.

Event interceptors are created by implementing the `EventInterceptor` interface. The interface is defined using generics and allows you to filter the events of interest by specifying the generic type of the event as a type parameter. The inherited `onEvent` method provides the ability to perform any necessary processing upon receiving an event. For details on the `EventInterceptor` API, see *Oracle Fusion Middleware Java API Reference for Oracle Coherence*.

The `@Interceptor` annotation is used to restrict the events to specific event types and also provides further configuration of the interceptor. The `@Interceptor` annotation includes the following attributes:

- `identifier`—Specifies a unique identifier for the interceptor. This value can be overridden when registering an interceptor class in the cache configuration file.
- `entryEvents`—Specifies an array of entry event types to which the interceptor wants to subscribe.
- `entryProcessorEvents`—Specifies an array of entry processor event types to which the interceptor wants to subscribe.
- `transferEvents`—Specifies an array of transfer event types to which the interceptor wants to subscribe.
- `transactionEvents`—Specifies an array of transaction event types to which the interceptor wants to subscribe.
- `order`—Specifies whether the interceptor is placed at the front of a chain of interceptors. The legal values are `HIGH` and `LOW`. A value of `HIGH` indicates that the interceptor is placed at the front in the chain of interceptors. A value of `LOW` indicates no order preference. The default value is `LOW`. This value can be overridden when registering an interceptor class in the cache configuration file.

For more information on the `@Interceptor` annotation, see "Creating Event Interceptors" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*

### 11.1.2 About Cache Events

Cache events are raised due to some operation performed against one or many entries in a cache. Cache events include entry events and entry processor events. An entry event (`EntryEvent`) can represent one of several operations (inserting, updating, and removing) performed against entries in a cache. Entry events can be divided into precommit events (`INSERTING`, `UPDATING`, and `REMOVING`), which are raised before the operation is performed to allow modification to an entry, and post-commit events (`INSERTED`, `UPDATED`, and `REMOVED`) which are raised after an operation has completed and in the same order as the events occurred.

Entry processor (`EntryProcessor`) events represent the execution of an `EntryProcessor` on a set of entries in a cache. Entry processor events can be divided into precommit events (`EXECUTING`), which are raised before an entry processor is executed to allow modification to the entry processor implementation, and post-commit events (`EXECUTED`), which are raised after an entry processor is executed and in the same order that the events occurred.

### 11.1.3 About Partitioned Service Events

Partitioned service (`PartitionedService`) events are comprised of transfer events, which represent partition transfers between storage-enabled members, and transaction events. Transfer events are dispatched in the context of a partition being transferred, however the contents belonging to a partition are immutable.

### 11.1.4 About Event Interceptor Registration

You register an event interceptor either in a cache configuration file or programmatically. An event interceptor is registered either for one or many caches, or for a specific partitioned service. An event interceptor that is registered for a specific cache only receives events that pertain to that cache. An event interceptor that is

registered for a specific partitioned service receives events for all caches that are managed by the service.

In the cache configuration file, the full class name of the event interceptor is specified in the `<interceptor>` element, which appears under `<cache-name>` in the `<cache-mapping>` stanza. The interceptor is associated with the cache specified in the `<cache-name>` element. An event interceptor can also be registered for a partitioned service in the `<caching-schemes>` stanza. To do this, include an `<interceptors>` element, within `<distributed-scheme>` element, that includes any number of `<interceptor>` subelements.

Instead of using the cache configuration file, event interceptors can be registered programmatically. The key classes and methods to register event interceptors are the `getInterceptorRegistry` method on the `ConfigurableCacheFactory` interface and the `getEventInterceptor` and `registerEventInterceptor` methods on the `InterceptorRegistry` interface. For example, the following code registers the `TimedTraceInterceptor`, which is an `EventInterceptor` introduced later in this chapter:

```
ConfigurableCacheFactory ccf = CacheFactory.getConfigurableCacheFactory();
InterceptorRegistry reg = ccf.getInterceptorRegistry();

reg.registerEventInterceptor(new TimedTraceInterceptor(), RegistrationBehavior.
    FAIL);
```

A detailed description and examples of registering event interceptors programmatically is beyond the scope of this documentation. For more information, see "Using Live Events" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence* and *Oracle Fusion Middleware Java API Reference for Oracle Coherence*.

## 11.2 Creating, Registering, and Executing an Event Interceptor

The following sections describe how to create, register, and execute an event interceptor. In this exercise, you will work with an event interceptor that will measure the timing between pre- and post-commit events.

To complete this exercise, follow these steps:

1. [Create an Event Interceptor to Measure the Time Between a Pre- and a Post-commit Event](#)
2. [Create a Class to Delay the Processing of Events](#)
3. [Register the Timed Events Event Interceptor](#)
4. [Create a POF Configuration File for the Lazy Processor Class](#)
5. [Create a Class to Exercise the Timed Events Event Interceptor](#)
6. [Create a Driver File for Timed Events Example](#)
7. [Create a Cache Server Startup Configuration](#)
8. [Create a Startup Configuration for the Timed Events Driver](#)
9. [Run the Timed Events Example](#)

## 11.2.1 Create a Event Interceptor to Measure the Time Between a Pre- and a Post-commit Event

Create a event interceptor named `TimedTraceInterceptor` to measure the timings between precommit and post-commit events (that is, `INSERTING/INSERTED`, `UPDATING/UPDATED`, and `REMOVING/REMOVED`) for each type of event.

To create the `TimedTraceInterceptor` event interceptor class:

1. Create a new Application Client Project in Eclipse named `UEMEvents`. Ensure that `CoherenceConfig` is selected in the **Configuration** field on the opening page and the **Create a default main** is *not* selected on the Application Client module page.  
See "Creating a New Project in the Eclipse IDE" on page 2-3 for detailed information.
2. Create a new Java class called `TimedTraceInterceptor`. Ensure that the **Default Package** is `com.oracle.handson`. Do not select the **Main Method** check box.  
See "Creating a Java Class" on page 2-11 for more information.
3. Write an event interceptor that implements the `EventInterceptor` interface. The interceptor should provide timings between pre- and post-commit events for each type of event: inserts, updates, removes, and an entry processor. You can write your own interceptor or use the code that is provided in [Example 11-1](#).

[Example 11-1](#) illustrates the event interceptor `TimedTraceInterceptor`. The interceptor implements the `EventInterceptor` interface. The `@Interceptor` annotation provides the unique name of the interceptor with the `identifier` attribute and the order in which it should be executed (`Order.HIGH`) with the `order` attribute.

The interceptor also contains a protected `EventTimer` inner-class. This class times the elapsed time for each event it is notified of. The interceptor tracks the time between a pre- and post-commit event for each entry and the respective event types (`INSERT`, `UPDATE`, `REMOVE`). The timings are sent to the Coherence log in batches displaying sample and cumulative statistics.

As the generic argument is `com.tangosol.net.events.partition.cache.Event` you will only get events that are consumers of that event, that is, `EntryEvent` and `EntryProcessorEvent`, without specifying any filtering.

### **Example 11-1 Class to Provide Timings Between Pre- and Post-commit Events**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import com.tangosol.net.events.EventInterceptor;
import com.tangosol.net.events.annotation.Interceptor;
import com.tangosol.net.events.annotation.Interceptor.Order;
import com.tangosol.net.events.partition.cache.EntryEvent;
import com.tangosol.net.events.partition.cache.EntryEvent.Type;
import com.tangosol.net.events.partition.cache.EntryProcessorEvent;
import com.tangosol.net.events.partition.cache.Event;

import com.tangosol.util.Binary;
import com.tangosol.util.BinaryEntry;

import java.util.HashMap;
import java.util.Map;
```

```

import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.atomic.AtomicInteger;
import java.util.concurrent.atomic.AtomicLong;

/**
 * TimedTraceInterceptor provides timings between pre- and post-commit events
 * for each type of event i.e. inserts, updates, removes, and entry processor
 * execution.
 * <p>
 * These timings are collected and averaged at a sample rate defined by
 * parameter <tt>cSample</tt>. Additionally they are output to the log
 * at the same time. This implementation does maintain a strong reference to
 * the each binary key however this is removed upon receiving the post-commit
 * event
 * for the same key.
 *
 * @since Coherence 12.1.2
 */
@Interceptor(identifier = "trace", order = Order.HIGH)
public class TimedTraceInterceptor
    implements EventInterceptor<Event<? extends Enum<?>>>
    {
        // ----- constructors -----

        /**
         * Default no-arg constructor.
         */
        public TimedTraceInterceptor()
        {
            this(DEFAULT_SAMPLE_RATE);
        }

        /**
         * Construct an TimedTraceInterceptor with specified {@link Event}
         * types and chain position.
         *
         * @param cSample    sample size to calculate mean and output statistics
         */
        public TimedTraceInterceptor(int cSample)
        {
            Map<Enum, EventTimer> mapTimedEvents = m_mapTimedEvents = new
HashMap<Enum, EventTimer>(3);
            EventTimer insertTimer = new EventTimer(Type.INSERTED, cSample);
            EventTimer updateTimer = new EventTimer(Type.UPDATED, cSample);
            EventTimer removeTimer = new EventTimer(Type.REMOVED, cSample);
            EventTimer invocationTimer = new EventTimer(EntryProcessorEvent.Type.
EXECUTED, cSample);

            mapTimedEvents.put(Type.INSERTED, insertTimer);
            mapTimedEvents.put(Type.INSERTING, insertTimer);
            mapTimedEvents.put(Type.UPDATED, updateTimer);
            mapTimedEvents.put(Type.UPDATING, updateTimer);
            mapTimedEvents.put(Type.REMOVED, removeTimer);
            mapTimedEvents.put(Type.REMOVING, removeTimer);
            mapTimedEvents.put(EntryProcessorEvent.Type.EXECUTED, invocationTimer);
            mapTimedEvents.put(EntryProcessorEvent.Type.EXECUTING, invocationTimer);
        }

        // ----- EventInterceptor methods -----

```

```

/**
 * {@inheritDoc}
 */
public void onEvent(Event event)
{
    if (event instanceof EntryEvent)
    {
        process((EntryEvent) event);
    }
    else if (event instanceof EntryProcessorEvent)
    {
        process((EntryProcessorEvent) event);
    }
}

/**
 * This method will be invoked upon execution of an entry processor and
 * will time its execution from prior to post execution, including any
 * backup requests that need to be made as a result.
 *
 * @param event the {@link EntryProcessorEvent} that encompasses the
 * requested event
 */
protected void process(EntryProcessorEvent event)
{
    EventTimer mapTimedEvents = m_mapTimedEvents.get(event.getType());

    for (BinaryEntry binEntry : event.getEntrySet())
    {
        if (event.getType() == EntryProcessorEvent.Type.EXECUTING)
        {
            mapTimedEvents.starting(binEntry);
        }
        else if (event.getType() == EntryProcessorEvent.Type.EXECUTED)
        {
            mapTimedEvents.started(binEntry);
        }
    }
}

/**
 * This method will be invoked upon execution of a data mutating request
 * and will time its execution from prior to post execution, including
 * any backup requests that need to be made as a result.
 *
 * @param event the {@link EntryEvent} that encompasses the
 * requested event
 */
protected void process(EntryEvent event)
{
    EventTimer mapTimedEvents = m_mapTimedEvents.get(event.getType());

    switch ((Type) event.getType())
    {
        case INSERTING:
        case UPDATING:
        case REMOVING:
            for (BinaryEntry binEntry : event.getEntrySet())
            {

```

```

        mapTimedEvents.starting(binEntry);
    }
    break;
case INSERTED:
case UPDATED:
case REMOVED:
    for (BinaryEntry binEntry : event.getEntrySet())
    {
        mapTimedEvents.started(binEntry);
    }
    break;
}
}

// ----- inner class: EventTimer -----

/**
 * The EventTimer times the elapsed time for each event it is notified
 * of. It correlates the completion event based on equality comparisons
 * of the Binary provided. Additionally it calculates the mean based on a
 * sample set of <tt>cSample</tt> size. When reaching this sample set
 * a log will be made of the current sample set mean and the cumulative
 * mean.
 */
protected class EventTimer
{
    // ----- constructors -----

    /**
     * Construct an EventTimer with the event type provided.
     *
     * @param eventType the type of event this timer will be timing
     */
    protected EventTimer(Enum eventType, int cSample)
    {
        m_eventType = eventType;
        m_cSampleSize = cSample;
    }

    /**
     * Notifies the timer of the execution of the provided key will
     * imminently commence.
     *
     * @param binEntry the event will commence for this <tt>binEntry</tt>
     */
    public void starting(BinaryEntry binEntry)
    {
        m_mapElapsedTimes.put(binEntry.getBinaryKey(), System.nanoTime());
    }

    /**
     * Notifies the timer of the completion of the event for the provided
     * key.
     *
     * @param binEntry the event has completed for this <tt>binEntry</tt>
     */
    public void started(BinaryEntry binEntry)
    {
        Long lStart = m_mapElapsedTimes.remove(binEntry.getBinaryKey());

```

```

        if (lStart == null)
        {
            return;
        }

        add(System.nanoTime() - lStart);
    }

/**
 * Regardless of the specific data item add the elapsed time taken to
 * process the data item. Upon reaching the sample set size of events
 * calculate the mean, reset timings and continue.
 *
 * @param lElapsed the number of nanos taken for a data item to
 *                process
 */
protected void add(long lElapsed)
    {
        AtomicInteger nEvents          = m_nEvents;
        AtomicLong   lTotalElapsed    = m_lTotalElapsed;
        int          nCurrEvents      = nEvents.incrementAndGet();
        long         lCurrTotalElapsed = lTotalElapsed.addAndGet(lElapsed);

        if (nCurrEvents % m_cSampleSize == 0)
        {
            nEvents.set(0);
            lTotalElapsed.set(0L);
            ++m_cSamples;

            long lMean = lCurrTotalElapsed / nCurrEvents;
            m_lMean = m_lMean == 0 ? lMean : lMean + m_lMean / 2;

            String sStats = String.format("EventStats[name = %s, sampleMean =
            %fms, mean = %fms]",
            m_eventType, (double) lMean / 1000000, (double) m_lMean /
            1000000);

            CacheFactory.log(sStats, CacheFactory.LOG_INFO);

            NamedCache cacheResults = CacheFactory.getCache("events-results");
            int nMemberId = CacheFactory.getCluster().
            getLocalMember().getId();

            cacheResults.put(
                String.format("%d-%s-%d", nMemberId, m_eventType.name(),
                m_cSamples),
                sStats);
        }
    }

// ----- data members -----

/**
 * Sample size to calculate mean and output statistics.
 */
private int m_cSampleSize;

/**
 * The start times for a number of Binary keys.
 */

```

```

        private Map<Binary, Long> m_mapElapsedTimes = new
ConcurrentHashMap<Binary, Long>();

        /**
         * A counter of the total elapsed time.
         */
        private AtomicLong      m_lTotalElapsed = new AtomicLong();

        /**
         * A counter of the number of events processed
         */
        private AtomicInteger    m_nEvents = new AtomicInteger();

        /**
         * An average over time.
         */
        private long            m_lMean;

        /**
         * The number of samples taken.
         */
        private int             m_cSamples;

        /**
         * The type of event being timed.
         */
        private Enum            m_eventType;
    }

    // ----- constants -----

    /**
     * The sample size for elapsed times.
     */
    protected static final int DEFAULT_SAMPLE_RATE = 100;

    // ----- data members -----

    /**
     * A map of event types to their timers.
     */
    private Map<Enum, EventTimer> m_mapTimedEvents;
}

```

## 11.2.2 Create a Class to Delay the Processing of Events

Create a class named `LazyProcessor` to create a superficial delay between the processing of events. The class should be able to specify the number of milliseconds this processor should sleep between processing events. This class will be used by the `EventsTimingExample` subclass in the `EventsExamples` class. You will create the `EventsExamples` class in a later step.

The data that the `LazyProcessor` class produces will be sent across the wire, so the class should use POF (Portable Object Format). You will add the `LazyProcessor` class to the POF configuration file in a later step.

To create the `LazyProcessor` class:

1. Create a new Java class called `LazyProcessor`. Do not include a main method.

See "[Creating a Java Class](#)" on page 2-11 for more information.

2. Create the code for the `LazyProcessor` class. Because the class uses the `PortableObject` interface for data serialization, the `LazyProcessor` class must implement `PortableObject` interface. The class must also extend the `AbstractProcessor` class. Import the `Base`, `InvocableMap.Entry`, `AbstractProcessor`, `PortableObject`, `PofReader`, and `PofWriter` classes and interfaces. You can write your own code for the `LazyProcessor` class or use the code that is provided in [Example 11–2](#).

**Example 11–2 Class to Delay the Processing of Events**

```
package com.oracle.handson;

import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PofWriter;
import com.tangosol.io.pof.PortableObject;
import com.tangosol.util.Base;
import com.tangosol.util.InvocableMap.Entry;
import com.tangosol.util.processor.AbstractProcessor;

import java.io.IOException;

/**
 * LazyProcessor will sleep for a specified period of time.
 *
 *
 * @since 12.1.2
 */
public class LazyProcessor
    extends AbstractProcessor
    implements PortableObject
{
    // ----- constructors -----

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    /**
     * Default no-arg constructor.
     */
    public LazyProcessor()
    {
    }

    /**
     * Constructs a LazyProcessor with a specified time to sleep.
     *
     * @param lSleepTime the number of milliseconds this processor should
     *                   sleep
     */
    public LazyProcessor(long lSleepTime)
    {
        m_lSleepTime = lSleepTime;
    }

    /**
     * {@inheritDoc}
     */
    public Object process(Entry entry)
```

```

    {
    try
        {
            Thread.sleep(m_lSleepTime);
        }
    catch (InterruptedException e)
        {
            throw Base.ensureRuntimeException(e);
        }
    return null;
    }

// ----- PortableObject members -----

/**
 * {@inheritDoc}
 */
public void readExternal(PofReader in) throws IOException
    {
        m_lSleepTime = in.readLong(0);
    }

/**
 * {@inheritDoc}
 */
public void writeExternal(PofWriter out) throws IOException
    {
        out.writeLong(0, m_lSleepTime);
    }

// ----- data members -----

/**
 * The number of milliseconds this processor should sleep.
 */
private long m_lSleepTime;
}

```

### 11.2.3 Register the Timed Events Event Interceptor

In the UEMEvents project, the interceptors are registered in the cache configuration file. The fully-qualified name of the event interceptor is specified in the `interceptor` element, which appears under `<cache-name>` element in the `<cache-mapping>` stanza. The interceptor is associated with the cache specified in the `<cache-name>` element. In this example, `TimedTraceInterceptor` is the event interceptor on the events cache.

To create a cache configuration file which defines the event interceptors:

1. Open the `coherence-cache-config.xml` file from the Project Explorer window. You can find the file under `Events/appClientModule`.
2. Save the file as `uem-cache-config.xml`.
3. Write the cache configuration that calls the event interceptors. The following list highlights some key elements:
  - Under the `<cache-mapping>` element there is a reference from the `com.oracle.handson.TimedTraceInterceptor` class in the `<interceptor>` element to the events cache in the `<cache-name>` element. The events cache uses the `events-distributed-scheme`. This scheme uses the `PartitionedPofCache` service which is listed under `<distributed-schemes>`.

- There is a mapping between the events-results cache and the dist-events-results scheme. In the <distributed-scheme> section, there is an association between the dist-events-results scheme and the PartitionedEventsResults service.

Example 11-3 illustrates a possible implementation for the uem-cache-config.xml file.

**Example 11-3 Cache Configuration File That Registers the TimedTraceInterceptor**

```
<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
              xsi:schemaLocation="http://xmlns.oracle.
com/coherence/coherence-cache-config coherence-cache-config.xsd">
  <defaults>
    <serializer>pof</serializer>
  </defaults>

  <caching-scheme-mapping>
    <cache-mapping>
      <cache-name>events</cache-name>
      <scheme-name>events-distributed-scheme</scheme-name>
      <interceptors>
        <interceptor>
          <instance>
            <class-name>com.oracle.handson.
TimedTraceInterceptor</class-name>
            <init-params>
              <init-param>
                <param-type>int</param-type>
                <param-value>100</param-value>
              </init-param>
            </init-params>
          </instance>
        </interceptor>
      </interceptors>
    </cache-mapping>
  </caching-scheme-mapping>

  <cache-mapping>
    <cache-name>events-results</cache-name>
    <scheme-name>dist-events-results</scheme-name>
  </cache-mapping>
</caching-scheme-mapping>

<caching-schemes>
  <distributed-scheme>
    <scheme-name>events-distributed-scheme</scheme-name>
    <service-name>PartitionedPofCache</service-name>
    <thread-count>5</thread-count>
    <backing-map-scheme>
      <local-scheme>
        <!-- each node will be limited to 32MB -->
        <high-units>32M</high-units>
        <unit-calculator>binary</unit-calculator>
      </local-scheme>
    </backing-map-scheme>
    <autostart>true</autostart>
  </distributed-scheme>

  <!-- A PartitionedCache service used to store results for events examples
```

```

-->
<distributed-scheme>
  <scheme-name>dist-events-results</scheme-name>
  <service-name>PartitionedEventsResults</service-name>
  <thread-count>5</thread-count>
  <backing-map-scheme>
    <local-scheme/>
  </backing-map-scheme>
  <autostart>true</autostart>
</distributed-scheme>

<!--
Invocation Service scheme.
-->
<invocation-scheme>
  <scheme-name>examples-invocation</scheme-name>
  <service-name>InvocationService</service-name>

  <autostart system-property="tangosol.coherence.invocation.
autostart">true</autostart>
</invocation-scheme>

</caching-schemes>
</cache-config>

```

## 11.2.4 Create a POF Configuration File for the Lazy Processor Class

All of the information produced by the `TimedTraceInterceptor` class is exclusive to its local member. The `LazyProcessor`, and its state, is transmitted to storage-enabled members and executed, thus it must be added to the POF configuration file. In the Eclipse Project Explorer, right-click the `pof-config.xml` file and rename it `uem-pof-config.xml`. Open the `uem-pof-config.xml` file in the editor and replace its contents with the code in [Example 11-4](#). The example illustrates a POF configuration file containing this class.

### **Example 11-4** POF Configuration File for the `LazyProcessor` Class

```

<?xml version="1.0"?>
<pof-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://xmlns.oracle.com/coherence/coherence-pof-config"
  xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-pof-config
http://xmlns.oracle.com/coherence/coherence-pof-config/1.2/coherence-pof-config.
xsd">
  <user-type-list>
  <!-- include all "standard" Coherence POF user types -->
  <include>coherence-pof-config.xml</include>
  <user-type>
    <type-id>1008</type-id>
    <class-name>com.oracle.handson.LazyProcessor</class-name>
  </user-type>
  </user-type-list>
</pof-config>

```

## 11.2.5 Create a Class to Exercise the Timed Events Event Interceptor

Create a class named `EventsExamples` to trigger actions to be performed by the `TimedTraceInterceptor` class. The class should illustrate how the elapsed time can be measured between pre- and post-commit events inserted into a results cache. The

entries inserted into the results cache are sent to standard output by the process executing this class.

To create the `EventsExamples` class:

1. Create a new Java class called `EventsExamples`. Do not include a main method. See ["Creating a Java Class"](#) on page 2-11 for more information.
2. Write the code for the `EventsExamples` class. Import the `LazyProcessor`, `CacheFactory`, `NamedCache`, `PartitionedService`, `MapEvent`, `MapListener`, `MultiplexListener`, and `Callable` classes and interfaces. You can write your own code or use the code supplied in [Example 11-5](#).

[Example 11-5](#) illustrates a sample implementation of the `EventsExamples` class. The example contains the `EventsTimingExample` inner-class. This inner-class accesses the events and event-results caches and obtains the number of cache cluster members, the name of the triggering event, and the sample size from the `TimedTraceInterceptor` class logs. The calculation of the time for the sample to be processed and the mean time for all of the samples to be processed are provided by the `TimedTraceInterceptor` class.

The `EventsTimingExample` subclass provides the values for the total amount of time for event processing and the number of threads on which the events can run. It also calls the `LazyProcessor` class to calculate the amount of time between processing events (sleep time).

**Example 11-5 Class to Exercise the `TimedTraceInterceptor` Event Interceptor**

```
package com.oracle.handson;

import com.oracle.handson.LazyProcessor;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.net.PartitionedService;
import com.tangosol.util.MapEvent;
import com.tangosol.util.MapListener;
import com.tangosol.util.MultiplexingMapListener;

import java.util.concurrent.Callable;

/**
 * EventsExamples illustrates various features within the Live Events
 * Model. This includes providing mean elapsed times split by event type.
 *
 * @since Coherence 12.1.2
 */
@SuppressWarnings("unchecked")
public class EventsExamples
{
    // ----- inner-class: EventsTimingExample -----

    /**
     * The EventsTimingExample is a catalyst for action to be performed by
     * {@link TimedTraceInterceptor}. This illustrates how the elapsed time
     * between pre- and post-commit events can be measured which are inserted
     * into a results cache. The entries inserted into the results cache are
     * displayed via the stdout of the process executing this class.
     */
}
```

```

public static class EventsTimingExample
    implements Callable<Boolean>
{
    // ----- Callable methods -----

    /**
     * {@inheritDoc}
     */
    public Boolean call() throws Exception
    {
        NamedCache cacheEvents = CacheFactory.getCache("events");
        NamedCache cacheResults = CacheFactory.getCache("events-results");
        int cFrequency = ((PartitionedService) cacheEvents.
getCacheService()).getOwnershipEnabledMembers().size();
        int cSet = 110;

        MapListener ml = new MultiplexingMapListener()
        {
            @Override
            protected void onMapEvent(MapEvent evt)
            {
                String[] asKey = ((String) evt.getKey()).split("-");

                System.out.printf("Received stats [memberId=%s, eventType=%s,
sample=%s] = %s\n",
                                asKey[0], asKey[1], asKey[2], evt.getNewValue());
            }
        };

        try
        {
            cacheResults.addMapListener(ml);

            // execute inserts and updates
            for (int i = cFrequency; i > 0; --i)
            {
                for (int j = 1, cMax = cSet * cFrequency; j <= cMax; ++j)
                {
                    cacheEvents.put(j, "value " + j);
                }
            }

            // execute processors
            int nTotalTime = 3000;
            int cThreads = 5;
            int nSleepTime = nTotalTime / (cThreads * cSet * cFrequency);
            for (int i = 1, cMax = cSet * cFrequency; i <= cMax; ++i)
            {
                cacheEvents.invoke(i, new LazyProcessor(nSleepTime));
            }
        }
        finally
        {
            cacheEvents.clear();
            cacheResults.removeMapListener(ml);
            cacheResults.clear();
        }
        return true;
    }
}

```

}

## 11.2.6 Create a Driver File for Timed Events Example

Create a driver file to run the `EventsTimingExample` example defined in the `EventsExamples` class.

To create a driver file:

1. Create a new Java class called `Driver` in the `UEMEvents` project. Ensure that it includes a main method.  
See ["Creating a Java Class"](#) on page 2-11 for detailed information.
2. Write the code to run the `EventsTimingExample` example defined in the `EventsExamples` class. You can write your own driver or use the code supplied in [Example 11-6](#).

### **Example 11-6 Driver File for Timed Events Example**

```
package com.oracle.handson;

import com.oracle.handson.EventsExamples.EventsTimingExample;

import java.util.LinkedHashMap;
import java.util.Map;
import java.util.concurrent.Callable;

/**
 * Driver executes all the Coherence Events examples.
 * <p>
 * <strong>Timed Events Example</strong> - In this example we time
 * the elapsed time between pre- and post-commit events for each of the events
 * that occur.
 *
 * @since Coherence 12.1.2
 */
public class Driver
{
    // ----- static methods -----

    /**
     * Execute Events examples.
     *
     * @param asArg  command line arguments
     */
    public static void main(String[] asArg)
    {
        System.out.println("----- events examples begin -----");

        // Run examples
        for (Map.Entry<String, Callable<Boolean>> example : EVENTS_EXAMPLES.
entrySet())
        {
            String sExample = example.getKey();

            try
            {
                System.out.printf("-----  %s begin\n\n", sExample);
                boolean fSuccess = example.getValue().call();
            }
        }
    }
}
```

```

        System.out.printf("\n-----  %s completed %ssuccessfully\n",
sExample, fSuccess ? "" : "un");
    }
    catch(Exception e)
    {
        System.err.printf("-----%s completed unsuccessfully with the
following exception:\n", sExample);
        e.printStackTrace();
    }
}

    System.out.println("----- events examples completed-----");
}

// ----- constants -----

/**
 * All the examples to be executed in insertion order.
 */
protected static final Map<String, Callable<Boolean>> EVENTS_EXAMPLES = new
LinkedHashMap<String, Callable<Boolean>>();

/**
 * Default examples.
 */
static
{
    EVENTS_EXAMPLES.put("timing interceptor", new EventsTimingExample());
}
}

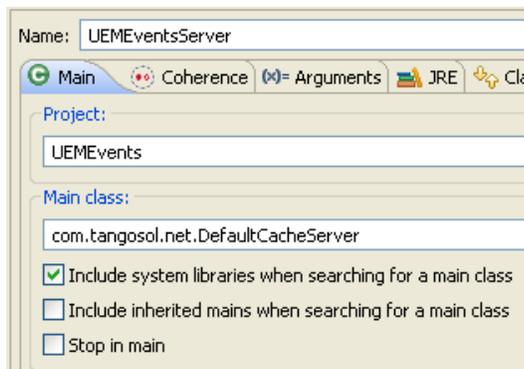
```

## 11.2.7 Create a Cache Server Startup Configuration

Create a configuration to start the cache server for the UEMEEvents project.

1. Right click the project and select **Run As** then **Run Configurations**. Double click the **Oracle Coherence** icon in the **Run Configurations** dialog box to create a new launch configuration.
2. In the **Main** tab, enter `UEMEEventsServer` in the **Name** field. Click **Browse** in the **Project** field and select the **UEMEEvents** project in the **Project Selection** dialog box. Select the **Include system libraries when searching for a main class** checkbox and click **Search**. Enter `DefaultCacheServer` in the **Select Type** field and select **com.tangosol.net.DefaultCacheServer**. Click **Apply**. The **Main** tab should look similar to [Figure 11-1](#).

**Figure 11–1 Main Tab for the Events Server Startup Configuration**

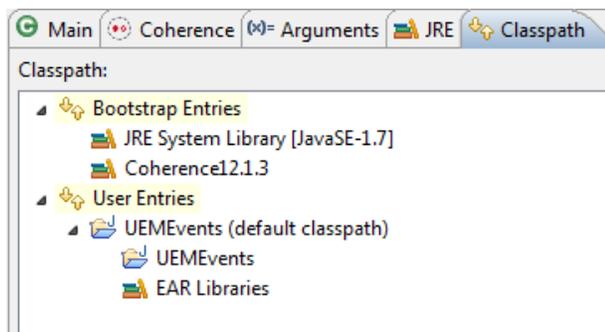


3. In the **General** tab of the **Coherence** tab, identify the path to the cache configuration file under **Cache configuration descriptor**. Click the **Browse** button to navigate to the **Absolute file path** of the cache configuration file `C:\home\oracle\workspace\UEMEvents\appClientModule\uem-cache-config.xml`. Select **Enabled (cache server)** under **Local storage**. Enter a unique value, such as 3155, for the **Cluster port**.

In the **Other** tab, set the `tangosol.pof.config` item to `uem-pof-config.xml`.

4. In the **Common** tab, select **Shared file** and browse to the `\UEMEvents` project.
5. The **Classpath** tab should look similar to [Example 11–2](#). The `UEMEvents` project appears below **User Entries**. The **JRE System Library** and **Coherence12.1.3** library appear in the **Bootstrap Entries** section.

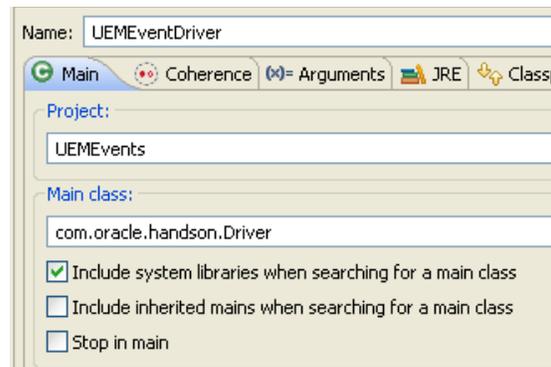
**Figure 11–2 Classpath Tab for the Events Server Startup Configuration**



## 11.2.8 Create a Startup Configuration for the Timed Events Driver

Create a configuration to start the client driver for the `UEMEvents` project.

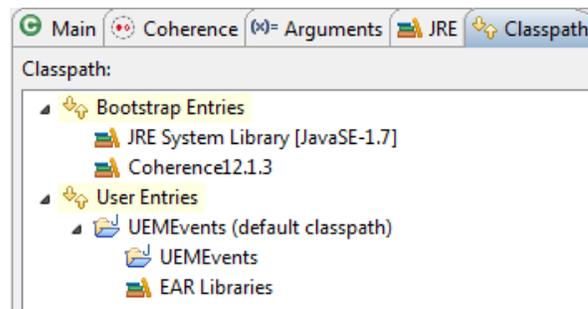
1. Right click the project and select **Run As** then **Run Configurations**. Double click the **Oracle Coherence** icon in the **Run Configurations** dialog box to create a new launch configuration.
2. In the **Main** tab, enter `UEMEventDriver` in the **Name** field. Click **Browse** in the **Project** field and select the `UEMEvents` project in the **Project Selection** dialog box. Select the **Include system libraries when searching for a main class** checkbox and click **Search**. Enter `Driver` in the **Select Type** field and select `com.oracle.handson.Driver`. Click **Apply**. The **Main** tab should look similar to [Figure 11–3](#).

**Figure 11–3 Main Tab for the Events Client Startup Configuration**

3. In the **General** tab of the **Coherence** tab, identify the path to the cache configuration file under **Cache configuration descriptor**. Click the **Browse** button to navigate to the **Absolute file path** of the cache configuration file `C:\home\oracle\workspace\UEMEvents\appClientModule\uem-cache-config.xml`. Select **Disabled (cache client)** under **Local storage**. Enter a unique value, such as 3155, for the **Cluster port**.

In the **Other** tab, set the **tangosol.pof.config** item to the `uem-pof-config.xml`.

4. In the **Common** tab, select **Shared file** and browse to the `\UEMEvents` project.
5. The **Classpath** tab should look similar to [Example 11–4](#). The `UEMEvents` project appears below **User Entries**. The **JRE System Library** and **Coherence12.1.3** library appear in the **Bootstrap Entries** section.

**Figure 11–4 Classpath Tab for the Events Client Startup Configuration**

## 11.2.9 Run the Timed Events Example

Right-click the `UEMEvents` project in the Project Explorer and select **Run As**, then **Run Configurations**. In the **Run Configurations** dialog box, select the `UEMEventsServer` launch configuration and click **Run** to start the cache server. After the cache server starts, select `UEMEventsServer` and click **Run** a second and a third time to start a total of three cache servers.

After the third cache server starts, select the `UEMEventDriver` configuration and click **Run**. The output from the cache client should look similar to [Example 11–7](#).

### **Example 11–7 Output from the Cache Client**

```
----- events examples begin -----
----- timing interceptor begin
2014-01-02 15:43:20.284/0.340 Oracle Coherence 12.1.3.0.0 <Info> (thread=main,
```

```

member=n/a): Loaded operational configuration from
"jar:file:/C:/Oracle/coherence/lib/coherence.jar!/tangosol-coherence.xml"
2014-01-02 15:43:20.384/0.440 Oracle Coherence 12.1.3.0.0 <Info> (thread=main,
member=n/a): Loaded operational overrides from
"jar:file:/C:/Oracle/coherence/lib/coherence.jar!/tangosol-coherence-override-dev.
xml"
2014-01-02 15:43:20.464/0.520 Oracle Coherence 12.1.3.0.0 <Info> (thread=main,
member=n/a): Loaded operational overrides from
"file:/C:/home/oracle/workspace/UEMEvents/build/classes/tangosol-coherence-overrid
e.xml"
2014-01-02 15:43:20.474/0.530 Oracle Coherence 12.1.3.0.0 <D5> (thread=main,
member=n/a): Optional configuration override "cache-factory-config.xml" is not
specified
2014-01-02 15:43:20.474/0.530 Oracle Coherence 12.1.3.0.0 <D5> (thread=main,
member=n/a): Optional configuration override "cache-factory-builder-config.xml" is
not specified
2014-01-02 15:43:20.474/0.530 Oracle Coherence 12.1.3.0.0 <D5> (thread=main,
member=n/a): Optional configuration override "/custom-mbeans.xml" is not specified
...
...
2014-01-02 15:43:24.650/4.706 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedPofCache, member=5): Service
PartitionedPofCache joined the cluster with senior service member 1
2014-01-02 15:43:24.699/4.755 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedEventsResults, member=5): Service
PartitionedEventsResults joined the cluster with senior service member 1
Received stats [memberId=2, eventType=INSERTED, sample=1] = EventStats[name =
INSERTED, sampleMean = 0.294040ms, mean = 0.294040ms]
Received stats [memberId=3, eventType=INSERTED, sample=1] = EventStats[name =
INSERTED, sampleMean = 0.397855ms, mean = 0.397855ms]
Received stats [memberId=1, eventType=INSERTED, sample=1] = EventStats[name =
INSERTED, sampleMean = 0.373270ms, mean = 0.373270ms]
Received stats [memberId=3, eventType=UPDATED, sample=1] = EventStats[name =
UPDATED, sampleMean = 0.187132ms, mean = 0.187132ms]
Received stats [memberId=2, eventType=UPDATED, sample=1] = EventStats[name =
UPDATED, sampleMean = 0.234314ms, mean = 0.234314ms]
Received stats [memberId=1, eventType=UPDATED, sample=1] = EventStats[name =
UPDATED, sampleMean = 0.237622ms, mean = 0.237622ms]
Received stats [memberId=2, eventType=UPDATED, sample=2] = EventStats[name =
UPDATED, sampleMean = 1.315323ms, mean = 1.432480ms]
Received stats [memberId=3, eventType=UPDATED, sample=2] = EventStats[name =
UPDATED, sampleMean = 0.417201ms, mean = 0.510767ms]
Received stats [memberId=1, eventType=UPDATED, sample=2] = EventStats[name =
UPDATED, sampleMean = 0.190555ms, mean = 0.309366ms]
Received stats [memberId=2, eventType=EXECUTED, sample=1] = EventStats[name =
EXECUTED, sampleMean = 1.766313ms, mean = 1.766313ms]
Received stats [memberId=3, eventType=EXECUTED, sample=1] = EventStats[name =
EXECUTED, sampleMean = 1.672603ms, mean = 1.672603ms]
Received stats [memberId=1, eventType=EXECUTED, sample=1] = EventStats[name =
EXECUTED, sampleMean = 1.676003ms, mean = 1.676003ms]

----- timing interceptor completed successfully
----- events examples completed-----
2014-01-02 15:43:28.344/8.400 Oracle Coherence GE 12.1.3.0.0 <D4>
(thread=ShutdownHook, member=5): ShutdownHook: stopping cluster node

```

## 11.3 Vetoing Pre- and Post-commit Events Using an Event Interceptor

In this exercise you will create an event interceptor to detect and veto events based on a specified key. To complete this exercise, follow these steps:

1. [Create an Event Interceptor to Detect and Veto Events](#)
2. [Register the Veto Events Event Interceptor](#)
3. [Create a Class to Exercise the Veto Events Event Interceptor](#)
4. [Edit the Driver File for the Veto Events Example](#)
5. [Run the Veto Events Example](#)

### 11.3.1 Create an Event Interceptor to Detect and Veto Events

To exercise the ability of Live Events to accept or veto events, create an event interceptor named `CantankerousInterceptor`. The interceptor will throw exceptions based on events that correspond to a specified key.

1. Create a new Java class called `CantankerousInterceptor`. Ensure that the **Default Package** is `com.oracle.handson`. Do not select the **Main Method** check box.

See "[Creating a Java Class](#)" on page 2-11 for more information.

2. Write the code for the event interceptor. Import the `Event`, `EventInterceptor`, `Interceptor`, `EntryEvent`, and `EntryEvent.Type` classes and interfaces. You can write your own interceptor code or use the code that is provided in [Example 11-8](#).

[Example 11-8](#) illustrates an event interceptor that throws a runtime exception during pre or post-commit events, based on the key that is attempting to be inserted. If the exception is thrown at precommit time, then a rollback occurs and the exception is propagated to the client. If the exception occurs at post-commit time, then a log event is recorded. The keys used for the exceptions are `VETO` and `NON-VETO`. `INSERTING` and `UPDATING` are events that can be vetoed, whereas `INSERTED` and `UPDATED` events cannot be vetoed.

#### **Example 11-8 Class to Detect and Veto Events**

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

import com.tangosol.net.events.Event;
import com.tangosol.net.events.EventInterceptor;
import com.tangosol.net.events.annotation.Interceptor;
import com.tangosol.net.events.partition.cache.EntryEvent;
import com.tangosol.net.events.partition.cache.EntryEvent.Type;

import com.tangosol.util.BinaryEntry;

/**
 * A CantankerousInterceptor is an {@link EventInterceptor} implementation
 * that is argumentative in nature, hence the event of inserting certain keys
 * will result in {@link RuntimeException}s at either pre- or post-commit
 * phases. Throwing a {@link RuntimeException} during the precommit phase
 * will result in a rollback and related exception being propagated to the
 * client. A post-commit exception will result in a log event. A precommit
 * event is considered an *ING event with a post-commit event being a
 * *ED event.
 */
```

```

* <p>
* This interceptor assumes it will be working against a cache with strings
* as keys with the following items that will be considered objectionable.
* <table>
*   <tr><td>Key</td><td>Exception Thrown During Event</td></tr>
*   <tr><td>{@value #VETO}</td><td>{@link Type#INSERTING} ||
*     {@link Type#UPDATING}</td></tr>
*   <tr><td>{@value #NON_VETO}</td><td>{@link Type#INSERTED} ||
*     {@link Type#UPDATED}</td></tr>
* </table>
*
*
* @since Coherence 12.1.2
*/
@Interceptor(identifier = "cantankerous",
    entryEvents = {Type.INSERTING, Type.INSERTED, Type.UPDATING, Type.
UPDATED})
public class CantankerousInterceptor
    implements EventInterceptor<EntryEvent>
{
    // ----- EventInterceptor methods -----

    /**
     * Throws {@link RuntimeException} iff the key used for this event is
     * {@code #VETO} or {@code #NON_VETO}.
     *
     * @param event the {@link Event} to be processed
     *
     * @throws RuntimeException iff {@code #VETO} || {@code #NON_VETO} are
     *     keys of the event
     */
    public void onEvent(EntryEvent event)
    {
        for (BinaryEntry binEntry : event.getEntrySet())
        {
            if (VETO.equals(binEntry.getKey()))
            {
                throw new RuntimeException("Objection! value = " + binEntry.
getValue());
            }
            else if (NON_VETO.equals(binEntry.getKey())
                && (event.getType() == Type.INSERTED || event.getType() ==
Type.UPDATED))
            {
                NamedCache cacheResults = CacheFactory.getCache("events-results");
                int nMemberId = CacheFactory.getCluster().
getLocalMember().getId();
                String sMessage = "Objection falls on deaf ears! value = "
+ binEntry.getValue();

                cacheResults.put(
                    String.format("%d-NON_VETO-%d", nMemberId, ++m_
cNonVetoableEvents),
                    sMessage);

                throw new RuntimeException(sMessage);
            }
        }
    }
}

```

```

// ----- constants -----
/**
 * String used to determine whether the event should be VETO'd during the
 * precommit phase.
 */
public static final String VETO      = "VETO";

/**
 * String used to determine whether the event should be VETO'd during the
 * post-commit phase.
 */
public static final String NON_VETO = "NON-VETO";

// ----- data members -----

/**
 * A counter of the number of non-vetoable exceptions raised.
 */
private int m_cNonVetoableEvents;
}

```

### 11.3.2 Register the Veto Events Event Interceptor

Open the `uem-cache-config.xml` cache configuration file and add the code to register the `CantankerousInterceptor` event interceptor and its cache. List the cache it references, `vetod-events`, in the `<cache-name>` element and its associated scheme events-distributed-scheme in the `<scheme-name>` element. List the fully-qualified class name of the `CantankerousInterceptor` class in the `<class-name>` subelement of the `<interceptors>` stanza. The added code is illustrated in bold font.

#### **Example 11–9 Cache Configuration to Register the CantankerousInterceptor Class**

```

<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
              xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-cache-config coherence-cache-config.xsd">
  <defaults>
    <serializer>pof</serializer>
  </defaults>

  <caching-scheme-mapping>
    <cache-mapping>
      <cache-name>events</cache-name>
      <scheme-name>events-distributed-scheme</scheme-name>
      <interceptors>
        <interceptor>
          <instance>
            <class-name>com.oracle.handson.
TimedTraceInterceptor</class-name>
            <init-params>
              <init-param>
                <param-type>int</param-type>
                <param-value>100</param-value>
              </init-param>
            </init-params>
          </instance>

```

```

        </interceptor>
    </interceptors>
</cache-mapping>
<cache-mapping>
    <cache-name>vetod-events</cache-name>
    <scheme-name>events-distributed-scheme</scheme-name>
    <interceptors>
        <interceptor>
            <instance>
                <class-name>com.oracle.handson.
CantankerousInterceptor</class-name>
            </instance>
        </interceptor>
    </interceptors>
</cache-mapping>
<cache-mapping>
    <cache-name>events-results</cache-name>
    <scheme-name>dist-events-results</scheme-name>
</cache-mapping>
</caching-scheme-mapping>

<caching-schemes>
<distributed-scheme>
    <scheme-name>events-distributed-scheme</scheme-name>
    <service-name>PartitionedPofCache</service-name>
    <thread-count>5</thread-count>
    <backing-map-scheme>
        <local-scheme>
            <!-- each node will be limited to 32MB -->
            <high-units>32M</high-units>
            <unit-calculator>binary</unit-calculator>
        </local-scheme>
    </backing-map-scheme>
    <autostart>true</autostart>
</distributed-scheme>

<!-- A PartitionedCache service used to store results for events examples
-->
<distributed-scheme>
    <scheme-name>dist-events-results</scheme-name>
    <service-name>PartitionedEventsResults</service-name>
    <thread-count>5</thread-count>
    <backing-map-scheme>
        <local-scheme/>
    </backing-map-scheme>
    <autostart>true</autostart>
</distributed-scheme>

<!--
Invocation Service scheme.
-->
<invocation-scheme>
    <scheme-name>examples-invocation</scheme-name>
    <service-name>InvocationService</service-name>

    <autostart system-property="tangosol.coherence.invocation.
autostart">true</autostart>
</invocation-scheme>

</caching-schemes>

```

```
</cache-config>
```

### 11.3.3 Create a Class to Exercise the Veto Events Event Interceptor

Create a class named `VetoExample` to exercise the `CantankerousInterceptor` event interceptor. Within the `VetoExample` class, create a subclass, `VetoedEventsExample`. The `VetoedEventsExample` subclass initiates the action to be performed by `CantankerousInterceptor`. The code illustrates the semantics of throwing exceptions in pre- and post-commit events. The exceptions that are expected only to be logged are inserted into a results cache. The entries inserted into the results cache are displayed by using the standard output of the process executing this class.

#### **Example 11–10** Class to Exercise the *TimedTraceInterceptor EventInterceptor*

```
package com.oracle.handson;

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.util.MapEvent;
import com.tangosol.util.MapListener;
import com.tangosol.util.MultiplexingMapListener;

import java.util.concurrent.Callable;

/**
 * VetoExample illustrates the different semantics in throwing exceptions in pre
 * events compared to post events.
 *
 * @since Coherence 12.1.2
 */
@SuppressWarnings("unchecked")
public class VetoExample
{
    /**
     * The VetoExample is a catalyst for action to be performed by
     * {@link CantankerousInterceptor}. This illustrates the semantics of
     * throwing exceptions in pre- and post-commit events. The exceptions that are
     * expected to only be logged are inserted into a results cache. The
     * entries inserted into the results cache are
     * displayed via the stdout of the process executing this class.
     */
    public static class VetoedEventsExample
        implements Callable<Boolean>
    {
        // ----- Callable methods -----

        /**
         * {@inheritDoc}
         */
        public Boolean call() throws Exception
        {
            // perform events to cause interceptors to veto said event
            NamedCache cacheVetoEvents = CacheFactory.getCache("vetod-events");
            NamedCache cacheResults    = CacheFactory.getCache("events-results");
            MapListener ml              = new MultiplexingMapListener()
            {
                @Override
                protected void onMapEvent(MapEvent evt)
            }
        }
    }
}
```

```

        {
            String[] asKey = ((String) evt.getKey()).split("-");

            System.out.printf("Received event [memberId=%s, eventType=%s,
count=%s] = %s\n",
                            asKey[0], asKey[1], asKey[2], evt.getNewValue());
        }
    };
    try
    {
        int cSet      = 110;
        int cVetos    = 5;
        int cNonVetos = 10;
        int cVetod    = 0;

        cacheResults.addMapListener(ml);

        for (int i = 1; i <= cSet; ++i)
        {
            boolean fVetod = false;

            if (i % (cSet / cVetos) == 0)
            {
                try
                {
                    cacheVetoEvents.put(CantankerousInterceptor.VETO,
"value: " + i);
                }
                catch(Throwable e)
                {
                    fVetod = true;
                    ++cVetod;
                }
            }
            if (i % (cSet / cNonVetos) == 0)
            {
                cacheVetoEvents.put(CantankerousInterceptor.NON_VETO,
"value: " + i);
                fVetod = true;
            }

            if (!fVetod)
            {
                cacheVetoEvents.put(String.valueOf(i), "value: " + i);
            }
        }
        System.out.printf("Number of veto'd events: %d\n", cVetod);
    }
    finally
    {
        cacheVetoEvents.clear();
        cacheResults.removeMapListener(ml);
        cacheResults.clear();
    }
    return true;
}
}
}

```

### 11.3.4 Edit the Driver File for the Veto Events Example

Edit the driver file that you created in ["Create a Driver File for Timed Events Example"](#) on page 11-16 to run the `VetoedEventsExample` example defined in the `VetoExample` class.

To edit the driver file:

1. Replace the import statement for the Events Timing Example:

```
import com.oracle.handson.EventsExamples.EventsTimingExample;
```

with an import statement for the `VetoedEventsExample` subclass.

```
import com.oracle.handson.VetoExample.VetoedEventsExample;
```

2. Replace the command which calls the Timed Events example:

```
EVENTS_EXAMPLES.put("timing interceptor", new EventsTimingExample());
```

with the following command to run the `VetoedEventsExample` example defined in the `VetoExample` class:

```
EVENTS_EXAMPLES.put("veto interceptor", new VetoedEventsExample());
```

### 11.3.5 Run the Veto Events Example

Right-click the `UEMEEvents` project in the Project Explorer and select **Run As**, then **Run Configurations**. In the **Run Configurations** dialog box, select the `UEMEEventsServer` launch configuration that you created in ["Create a Cache Server Startup Configuration"](#) on page 11-17. Click **Run** to start the cache server. After the cache server starts, select `UEMEEventsServer` and click **Run** a second and a third time to start a total of three cache servers.

After the third cache server starts, select the `UEMEEventDriver` launch configuration and click **Run**. The output from the veto events client should look similar to [Example 11-11](#).

#### **Example 11-11 Output from the Veto Events Client**

```
----- events examples begin -----
----- veto interceptor begin
2014-01-02 16:07:28.606/0.340 Oracle Coherence 12.1.3.0.0 <Info> (thread=main,
member=n/a): Loaded operational configuration from
"jar:file:/C:/Oracle/coherence/lib/coherence.jar!/tangosol-coherence.xml"
2014-01-02 16:07:28.716/0.450 Oracle Coherence 12.1.3.0.0 <Info> (thread=main,
member=n/a): Loaded operational overrides from
"jar:file:/C:/Oracle/coherence/lib/coherence.jar!/tangosol-coherence-override-dev.
xml"
...
2014-01-02 16:07:33.018/4.752 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedEventsResults, member=4): Service
PartitionedEventsResults joined the cluster with senior service member 1
Received event [memberId=3, eventType=NON_VETO, count=1] = Objection falls on deaf
ears! value = value: 11
Received event [memberId=3, eventType=NON_VETO, count=2] = Objection falls on deaf
ears! value = value: 22
Received event [memberId=3, eventType=NON_VETO, count=3] = Objection falls on deaf
ears! value = value: 33
Received event [memberId=3, eventType=NON_VETO, count=4] = Objection falls on deaf
ears! value = value: 44
Received event [memberId=3, eventType=NON_VETO, count=5] = Objection falls on deaf
```

```

ears! value = value: 55
Received event [memberId=3, eventType=NON_VETO, count=6] = Objection falls on deaf
ears! value = value: 66
Received event [memberId=3, eventType=NON_VETO, count=7] = Objection falls on deaf
ears! value = value: 77
Received event [memberId=3, eventType=NON_VETO, count=8] = Objection falls on deaf
ears! value = value: 88
Received event [memberId=3, eventType=NON_VETO, count=9] = Objection falls on deaf
ears! value = value: 99
Number of veto'd events: 5
Received event [memberId=3, eventType=NON_VETO, count=10] = Objection falls on
deaf ears! value = value: 110
----- veto interceptor completed successfully
----- events examples completed-----

```

Notice the output from the third cache server illustrated in [Example 11–12](#). The output displays the exceptions caused by the vetoed events.

### **Example 11–12 Output from the Cache Server**

```

Started DefaultCacheServer...

2014-01-02 16:06:50.262/5.785 Oracle Coherence GE 12.1.3.0.0 <D4>
(thread=DistributedCache:PartitionedPofCache, member=2): Asking member 1 for
primary ownership of PartitionSet{0..127}
...
...
2014-01-02 16:07:33.178/48.701 Oracle Coherence GE 12.1.3.0.0 <Error>
(thread=DistributedCache:PartitionedPofCache:EventDispatcher, member=2): Exception
caught while dispatching to "<cantankerous, com.oracle.handson.
CantankerousInterceptor>": java.lang.RuntimeException: Objection falls on deaf
ears! value = value: 11
    at com.oracle.handson.CantankerousInterceptor.
onEvent(CantankerousInterceptor.java:74)
    at com.oracle.handson.CantankerousInterceptor.
onEvent(CantankerousInterceptor.java:1)
    at com.tangosol.net.events.internal.NamedEventInterceptor.
onEvent(NamedEventInterceptor.java:240)
    at com.tangosol.net.events.internal.AbstractEvent.
nextInterceptor(AbstractEvent.java:116)
    at com.tangosol.net.events.internal.AbstractEvent.dispatch(AbstractEvent.
java:154)
    at com.tangosol.net.events.internal.AbstractEventDispatcher$1.
proceed(AbstractEventDispatcher.java:254)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
PartitionedService$Continuations$Task.run(PartitionedService.CDB:6)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.
Service$EventDispatcher.onNotify(Service.CDB:26)
    at com.tangosol.coherence.component.util.Daemon.run(Daemon.CDB:51)
    at java.lang.Thread.run(Thread.java:722)

2014-01-02 16:07:33.238/48.761 Oracle Coherence GE 12.1.3.0.0 <Error>
(thread=DistributedCache:PartitionedPofCache:EventDispatcher, member=3): Exception
caught while dispatching to "<cantankerous, com.oracle.handson.
CantankerousInterceptor>": Objection falls on deaf ears! value = value: 22
    at com.oracle.handson.CantankerousInterceptor.
onEvent(CantankerousInterceptor.java:74)
    at com.oracle.handson.CantankerousInterceptor.
onEvent(CantankerousInterceptor.java:1)
    at com.tangosol.net.events.internal.NamedEventInterceptor.

```

```

onEvent (NamedEventInterceptor.java:240)
    at com.tangosol.net.events.internal.AbstractEvent.
nextInterceptor (AbstractEvent.java:116)
    at com.tangosol.net.events.internal.AbstractEvent.dispatch (AbstractEvent.
java:154)
    at com.tangosol.net.events.internal.AbstractEventDispatcher$1.
proceed (AbstractEventDispatcher.java:254)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
PartitionedService$Continuations$Task.run (PartitionedService.CDB:6)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.
Service$EventDispatcher.onNotify (Service.CDB:26)
    at com.tangosol.coherence.component.util.Daemon.run (Daemon.CDB:51)
    at java.lang.Thread.run (Thread.java:722)

```

...

```

2014-01-02 16:07:33.238/48.761 Oracle Coherence GE 12.1.3.0.0 <Error>
(thread=DistributedCache:PartitionedPofCache:EventDispatcher, member=3): Exception
caught while dispatching to "<cantankerous, com.oracle.handson.
CantankerousInterceptor>": Objection falls on deaf ears! value = value: 110
    at com.oracle.handson.CantankerousInterceptor.
onEvent (CantankerousInterceptor.java:74)
    at com.oracle.handson.CantankerousInterceptor.
onEvent (CantankerousInterceptor.java:1)
    at com.tangosol.net.events.internal.NamedEventInterceptor.
onEvent (NamedEventInterceptor.java:240)
    at com.tangosol.net.events.internal.AbstractEvent.
nextInterceptor (AbstractEvent.java:116)
    at com.tangosol.net.events.internal.AbstractEvent.dispatch (AbstractEvent.
java:154)
    at com.tangosol.net.events.internal.AbstractEventDispatcher$1.
proceed (AbstractEventDispatcher.java:254)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.service.grid.
PartitionedService$Continuations$Task.run (PartitionedService.CDB:6)
    at com.tangosol.coherence.component.util.daemon.queueProcessor.
Service$EventDispatcher.onNotify (Service.CDB:26)
    at com.tangosol.coherence.component.util.Daemon.run (Daemon.CDB:51)
    at java.lang.Thread.run (Thread.java:722)

```

...

```

2014-01-02 16:07:33.649/49.172 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Cluster, member=2): Member (Id=4, Timestamp=2014-01-02 16:07:33.649,
Address=10.159.154.103:8094, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8152, Role=OracleHandsonDriver) left
Cluster with senior member 1

```

...

## 11.4 Logging Partition Activity Using an Event Interceptor

In this exercise you will create an event interceptor to log partition events for a partitioned service. To complete this exercise, follow these steps:

1. [Create a Class to Terminate a JVM and to Enable and Disable Logging](#)
2. [Create an Event Interceptor to Log Partition Activity](#)
3. [Create a Class to Exercise the Log Partition Activity Example](#)
4. [Register the Log Partition Activity Event Interceptor](#)
5. [Edit the POF Configuration File](#)

6. [Edit the Driver File for the Log Partition Activity Example](#)
7. [Run the Log Partition Activity Example](#)

### 11.4.1 Create a Class to Terminate a JVM and to Enable and Disable Logging

Create a class named `RedistributionInvocable` that defines three actionable states that will be executed on various members of the cluster. For this example, define the states as follows:

- **DISABLE:** Disable the logging performed by the `RedistributionInterceptor` event interceptor.
- **ENABLE:** Enable the logging performed by the `RedistributionInterceptor` event interceptor.
- **KILL:** Terminate the JVM that this invocable (`RedistributionInvocable`) is executed on.

You will create the `RedistributionInterceptor` event interceptor in a later step.

For the variable that determines whether logging is enabled or disabled, use the `AtomicBoolean` class. For example:

```
public static final AtomicBoolean ENABLED = new AtomicBoolean(false)
```

To terminate the invocable, the **KILL** state can simply call `System.exit`.

The data that the class produces will be sent across the wire, so the class should use POF (Portable Object Format). You will add the class to the POF configuration file in a later step.

To create the `RedistributionInvocable` class:

1. Create a new Java class called `RedistributionInvocable`. Ensure that the **Default Package** is `com.oracle.handson`. Do not select the **Main Method** check box.

See ["Creating a Java Class"](#) on page 2-11 for more information.

2. Write the class to define three different states that can be assigned to an interceptor object. Import the `AbstractInvocable`, `AtomicBoolean`, `PortableObject`, `PofReader`, and `PofWriter` classes and interfaces. The `RedistributionInvocable` class should extend the `AbstractInvocable` class and implement the `PortableObject` interface. You can write your own `RedistributionInvocable` class or use the code provided in [Example 11-13](#).

#### **Example 11-13 Class to Terminate a JVM and to Enable or Disable Logging**

```
package com.oracle.handson;

import com.tangosol.io.pof.PofReader;
import com.tangosol.io.pof.PofWriter;
import com.tangosol.io.pof.PortableObject;

import com.tangosol.net.AbstractInvocable;

import java.io.IOException;

import java.util.concurrent.atomic.AtomicBoolean;

/**
 * RedistributionInvocable has three states in which appropriate action is
 * taken:
```

```

* <ol>
*   <li><strong>{@link State#DISABLE}</strong> - Disables the logging
*   performed by {@link com.oracle.handson.RedistributionInterceptor}.</li>
*   <li><strong>{@link State#ENABLE}</strong> - Enables the logging
*   performed by {@link com.oracle.handson.RedistributionInterceptor}.</li>
*   <li><strong>{@link State#KILL}</strong> - Kills the JVM this invocable
*   is executed on.</li>
* </ol>
*
*
* @since 12.1.2
*/
public class RedistributionInvocable
    extends AbstractInvocable
    implements PortableObject
{
    // ----- constructors -----

    /**
     *
     */
    private static final long serialVersionUID = 1L;

    /**
     * Default no-arg constructor.
     */
    public RedistributionInvocable()
    {
        this(State.DISABLE);
    }

    /**
     * Constructs a RedistributionInvocable with the specified state.
     *
     * @param state the state indicating the action to be performed
     */
    public RedistributionInvocable(State state)
    {
        m_state = state;
    }

    // ----- Invocable methods -----

    /**
     * {@inheritDoc}
     */
    public void run()
    {
        switch (m_state)
        {
            case DISABLE:
                ENABLED.set(false);
                break;
            case ENABLE:
                ENABLED.set(true);
                break;
            case KILL:
                System.exit(1);
        }
    }
}

```

```

// ----- PortableObject methods -----

/**
 * {@inheritDoc}
 */
public void readExternal(PofReader in) throws IOException
{
    m_state = State.values()[in.readInt(0)];
}

/**
 * {@inheritDoc}
 */
public void writeExternal(PofWriter out) throws IOException
{
    out.writeInt(0, m_state.ordinal());
}

// ----- inner class: State -----

/**
 * Representation of the action to be performed when
 * {@link RedistributionInvocable#run()}.
 */
public enum State
{
    /**
     * Disables the logging performed by
     * {@link com.oracle.handson.RedistributionInterceptor}
     */
    DISABLE,
    /**
     * Enables the logging performed by
     * {@link com.oracle.handson.RedistributionInterceptor}
     */
    ENABLE,
    /**
     * Terminates the JVM process in which the
     * {@link RedistributionInvocable} is executed.
     */
    KILL
}

// ----- constants -----

/**
 * Flag used to determine whether to log partition events.
 */
public static final AtomicBoolean ENABLED = new AtomicBoolean(false);

// ----- data members -----

/**
 * The state used to determine which action to perform.
 */
private State m_state;
}

```

## 11.4.2 Create an Event Interceptor to Log Partition Activity

Create an event interceptor named `RedistributionInterceptor` to log partition events for a partitioned service.

To create the `RedistributionInterceptor` event interceptor:

1. Create a new Java class called `RedistributionInterceptor`. Ensure that the **Default Package** is `com.oracle.handson`. Do not select the **Main Method** check box.

See ["Creating a Java Class"](#) on page 2-11 for more information.

2. Write the `RedistributionInterceptor` class to log partition events. Import the `RedistributionInvocable`, `CacheFactory`, `EventInterceptor`, `Interceptor`, `TransferEvent` classes and interfaces. The `RedistributionInterceptor` class should implement the `EventInterceptor<TransferEvent>` interface. You can write your own class to log partition events or use the code provided in [Example 11-14](#).

[Example 11-14](#) illustrates an event interceptor to log partition events. A name can be assigned to the interceptor by using the optional `identifier` attribute in the `@Interceptor` annotation. The event interceptor determines whether the partition event should be logged by referencing the value of the `RedistributionInvocable.ENABLED` constant.

### Example 11-14 Class to Log Partition Events

```
package com.oracle.handson;

import com.oracle.handson.RedistributionInvocable;

import com.tangosol.net.CacheFactory;

import com.tangosol.net.events.EventInterceptor;
import com.tangosol.net.events.annotation.Interceptor;
import com.tangosol.net.events.partition.TransferEvent;

/**
 * RedistributionInterceptor is an {@link
 * com.tangosol.net.events.EventInterceptor}
 * that logs partition activity when enabled. Logging can be enabled via
 * setting the {@link RedistributionInvocable#ENABLED} constant.
 *
 * @since Coherence 12.1.2
 */
@Interceptor(identifier = "redist")
public class RedistributionInterceptor
    implements EventInterceptor<TransferEvent>
{
    // ----- EventInterceptor methods -----

    /**
     * {@inheritDoc}
     */
    public void onEvent(TransferEvent event)
    {
        if (RedistributionInvocable.ENABLED.get())
        {
            CacheFactory.log(String.format("Discovered event %s for partition-id

```

```

    %d from remote member %s\n",
        event.getType(), event.getPartitionId(), event.getRemoteMember()),
        CacheFactory.LOG_INFO);
    }
}
}

```

### 11.4.3 Create a Class to Exercise the Log Partition Activity Example

Create a class called `LogExample` to trigger actions to be performed by the `RedistributionInterceptor` class.

To create the `LogExample` class:

1. Create a new Java class called `LogExample`. Do not include a main method. See ["Creating a Java Class"](#) on page 2-11 for more information.
2. Write the code for the `LogExample` class. Import the `RedistributionInvocable`, `RedistributionInvocable.State`, `CacheFactory`, `InvocationService`, `Member`, `ArrayList`, `Collection`, `Random`, `Set` and `Callable` classes and interfaces. You can write your own code or use the code supplied in [Example 11-15](#).

[Example 11-15](#) illustrates a sample implementation of the `LogExample` class. The example contains the subclass `RedistributionEventsExample` subclass which triggers actions to be performed by the `RedistributionInterceptor` class. The subclass illustrates how partition redistribution events can be logged. At least two cluster members must be running to run this example.

#### **Example 11-15 Sample Class to Exercise the Log Partition Activity Example**

```

package com.oracle.handson;

import com.oracle.handson.RedistributionInvocable;
import com.oracle.handson.RedistributionInvocable.State;
import com.tangosol.net.CacheFactory;
import com.tangosol.net.InvocationService;
import com.tangosol.net.Member;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Random;
import java.util.Set;
import java.util.concurrent.Callable;

/**
 * LogExample illustrates logging of partition movement when enabled.
 *
 * @since Coherence 12.1.2
 */
@SuppressWarnings("unchecked")
public class LogExample
{
    /**
     * The RedistributionEventsExample is a catalyst for action to be
     * performed by {@link RedistributionInterceptor}. This illustrates how
     * partition redistribution events can be logged.
     */
    public static class RedistributionEventsExample
        implements Callable<Boolean>

```

```

    {
        // ----- Callable methods -----

        /**
         * {@inheritDoc}
         */
        public Boolean call() throws Exception
        {
            // transfer events
            try
            {
                InvocationService is = (InvocationService) CacheFactory.
getService("InvocationService");
                Random rnd = new Random();
                int cMembers = is.getInfo().getServiceMembers().
size();

                if (cMembers < 3)
                {
                    System.err.println("<Error> At least two members must exist
for the RedistributionEvent example");
                    return false;
                }

                // enable the logging of transfer event
                is.query(new RedistributionInvocable(State.ENABLE), null);

                Set<Member> isMembers = is.getInfo().getServiceMembers();
                isMembers.remove(is.getCluster().getLocalMember());

                Member memChosen = new ArrayList<Member>(isMembers).get(rnd.
nextInt(isMembers.size()));

                System.out.printf("Choosing to kill member %s\n", memChosen);
                is.query(new RedistributionInvocable(State.KILL), Collections.
singleton(memChosen));
            }
            finally
            {
            }

            return true;
        }
    }
}

```

#### 11.4.4 Register the Log Partition Activity Event Interceptor

The event interceptors can be registered either programmatically or by including references to them in the cache configuration file.

In the UEMEvents project, the interceptors are registered in the cache configuration file. The fully-qualified class name of the event interceptor is specified in the `<interceptor>` element. The interceptor is associated with the cache specified in the `<cache-name>` element.

For the log partition events example, the event interceptor, `RedistributionInterceptor`, is registered on the partitioned cache service under the `<distributed-scheme>` element.

To edit the cache configuration file to define the `RedistributionInterceptor` event interceptor:

1. Open the `uem-cache-config.xml` file from the Project Explorer window. You can find the file under `Events/appClientModule`.
2. Write the cache configuration that calls the event interceptors. Under the `<distributed-scheme>` element, there should be a reference to the fully-qualified `RedistributionInterceptor` class in the `<interceptor>` element.

[Example 11-16](#) illustrates a possible implementation for the `uem-cache-config.xml` file. The configuration for the `RedistributionInterceptor` event interceptor is illustrated in bold font.

**Example 11-16 Cache Configuration File with Event Interceptors**

```
<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
              xsi:schemaLocation="http://xmlns.oracle.
com/coherence/coherence-cache-config coherence-cache-config.xsd">
  <defaults>
    <serializer>pof</serializer>
  </defaults>

  <caching-scheme-mapping>
    <cache-mapping>
      <cache-name>events</cache-name>
      <scheme-name>events-distributed-scheme</scheme-name>
      <interceptors>
        <interceptor>
          <instance>
            <class-name>com.oracle.handson.
TimedTraceInterceptor</class-name>
            <init-params>
              <init-param>
                <param-type>int</param-type>
                <param-value>100</param-value>
              </init-param>
            </init-params>
          </instance>
        </interceptor>
      </interceptors>
    </cache-mapping>
    <cache-mapping>
      <cache-name>vetod-events</cache-name>
      <scheme-name>events-distributed-scheme</scheme-name>
      <interceptors>
        <interceptor>
          <instance>
            <class-name>com.oracle.handson.
CantankerousInterceptor</class-name>
          </instance>
        </interceptor>
      </interceptors>
    </cache-mapping>
    <cache-mapping>
      <cache-name>events-results</cache-name>
      <scheme-name>dist-events-results</scheme-name>
    </cache-mapping>
```

```

</caching-scheme-mapping>

<caching-schemes>
  <distributed-scheme>
    <scheme-name>events-distributed-scheme</scheme-name>
    <service-name>PartitionedPofCache</service-name>
    <thread-count>5</thread-count>
    <backing-map-scheme>
      <local-scheme>
        <!-- each node will be limited to 32MB -->
        <high-units>32M</high-units>
        <unit-calculator>binary</unit-calculator>
      </local-scheme>
    </backing-map-scheme>
    <autostart>true</autostart>
    <interceptors>
      <interceptor>
        <instance>
          <class-name>com.oracle.handson.
RedistributionInterceptor</class-name>
        </instance>
      </interceptor>
    </interceptors>
  </distributed-scheme>

  <!-- A PartitionedCache service used to store results for events examples
  -->
  <distributed-scheme>
    <scheme-name>dist-events-results</scheme-name>
    <service-name>PartitionedEventsResults</service-name>
    <thread-count>5</thread-count>
    <backing-map-scheme>
      <local-scheme/>
    </backing-map-scheme>
    <autostart>true</autostart>
  </distributed-scheme>

  <!--
  Invocation Service scheme.
  -->
  <invocation-scheme>
    <scheme-name>examples-invocation</scheme-name>
    <service-name>InvocationService</service-name>

    <autostart system-property="tangosol.coherence.invocation.
autostart">true</autostart>
  </invocation-scheme>

</caching-schemes>
</cache-config>

```

## 11.4.5 Edit the POF Configuration File

With the exception of the `RedistributionInvocable` class, all of the information produced by the classes in the log partition activity exercise remain on their own cluster members. The information produced by the `RedistributionInvocable` class however, will be sent across the wire to other cluster members. Thus, it must be added to the POF configuration file.

To edit the POF configuration file for the `RedistributionInvocable` data type:

1. Open the `uem-pof-config.xml` file. You can find the file under `UEMEvents/appClientModule/META-INF`.
2. Define `<user-type>` elements for the `com.oracle.handson.RedistributionInvocable`, class and assign type ID 1009 to it. The file must include the `coherence-pof-config.xml` file which reserves the first 1000 IDs for Coherence data types.

**Example 11-17** illustrates a sample `uem-pof-config.xml` file. The configuration for the `RedistributionInvocable` class is illustrated in bold font.

**Example 11-17 POF Configuration File for the Log Partition Events Example**

```
<?xml version="1.0"?>
<pof-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://xmlns.oracle.com/coherence/coherence-pof-config"
  xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-pof-config
http://xmlns.oracle.com/coherence/coherence-pof-config/1.2/coherence-pof-config.
xsd">
  <user-type-list>
    <!-- include all "standard" Coherence POF user types -->
    <include>coherence-pof-config.xml</include>
    <user-type>
      <type-id>1008</type-id>
      <class-name>com.oracle.handson.LazyProcessor</class-name>
    </user-type>
    <user-type>
      <type-id>1009</type-id>
      <class-name>com.oracle.handson.RedistributionInvocable</class-name>
    </user-type>
  </user-type-list>
</pof-config>
```

## 11.4.6 Edit the Driver File for the Log Partition Activity Example

Edit the driver file that you created in "[Create a Driver File for Timed Events Example](#)" on page 11-16 to run the `RedistributionEventsExample` example defined in the `LogExample` class.

To edit the driver file:

1. Replace the import statement for the `Vetoed Events` example:

```
import com.oracle.handson.VetoExample.VetoedEventsExample;
```

with an import statement for the `RedistributionEventsExample` subclass of the `LogExample` class.

```
import com.oracle.handson.LogExample.RedistributionEventsExample;
```

2. Replace the command which calls the `Vetoed Events` example

```
EVENTS_EXAMPLES.put("veto interceptor", new VetoedEventsExample());
```

with the following command to run the `RedistributionEventsExample` example defined in the `LogExample` class:

```
EVENTS_EXAMPLES.put("redistribution interceptor", new
RedistributionEventsExample());
```

## 11.4.7 Run the Log Partition Activity Example

Right-click the **UEMEEvents** project in the Project Explorer and select **Run As**, then **Run Configurations**. In the **Run Configurations** dialog box, select the **UEMEEventsServer** launch configuration that you created in "[Create a Cache Server Startup Configuration](#)" on page 11-17. Click **Run** to start the cache server. After the cache server starts, select **UEMEEventsServer** and click **Run** a second and a third time to start a total of three cache servers.

After the third cache server starts, select the **UEMEEventDriver** configuration and click **Run**. The output from the cache client should look similar to [Example 11-18](#).

### **Example 11-18** Output from the Cache Client

```
----- events examples begin -----
----- redistribution interceptor begin
2014-01-02 16:38:38.901/0.350 Oracle Coherence 12.1.3.0.0 <Info> (thread=main,
member=n/a): Loaded operational configuration from
"jar:file:/C:/Oracle/coherence/lib/coherence.jar!/tangosol-coherence.xml"

...

2014-01-02 16:38:43.249/4.698 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:InvocationService, member=4): Service InvocationService joined
the cluster with senior service member 1
Choosing to kill member Member(Id=3, Timestamp=2014-01-02 16:38:17.942,
Address=10.159.154.103:8092, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8168, Role=CoherenceServer)
Choosing to kill member Member(Id=3, Timestamp=2014-01-02 16:38:17.942,
Address=10.159.154.103:8092, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8168, Role=CoherenceServer)
----- redistribution interceptor completed successfully
----- events examples completed-----
2014-01-02 16:38:43.276/4.725 Oracle Coherence GE 12.1.3.0.0 <D5> (thread=Cluster,
member=4): TcpRing disconnected from Member(Id=3, Timestamp=2014-01-02 16:38:17.
942, Address=10.159.154.103:8092, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8168, Role=CoherenceServer) due to a
peer departure; removing the member.

...
```

[Example 11-19](#) displays the output from the cache server that was terminated. The output illustrates the client joining the cluster (Member 4) and the shutdown of the current cache server.

### **Example 11-19** Output from First Cache Server

```
...
2014-01-02 16:38:20.701/6.178 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache:PartitionedEventsResults, member=3): Transferring 1B of
backup[1] for PartitionSet{0} to member 2
2014-01-02 16:38:42.299/27.776 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Cluster, member=3): Member(Id=4, Timestamp=2014-01-02 16:38:42.09,
Address=10.159.154.103:8094, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:1620, Role=OracleHandsonDriver) joined
Cluster with senior member 1
2014-01-02 16:38:42.495/27.972 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:Management, member=3): Member 4 joined Service Management with
senior member 1
2014-01-02 16:38:43.254/28.731 Oracle Coherence GE 12.1.3.0.0 <D5>
```

```
(thread=Invocation:InvocationService, member=3): Member 4 joined Service  
InvocationService with senior member 1  
2014-01-02 16:38:43.254/28.731 Oracle Coherence GE 12.1.3.0.0 <D4>  
(thread=ShutdownHook, member=1): ShutdownHook: stopping cluster node
```

---

---

## Working with JCache

In this exercise, you learn how to use JCache, the Java standard APIs for caching on the Java platform. This exercise is similar to [Chapter 3, "Accessing the Data Grid from Java"](#) where you created a Java console-based application to access, update, and remove simple types of information from a Coherence clustered cache. However, instead of using `NamedCache` and the Coherence API, it uses `Cache` and the JCache API.

This chapter has the following sections:

- [Introduction](#)
- [Creating a JCache-Based Java Project](#)
- [Creating a JCache-Based Application to Put Values in the Cache](#)
- [Running a JCache Application in a Cluster](#)

### 12.1 Introduction

Coherence includes a JCache provider implementation. JCache is a common API for using caching in Java. You can use the JCache API and Coherence provides the underlying caching capabilities. The provider-based approach guarantees cross-provider portability and allows developers to focus on application logic rather than creating and managing complex cache subsystems.

The Coherence JCache provider is built upon the existing capabilities of Coherence. The provider relies on the Coherence infrastructure and can be thought of as a wrapper for the Coherence `NamedCache` API. This allows Coherence to reuse and expose many of its best-in-class technologies using JCache interfaces.

The key files which support JCache are `cache-api.jar` and `coherence-jcache.jar`. The `cache-api.jar` file contains the JCache libraries. The `coherence-jcache.jar` file contains the Coherence implementation which is built on top of the JCache library. Both of these files must appear on the classpath of the application.

The APIs in the `coherence-jcache.jar` file allow JCache to support the use of distributed, local, passthrough, and remote caches. It also supports the use of POF serialization, events, and entry processors.

For more information on the JCache provider implementation, see "Introduction to Coherence JCache" in *Oracle Fusion Middleware Developing Applications with Oracle Coherence*.

Oracle is a lead contributor of the JCache specification. See the following URL to obtain the JCache specification (JSR-107 "Java Caching API"):

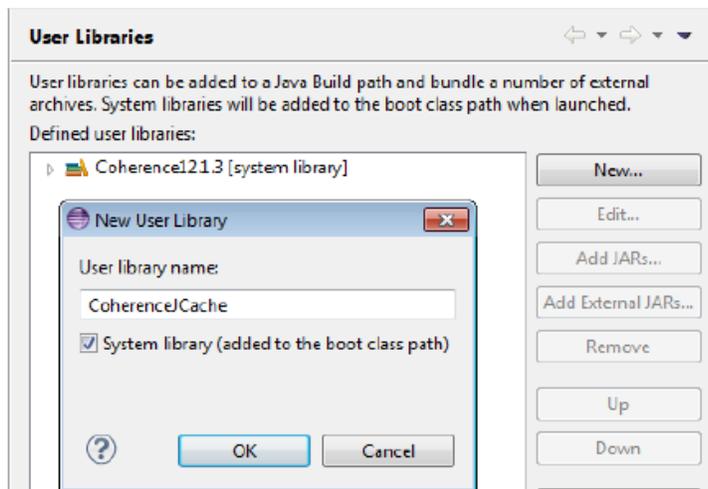
<https://jcp.org/en/jsr/detail?id=107>

## 12.2 Creating a JCache-Based Java Project

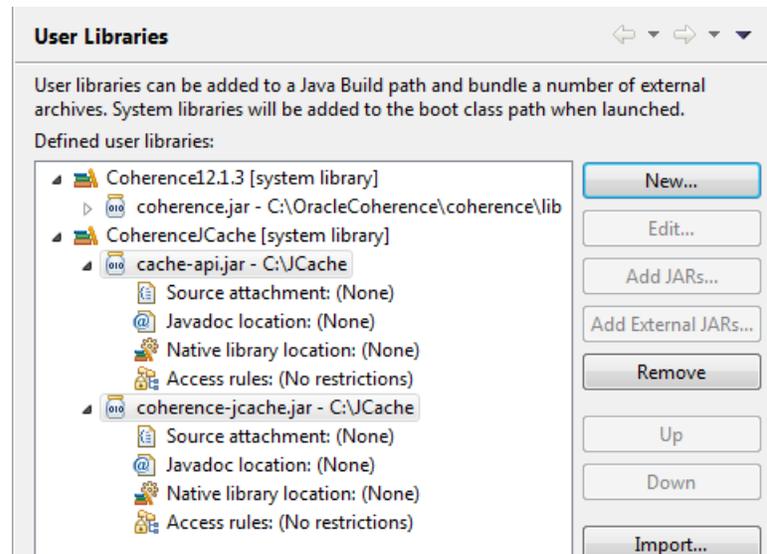
Follow these steps to create a project that uses JCache.

1. In the Eclipse IDE, select **File**, then **New**, then **Application Client Project** to create a new project called JCache. Select **CoherenceConfig** from the **Configuration** drop-down list. Click **Next**.
2. Click **Next** to accept the defaults in the Java page of the wizard. In the Application Client module page, deselect **Create a default Main class**. Click **Next**.
3. Click the **Manage Libraries** icon in the Coherence page to create a user library to contain the JCache and Coherence JCache libraries.
4. Click **New** in the User Libraries dialog box to name the user library. In the New User Library dialog box, enter CoherenceJCache and select the **System Library** checkbox as illustrated in [Figure 12-1](#). Click **OK**.

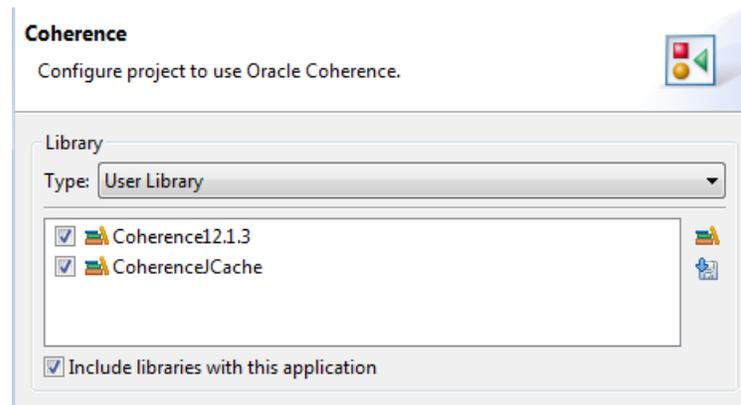
**Figure 12-1** Creating the CoherenceJCache User Library



5. Select the CoherenceJCache library in the User Libraries dialog box and click **Add External JARs**.
6. Navigate to the location of the `/coherence/lib` folder in your Coherence distribution and select the `cache-api.jar` and the `coherence-jcache.jar` files. When you are finished, the User Libraries dialog box should look similar to [Figure 12-2](#). Click **OK**.

**Figure 12–2 Adding Coherence JCache JARS to the User Libraries**

7. Select the **CoherenceJCache** library in the Coherence page. The Coherence page should look similar to [Figure 12–3](#). Click **Finish**.

**Figure 12–3 Coherence Configuration Page for a JCache Project**

## 12.3 Creating a JCache-Based Application to Put Values in the Cache

This section describes how to create a Java program that uses the JCache APIs to access and update simple types of information from a Coherence clustered cache.

To create a JCache-based application:

1. Create a new Application Client Project in Eclipse. Name the project JCache. Ensure that the folder is C:\home\oracle\workspace\JCache.

In the Configuration section of the **New Application Client** Project dialog box, click **Modify**. In the **Project Facets** dialog box, select **CoherenceConfig** from the **Configuration** drop down list.

See ["Creating a New Project in the Eclipse IDE"](#) on page 2-3 for detailed instructions.

2. Create your first Coherence JCache Java program. Name the class `MyFirstJCacheSample` and select the **public static void main(String[] args)** check box in the **New Java Class dialog box**.
3. In the Eclipse editor, write the code to create a Cache object, enter a value in the cache, and then verify the value that you entered.

Notice that in JCache, as opposed to Coherence, you do need to write more code to create a cache. In JCache, you must first get the cache manager and set the cache configuration parameters before you can create the cache.

[Example 12-1](#) illustrates a sample program.

**Example 12-1 Creating a JCache Cache Object: Inserting and Verifying Values**

```
package com.oracle.handson;

import javax.cache.Cache;
import javax.cache.CacheManager;
import javax.cache.Caching;
import javax.cache.configuration.MutableConfiguration;
import javax.cache.spi.CachingProvider;

public class MyFirstJCacheSample {
    public MyFirstJCacheSample() {
    }
    public static void main(String[] args) {

        // ensure we are in a cluster
        CacheFactory.ensureCluster();

        //get cache manager
        CachingProvider cachingProvider = Caching.getCachingProvider();
        CacheManager cacheManager = cachingProvider.getCacheManager();

        //set configuration for the cache
        PassThroughCacheConfiguration<String, String> config = new
        PassThroughCacheConfiguration();
        config.setTypes(String.class, String.class);

        //create and use the cache
        cacheManager.createCache("myCache", config);
        Cache<String, String> cache = cacheManager.getCache("myCache",
        String.class, String.class);

        // put key, value pair into the cache.
        cache.put("Name", "Gene Smith");

        System.out.println("Value in cache is " + cache.get("Name"));
    }
}
```

4. Stop any running cache servers. See ["Stopping Cache Servers"](#) on page 2-14 for detailed information.
5. Run the program in the Eclipse IDE:
  - a. Right-click the `MyFirstJCacheSample.java` class in the editor and select **Run As** then **Run Configuration**. Double-click Oracle Coherence to create a

Coherence configuration. Enter `MyFirstJCacheSample` in the **Name** field of the **Run Configuration** dialog box.

- b. In the **Main** tab, enter `JCache` in the **Project** field and `com.oracle.handson.MyFirstJCacheSample` in the **Main class** field.
- c. In the **Coherence** tab, enter a unique value, such as `3155`, in the **Cluster port** field to ensure that Coherence is limited to your own host. Select **Enabled (cache server)** under **Local storage**.
- d. Note that in the **Classpath** tab, the **CoherenceJCache** library should appear before the **Coherence12.1.3** library in the **Bootstrap Entries** list. Click **Apply**, then **Run**.

Messages similar to [Example 12–2](#) are displayed:

#### **Example 12–2 Output of MyFirstJCacheSample Program**

```
2014-02-20 15:28:53.188/0.377 Oracle Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=n/a): Loaded operational configuration from
"jar:file:/C:/OracleCoherence1/coherence/lib/coherence.jar!/tangosol-coherence.
xml"
2014-02-20 15:28:53.303/0.492 Oracle Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=n/a): Loaded operational overrides from
"jar:file:/C:/OracleCoherence1/coherence/lib/coherence.jar!/tangosol-coherence-
override-dev.xml"
2014-02-20 15:28:53.383/0.572 Oracle Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=n/a): Loaded operational overrides from
"file:/C:/home/oracle/workspace/JCache/build/classes/tangosol-coherence-override.
xml"
...
2014-02-20 15:28:53.668/0.857 Oracle Coherence GE 12.1.3.0.0 <D5> <Info>
(thread=main, member=n/a): Loaded cache configuration from
"jar:file:/C:/OracleCoherence1/coherence/lib/coherence-jcache.jar!/coherence-
jcache-cache-config.xml"
2014-02-20 15:28:53.848/1.037 Oracle Coherence GE 12.1.3.0.0 <D5> <Info>
(thread=main, member=n/a): Created cache factory com.tangosol.net.
ExtensibleConfigurableCacheFactory
2014-02-20 15:28:53.863/1.052 Oracle Coherence GE 12.1.3.0.0 <D5> <Info>
(thread=main, member=n/a): Mapping general javax.cache.Configuration
implementation to CoherenceBased JCacheConfiguration of com.tangosol.coherence.
jcache.localcache.LocalCacheConfiguration
Value in cache is Gene Smith
```

## 12.4 Running a JCache Application in a Cluster

This section describes how to create and run an application that uses the JCache API in a cluster environment.

1. [Configure an Example Cluster](#)
2. [Store the Object in a Pass-Through Cache](#)
3. [Start the Example Cache Server](#)
4. [Run the Application](#)
5. [Verify the Cache](#)

## 12.4.1 Configure an Example Cluster

Partitioned caches and pass-through caches use a Coherence cluster to distribute cached data. This task creates an operational override file to modify the out-of-box default cluster configuration. In particular, the default configuration is modified to create a private cluster which ensures that the JVM processes do not attempt to join an existing Coherence cluster that may be running on the network.

To configure an example cluster:

1. Edit the `tangosol-coherence-override.xml` file. Double-click the `tangosol-coherence-override.xml` file in the Project Navigator to open it in the editor.
2. Add the following override configuration and replace `cluster_name` and `port` with values that are unique for this cluster. For example, use `myCluster` for the cluster name and the last four digits of your telephone number for the port.

```
<?xml version='1.0'?>

<coherence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://xmlns.oracle.com/coherence/coherence-operational-config"
  xsi:schemaLocation="http://xmlns.oracle.com/coherence/
coherence-operational-config coherence-operational-config.xsd">
  <cluster-config>
    <member-identity>
      <cluster-name>cluster_name</cluster-name>
    </member-identity>

    <multicast-listener>
      <address>224.3.6.0</address>
      <port>port</port>
      <time-to-live>0</time-to-live>
    </multicast-listener>
  </cluster-config>

</coherence>
```

3. Save and close the file.

## 12.4.2 Store the Object in a Pass-Through Cache

A pass-through cache is a cache that delegates to a pre-existing Coherence cache (a cache that is defined in a Coherence cache configuration file). Pass-through caches allow you to use all of the native features of Coherence and provide greater control over cache configuration. The use of pass-through caching requires Coherence-specific changes to the client code and is therefore not intended for JCache applications that are concerned about portability.

In this example, two separate Java processes form the cluster: a cache server process and the `MyFirstJCacheSample` application process. For simplicity, the two processes are collocated on a single computer. The cache server, by default, is configured to store cache data. A Coherence `CacheFactory` is used to verify that the `MyFirstJCacheSample` application successfully created and loaded the cache on the cluster.

1. [Modify the Sample JCache Application](#)
2. [Define the Sample Cache for JCache](#)

### 12.4.2.1 Modify the Sample JCache Application

The `PassThroughCacheConfiguration` class is an implementation-specific configuration that provides a JCache interface to a native Coherence cache. The configuration is used instead of the standard `MutableConfiguration` class.

To configure a cache as a pass-through cache:

1. Open the `MyFirstJCacheSample` class that you created from [Example 12-1](#).
2. Modify the class to use a `PassThroughCacheConfiguration` configuration. For example:

```
...
//set configuration for the cache
PassThroughCacheConfiguration<String, String> config = new
PassThroughCacheConfiguration();
    config.setTypes(String.class, String.class);

//create and use the cache
cacheManager.createCache("myCache", config);
Cache<String, String> cache = cacheManager.getCache("myCache",
    String.class, String.class);
...
```

3. Save the file.

### 12.4.2.2 Define the Sample Cache for JCache

For this example, a cache configuration is created that defines a distributed cache that is explicitly mapped to the `myCache` name.

In the Eclipse environment, the cache configuration for an application which uses the JCache APIs must be stored in the `coherence-jcache-cache-config.xml` file.

To define the example cache:

1. Obtain the `coherence-jcache-cache-config.xml` file by using your favorite file compression utility to extract it from the `coherence-jcache.jar` file. Extract the file to the `JCache\appClientModule` folder.
2. Right-click the JCache project in the Project Explorer and select **Refresh**. The `coherence-jcache-cache-config.xml` file should appear in the list of files under the `JCache\appClientModule` folder.
3. Double-click the `coherence-jcache-cache-config.xml` file to open it in the editor.

Notice that in addition to the namespaces for the XML schema instance and the Coherence cache configuration, the file also contains the definition for the namespace for the JCache namespace (illustrated in bold).

```
<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
    xmlns:jcache="class://com.tangosol.coherence.jcache.JCacheNamespace"
    xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-cache-
config coherence-cache-config.xsd">
...
```

4. Copy the following distributed cache definition to the file:

```
<?xml version="1.0"?>

<cache-config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://xmlns.oracle.com/coherence/coherence-cache-config"
    xmlns:jcache="class://com.tangosol.coherence.jcache.JCacheNamespace"
```

```

        xsi:schemaLocation="http://xmlns.oracle.com/coherence/coherence-cache-
config
coherence-cache-config.xsd">
<キャッシング-スキームマッピング>
  <キャッシュマッピング>
    <キャッシュ名>myCache</キャッシュ名>
    <スキーム名>distributed</スキーム名>
  </キャッシュマッピング>
</キャッシング-スキームマッピング>

<キャッシングスキーム>
  <distributedスキーム>
    <スキーム名>distributed</スキーム名>
    <サービス名>DistributedCache</サービス名>
    <バックアップスキーム>
      <localスキーム/>
    </バックアップスキーム>
    <自動起動>true</自動起動>
  </distributedスキーム>
</キャッシングスキーム>
</cache-config>

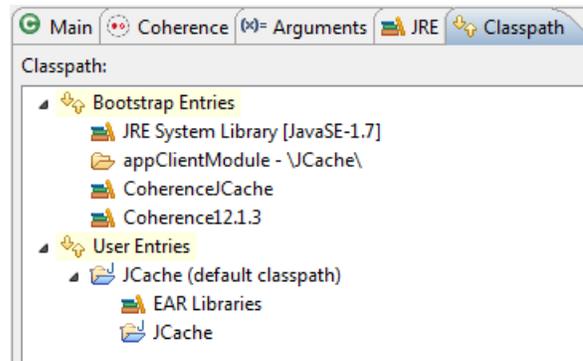
```

5. Save and close the file.

### 12.4.3 Start the Example Cache Server

Create a run configuration for the Coherence default cache server.

1. Right click the JCache project in the Eclipse IDE. Select **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, select **Oracle Coherence** then the **New launch configuration** icon. Enter `DefaultCacheServer` as the name for the cache server configuration.
2. Under **Project**, click **Browse** and select the name of the JCache project from the **Project Selection** dialog box.
3. Under **Main class**, select the **Include system libraries when searching for a main class** checkbox. Click the **Search** button and enter `DefaultCacheServer` in the **Select Main Type** dialog box. Select **com.tangosol.net.DefaultCacheServer** and click **OK**. Click **Apply**.
4. In the **Coherence** tab, select the **General** tab. Click the **Browse** icon to navigate to the cache configuration file `coherence-jcache-cache-config.xml`. Select local storage to be **Enabled** (cache server). Enter a unique value, such as 3155 for the **Cluster port**. Click **Apply**.
5. Open the **Arguments** tab. Enter `-showversion` in the **VM Arguments** field. Click **Apply**.
6. Open the **Classpath** tab of the dialog box. Click **Advanced**, then **Add Folders** to add the `JCache\appClientModule` folder (which contains the `tangosol-coherence-override.xml` override file that you configured in [Section 12.4.1, "Configure an Example Cluster"](#)) to the **Bootstrap Entries** section of the classpath. Use the **Up** and **Down** buttons to position `appClientModule` before the `CoherenceJCache` and `Coherenc12.1.3` libraries. This is because the compiler must encounter the `tangosol-coherence-override.xml` override file before the `coherence.jar` library. The **Classpath** tab should look similar to [Figure 12-4](#).

**Figure 12–4 Classpath for the DefaultCacheServer**

7. Open the **Common** tab of the dialog box. Click the **Shared file** radio button and click **Browse** to navigate to the project. Click **Apply**.
8. Click **Run** to start the cache server. The cache server should start and display output similar to [Example 12–3](#).

**Example 12–3 Output of the DefaultCacheServer**

```

java version "1.7.0_25"
Java(TM) SE Runtime Environment (build 1.7.0_25-b17)
Java HotSpot(TM) 64-Bit Server VM (build 23.25-b01, mixed mode)
...
2014-02-24 15:44:27.381/0.615 Oracle Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=n/a): Loaded operational overrides from
"file:/C:/home/oracle/workspace/JCache/appClientModule/tangosol-coherence-
override.xml"
...
2014-02-24 15:44:27.710/0.944 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main,
member=n/a): Loaded cache configuration from
"file:/C:/home/oracle/workspace/JCache/appClientModule/coherence-jcache-cache-
config.xml"
...
2014-02-24 15:44:34.557/7.791 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache, member=1): Service DistributedCache joined the cluster
with senior service member 1
2014-02-24 15:44:34.598/7.832 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main,
member=1):
Services
(
  ClusterService{Name=Cluster, State=(SERVICE_STARTED, STATE_JOINED), Id=0,
Version=12.1.3, OldestMemberId=1}
  InvocationService{Name=Management, State=(SERVICE_STARTED), Id=2, Version=12.1.
3, OldestMemberId=1}
  PartitionedCache{Name=DistributedCache, State=(SERVICE_STARTED),
LocalStorage=enabled, PartitionCount=257, BackupCount=1, AssignedPartitions=0,
BackupPartitions=0, CoordinatorId=1}
)
Started DefaultCacheServer...
...

```

**12.4.4 Run the Application**

To create a run configuration for the `MyFirstJCacheSample` class:

1. Right-click the JCache project in the Eclipse IDE. Select **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, select **Oracle Coherence** then the **New launch configuration** icon. Enter `MyFirstJCacheSample` as the name for the cache server configuration. Ensure that JCache appears in the **Project** field and `com.oracle.handson.MyFirstJCacheSample` appears in the **Main class** field. Click **Apply**.
2. In the **Coherence** tab, navigate to the `coherence-jcache-cache-config.xml` JCache cache configuration file in the **Cache configuration descriptor** field. Select **Disabled (cache client)** in the **Local storage** field. Enter a unique value such as 3155 in the **Cluster port** field.

---

**Note:** The value for the **Cluster port** must match the value you provided for the `<port>` attribute in the `tangosol-coherence-override.xml` file.

---

3. In the **Arguments** tab, enter the `-Dtangosol.coherence.jcache.configuration.uri` system property in the **VM arguments** field to specify the path to the JCache cache configuration file `coherence-jcache-cache-config.xml`. For example:
 

```
-Dtangosol.coherence.jcache.configuration.uri=C:\home\oracle\workspace\JCache\appClientModule\coherence-jcache-cache-config.xml
```
4. Open the **Classpath** tab of the dialog box. Click **Advanced**, then **Add Folders** to add the `JCache\appClientModule` folder (which contains the `tangosol-coherence-override.xml` override file that you configured in [Section 12.4.1, "Configure an Example Cluster"](#)) to the **Bootstrap Entries** section of the classpath. Use the **Up** and **Down** buttons to position `appClientModule` before the `CoherenceJCache` and `Coherenc12.1.3` libraries. This is because the Coherence compiler must encounter the `tangosol-coherence-override.xml` override file before the `coherence.jar` library. The **Classpath** tab should look similar to [Figure 12-4](#).
5. Open the **Common** tab of the dialog box. Click the **Shared file** radio button and click **Browse** to navigate to the JCache project. Click **Apply**.
6. Click **Run** to start the `MyFirstJCacheSample` application.

Notice that the `tangosol-coherence-override.xml` and `coherence-jcache-cache-config.xml` files were loaded from the `appClientModule` folder. The application process connects to the cluster that contains the cache server process and both processes are running the `DistributedCache` service. The application places the value `Gene Smith` in the cache, prints it to standard output, then exits the cluster.

#### **Example 12-4 Output of the MyFirstJCacheSample Application**

```
...
2014-02-25 11:09:24.145/0.630 Oracle Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=n/a): Loaded operational overrides from
"file:/C:/home/oracle/workspace/JCache/appClientModule/tangosol-coherence-
override.xml"
...
2014-02-25 11:09:27.078/3.563 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:Management, member=2): Service Management joined the cluster
with senior service member 1
...
```

```
2014-02-25 11:09:27.833/4.323 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main,
member=2): Loaded cache configuration from
"file:/C:/home/oracle/workspace/JCache/appClientModule/coherence-jcache-cache-
config.xml"
2014-02-25 11:09:27.986/4.471 Oracle Coherence GE 12.1.3.0.0 <Info> (thread=main,
member=2): Created cache factory com.tangosol.net.
ExtensibleConfigurableCacheFactory
2014-02-25 11:09:28.070/4.555 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=DistributedCache, member=2): Service DistributedCache joined the cluster
with senior service member 1
Value in cache is Gene Smith
```

## 12.4.5 Verify the Cache

The cache server in this example is configured, by default, to store the cache's data. The data is available to all members of the cluster and persists even after members leave the cluster. For example, the application exits after it loads and displays a key in the cache. However, the cache and key are still available for all cluster members.

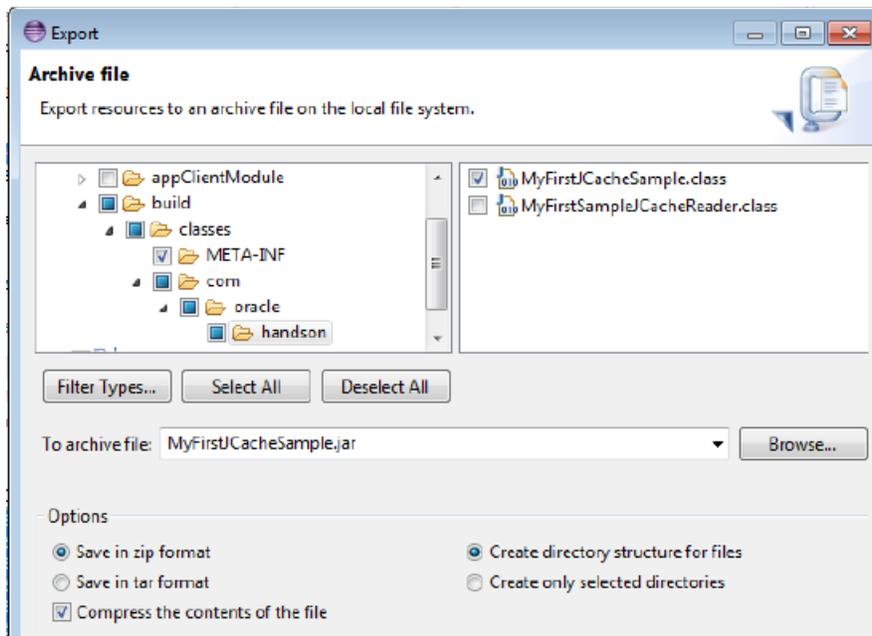
To complete this step, you first save the application to an archive file, then place it on the classpath. You can then use the cache factory command-line tool to connect to the myCache cache and list all items in the cache.

- [Create a JAR file for the Application](#)
- [Verify the Cache Contents Using a Cache Factory Instance](#)

### 12.4.5.1 Create a JAR file for the Application

To create a JAR file for the MyFirstJCacheSample application:

1. In the Eclipse project Explorer, right-click the **build** folder under **JCache** project and select **Export**.
2. In the **Select** screen of the **Export** wizard, open the **General** node and select **Archive File**. Click **Next**.
3. In the **Archive** file screen of the **Export** Wizard, open the **build** node and navigate down to the **handson** node. Select the **handson** node in the left pane and select the **MyFirstJCacheSample.class** file in the right pane. Enter **MyFirstJCacheSample.jar** in the **To archive file** field, as illustrated in [Figure 12-5](#).

**Figure 12–5 Creating an Archive File**

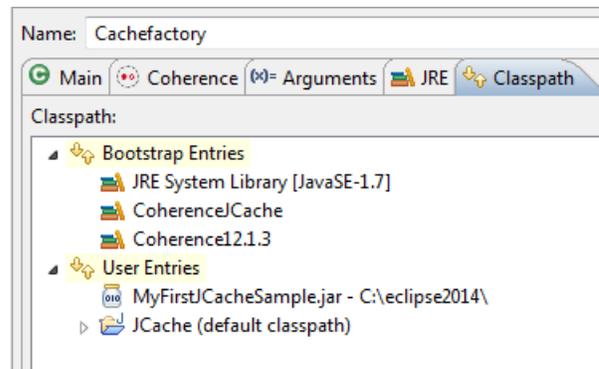
4. Click **Finish** to create the JAR file.

#### 12.4.5.2 Verify the Cache Contents Using a Cache Factory Instance

Start a cache factory instance to verify the contents of the cache.

To verify the cache:

1. Create a run configuration for the CacheFactory command line tool. Right click the **JCache** project in the Eclipse IDE. Select **Run As** then **Run Configurations**. In the **Run Configurations** dialog box, select **Oracle Coherence** then the **New launch configuration** icon. Enter CacheFactory as the name for the cache server configuration.
2. Under **Project**, click **Browse** and select the name of the JCache project from the **Project Selection** dialog box.
3. Under **Main class**, select the **Include system libraries when searching for a main class** checkbox. Click the **Search** button and enter CacheFactory in the **Select Main Type** dialog box. Select **com.tangosol.net.CacheFactory** and click **OK**. Click **Apply**.
4. In the Coherence tab navigate to the coherence-jcache-cache-config.xml file in the **Cache configuration descriptor** field. Select **Disabled (cache client)** in the **Local storage** field. Enter a unique value such as 3155 in the **Cluster port** field.
5. Open the **Classpath** tab of the dialog box. Click **Advanced**, then **Add External Jars** to add the MyFirstJCacheSample.jar file to the **User Entries** section of the classpath. Use the **Up** and **Down** buttons to position MyFirstJCacheSample.jar file before the JCache project folder. The **Classpath** tab should look similar to [Figure 12–6](#).

**Figure 12–6 Classpath for the Cache Factory Instance**

6. Open the **Common** tab of the dialog box. Click the **Shared file** radio button and click **Browse** to navigate to the JCache project. Click **Apply**.
7. Click **Run** to start the cache factory instance.

The cache factory instance starts and becomes a member of the cluster and returns a command prompt for the command-line tool. Use the `cache` command at the command-line tool command prompt to get the `myCache` cache, for example:

```
cache myCache
```

At the command-line tool command prompt, use the `list` command to retrieve the contents of the cache:

```
list
```

The command returns and displays:

```
Name = Gene Smith
```

[Example 12–5](#) displays the output of the cache factory instance.

#### **Example 12–5 Output of the CacheFactory Instance**

```
...
2014-02-24 15:58:26.240/0.610 Oracle Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=n/a): Loaded operational overrides from
"file:/C:/home/oracle/workspace/JCache/build/classes/tangosol-coherence-override.
xml"
...
2014-02-24 15:58:29.303/3.673 Oracle Coherence GE 12.1.3.0.0 <D5>
(thread=Invocation:Management, member=7): Service Management joined the cluster
with senior service member 1
...
Map (?): cache myCache
2014-02-24 16:03:13.374/287.744 Oracle Coherence GE 12.1.3.0.0 <Info>
(thread=main, member=7): Loaded cache configuration from
"file:/C:/home/oracle/workspace/JCache/appClientModule/coherence-jcache-cache-
config.xml"
...
Map (myCache): list
Name = Gene Smith

Map (myCache):
```



---

# Caching Sessions with Managed Coherence Servers

This exercise describes how to cache session information for Web application instances that are deployed across WebLogic Server instances. It highlights the use of managed servers, which allows you to use Coherence data caches and seamlessly incorporate Coherence\*Web for session management.

If you do not already have the current WebLogic Server release, you can get it at the following URL:

<http://www.oracle.com/technology/software/products/middleware/index.html>

This chapter has the following sections:

- [Introduction](#)
- [Caching Session Information for Web Application Instances](#)
- [Working with Custom Session Cache Configuration Files](#)

## Introduction

WebLogic Server includes features that enable deployed applications to use Coherence data caches and seamlessly incorporate Coherence\*Web for session management and TopLink Grid as an object-to-relational persistence framework. Collectively, these features are referred to as *Managed Coherence Servers*.

### About Managed Coherence Servers

Managed Coherence Servers are employed by applications running on WebLogic Server and provides replicated and distributed caching services that make an application's data available to all servers in a Coherence data cluster. Applications can obtain direct access to data caches either through resource injection or component-based JNDI lookup. You can display, monitor, create, and configure Coherence clusters using the WebLogic Server Administration Console and WLST.

Using Managed Coherence Servers with WebLogic Server instances enable you to create a data tier dedicated to caching application data and storing replicated session state. This is separate from the application tier, where the WebLogic Server instances are dedicated to running the application.

Using Coherence\*Web with Managed Coherence Servers enables you to provide Coherence-based HTTP session state persistence to applications. Coherence\*Web enables HTTP session sharing and management across different Web applications, domains, and heterogeneous application servers. Session data can be stored in data

caches outside of the application server, thus freeing application server heap space and enabling server restarts without losing session data.

Coherence and Coherence\*Web are included in the default installation of WebLogic Server. The Coherence and Coherence\*Web libraries (`coherence.jar` and `coherence-web.jar`) are included in the system classpath of WebLogic Server. These libraries will load application classes with the appropriate classloader in WebLogic Server. This means that you do not have to include the `coherence.jar` or `coherence-web.jar` files in the web application's classpath.

For more information on the integration of Oracle WebLogic Server, Coherence, and Coherence\*Web, see *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence\*Web*. For more information on Managed Coherence Servers, see *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*.

### About Coherence Clusters

Coherence clusters are different than WebLogic Server clusters. They use different clustering protocols and are configured separately. Multiple WebLogic Server clusters can be associated with a Coherence cluster and a WebLogic Server domain can contain only a single Coherence cluster. Managed servers that are configured as Coherence cluster members are referred to as managed Coherence servers.

---

---

**Note:** Using multiple Coherence clusters in a single WebLogic Server domain is not recommended.

---

---

Managed Coherence servers can be explicitly associated with a Coherence cluster or they can be associated with a WebLogic Server cluster that is associated with a Coherence cluster. Managed Coherence servers are typically setup in tiers based on their type: a data tier for storing data, an application tier for hosting applications, and a proxy tier for allowing access to external clients.

For more information on Coherence clusters in a WebLogic server environment, see "Configuring and Managing Coherence Clusters" in *Administering Clusters for Oracle WebLogic Server*.

## Caching Session Information for Web Application Instances

The following example demonstrates how Managed Coherence Servers and Coherence\*Web caches session information for Web application instances that are deployed across WebLogic Server instances. To do this, you will create a Web application and deploy it to two WebLogic Server instances that belong to a Coherence cluster. The application is a simple counter that stores the current count as a session attribute. Coherence\*Web automatically serializes and replicates the attribute across both server instances. A browser is used to access each application instance to demonstrate that the same session attribute is used among the instances.

1. [Configure and Start WebLogic Server](#)
2. [Create the WebLogic Servers](#)
3. [Create a Coherence Cluster](#)
4. [Enable a Server for Coherence\\*Web Local Storage](#)
5. [Create the Counter Web Application](#)
6. [Start the WebLogic Servers](#)

7. [Deploy the Application](#)
8. [Verify the Example](#)

## Configure and Start WebLogic Server

The following instructions assume that you have installed WebLogic Server in the default location: C:\Oracle\Middleware\Oracle\_Home.

To configure and start WebLogic Server:

1. Run the Oracle WebLogic Configuration Wizard to create a new WebLogic Server domain.
2. In the **Create Domain** page, select the **Create a new domain** option. In the **Domain location** field, enter C:\Oracle\Middleware\Oracle\_Home\user\_projects\domains\test\_domain as the domain name and location.
3. In the **Templates** page, ensure that the default **Basic WebLogic Server Domain** is selected.
4. In the **Administrator Account** page, enter a password for the domain.
5. In the **Domain Mode and JDK** and the **Advanced Configuration** pages, accept the defaults.
6. In the **Configuration Summary** page click **Create**.
7. In the final **Configuration Success** page, select the **Start Admin Server** check box and click **Done**. The Configuration Wizard automatically starts the Administration Server.
8. Start the WebLogic Server Administration Console.

From the browser, log in to the Oracle WebLogic Server Administration Console using the following URL: `http://hostname:7001/console`. The Console starts and the domain home page displays.

## Create a Machine

To create a Machine on which to host WebLogic Server instances:

From the **Domain Structure** window, select **Environment** and then **Machines**. Click **New**. The **Create a New Machine** page displays. Enter a name for the Machine (in this case, **Test**) and click **Next**. Click **Finish** on the following page. [Figure 13-1](#) illustrates the **Create a New Machine** page.

**Figure 13–1** *Creating a New Machine*

**Create a New Machine**

Back Next Finish Cancel

**Machine Identity**

The following properties will be used to identify your new Machine.  
\* Indicates required fields

What would you like to name your new Machine?

\* **Name:**

Specify the type of machine operating system.

\* **Machine OS:**

Back Next Finish Cancel

The **Summary of Machines** page should look similar to [Figure 13–2](#).

**Figure 13–2 Summary of Machines**

**Messages**

- ✔ All changes have been activated. No restarts are necessary.
- ✔ Machine created successfully

**Summary of Machines**

A machine is the logical representation of the computer that hosts one or more WebLogic Server instances (servers). WebLogic Server uses configured machine names to determine the optimum server in a cluster to which certain tasks, such as HTTP session replication, are delegated. The Administration Server uses the machine definition in conjunction with Node Manager to start remote servers.

This page displays key information about each machine that has been configured in the current WebLogic Server domain.

[Customize this table](#)

**Machines**

New Clone Delete Showing 1 to 1 of 1 Previous | Next

<input type="checkbox"/>	Name ↕	Type
<input type="checkbox"/>	Test	Machine

New Clone Delete Showing 1 to 1 of 1 Previous | Next

## Create the WebLogic Servers

Create two WebLogic server instances associated with the machine. One will be Coherence\*Web storage-enabled, the other will be Coherence\*Web storage-disabled. The application will be deployed to these servers in a later step.

To create server instances:

1. Click the name of the Machine in the **Summary of Machines** page to open the **Settings for machine** page. Click the **Servers** tab and then click **Add** to create a server.
2. Select **Create a new server and associate it with this machine** in the **Add a Server to Machine** page, and click **Next**.
3. Provide details about the server in the **Create a New Server** page.

Enter `ServerA` as the **Server Name** and `8081` as the **Server Listen Port**. Ensure that **No, this is a stand-alone server** is selected, as illustrated in [Figure 13–3](#). Click **Finish**.

**Figure 13–3 Adding a Server to a Machine**

**Add a Server to Machine**

Back Next Finish Cancel

**Server Properties**

The following properties will be used to identify your new server.

\* Indicates required fields

What would you like to name your new server?

\* **Server Name:**

Where will this server listen for incoming connections?

**Server Listen Address:**

\* **Server Listen Port:**

Back Next Finish Cancel

4. When you are returned to the **Settings for machine** page, click **Add**, and repeat the previous step to create a second server.

Enter `ServerB` as the **Server Name** and 8082 as the **Server Listen Port**. Ensure that **No, this is a stand-alone server** is selected. Click **Finish**.

5. Click **Servers** in the **Domain Structure**. The **Summary of Servers** page displays and is similar to [Figure 13–4](#).

Figure 13–4 Summary of Servers Page

**Summary of Servers**

**Configuration** Control

A server is an instance of WebLogic Server that runs in its own Java Virtual Machine (JVM) and has its own configuration.

This page summarizes each server that has been configured in the current WebLogic Server domain.

↻

▶ **Customize this table**

**Servers (Filtered - More Columns Exist)**

New Clone Delete Showing 1 to 3 of 3 Previous | Next

<input type="checkbox"/>	Name ↕	Type	Cluster	Machine	State	Health	Listen Port
<input type="checkbox"/>	AdminServer (admin)	Configured			RUNNING	✔ OK	7001
<input type="checkbox"/>	ServerA	Configured		Test	SHUTDOWN		8081
<input type="checkbox"/>	ServerB	Configured		Test	SHUTDOWN		8082

New Clone Delete Showing 1 to 3 of 3 Previous | Next

## Create a Coherence Cluster

A Coherence cluster is a group of Coherence members that share a group address which allows them to communicate. Coherence clusters consist of members formed by applications, modules, or application servers (WebLogic Server instances or cache servers). In this step you create a Coherence cluster and assign the two WebLogic Server instances to it.

To create a Coherence cluster:

1. Click **Environment** in the domain **Structure Window** and then click **Coherence Clusters**. In the **Summary of Coherence Clusters** page, click **New**. In the **Coherence Cluster Properties** page of the **Create Coherence Cluster Configuration** wizard, enter `CoherenceCluster` in the **Name** field, then click **Next**.

Figure 13–5 illustrates the **Create Coherence Cluster Configuration** page.

**Figure 13–5 Creating a Coherence Cluster**

**Create a Coherence Cluster Configuration**

Back Next Finish Cancel

**Coherence Cluster Properties**

The following properties will be used to identify your new Coherence cluster configuration.

\* Indicates required fields

What would you like to name your new Coherence cluster configuration?

\* **Name:**

Coherence clusters can be configured externally in a custom configuration file or configured within WebLogic Server. Select check-box if you would like to configure this Coherence cluster using a custom cluster configuration file. The values in this file will override any values set at the Coherence cluster level.

**Use a Custom Cluster Configuration File**

Back Next Finish Cancel

2. Enter a value such as 8085, in the **Unicast Listen Port** field. Do not change any of the other values and click **Next**.

**Figure 13–6 Specifying a Unicast Listen Port for a Coherence Cluster**

**Create a Coherence Cluster Configuration**

Back Next Finish Cancel

**Coherence Cluster Addressing**

This page indicates how this Coherence cluster will be located.

How should this Coherence cluster be addressed?

**Clustering Mode:**  ▼

**Unicast Listen Port:**

**Multicast Listen Address:**

**Multicast Listen Port:**

Back Next Finish Cancel

3. In the **Coherence Cluster Members** page of the **Create Coherence Cluster Configuration** wizard, select **ServerA** and **ServerB** as the targets. Do *not* select **AdminServer**. Click **Finish**.

**Figure 13–7 Choosing Coherence Cluster Members**

**Create a Coherence Cluster Configuration**

Back Next Finish Cancel

**Coherence Cluster Members**

This page specifies WebLogic Server or clusters that this Coherence Cluster is associated with.

Servers	
<input type="checkbox"/>	AdminServer
<input checked="" type="checkbox"/>	ServerA
<input checked="" type="checkbox"/>	ServerB

Back Next Finish Cancel

The **Summary of Coherence Clusters** page looks similar to [Figure 13–8](#).

**Figure 13–8 Summary of Coherence Clusters**

**Summary of Coherence Clusters**

Coherence provides replicated and distributed data management and caching services that you can use to reliably make an application's objects and data available to all servers in a Coherence cluster. To do this, WebLogic Server retains configuration information used to locate and communicate with a Coherence cluster.

This page displays the Coherence cluster configurations that have been created in this domain.

[Customize this table](#)

**Coherence Clusters (Filtered - More Columns Exist)**

New Delete Showing 1 to 1 of 1 Previous | Next

<input type="checkbox"/>	Name	Logging Enabled	Members
<input type="checkbox"/>	CoherenceCluster	true	ServerA, ServerB

New Delete Showing 1 to 1 of 1 Previous | Next

## Enable a Server for Coherence\*Web Local Storage

One of the WebLogic servers must be enabled for Coherence\*Web local storage. In this step, Coherence\*Web local storage will be enabled for *ServerB*.

1. Click **Servers** in the **Domain Structure** tree to open the Summary of Servers.

2. Click **ServerB**.
3. In **Settings for ServerB**, click the **Configuration** tab, then the **Coherence** tab.
4. Select the **Coherence\*Web Local Storage Enabled** checkbox, as illustrated in [Figure 13–9](#).
5. Click **Save**.

**Figure 13–9 Enabling a Server for Coherence\*Web Local Storage**

**Settings for ServerB**

**Configuration** Protocols Logging Debug Monitoring Control Deployments Services Security Notes

General Cluster Services Keystores SSL Federation Services Deployment Migration Tuning Overload

Health Monitoring Server Start Web Services **Coherence**

Use this page to configure this server instance as a Coherence cluster member by selecting the Coherence cluster that it belongs to. After this configuration is saved, you can configure Coherence cluster member properties as required.

**Coherence Cluster:**  The system-level Coherence cluster resource associated with this server. [More Info...](#)

**Unicast Listen Address:**  The IP address for the Coherence unicast listener. [More Info...](#)

**Unicast Listen Port:**  The port for the Coherence unicast listener. A value of 0 indicates that the unicast listener port value from the Coherence Cluster configuration will be used. Any other non-zero value set here will override the value set at the Coherence Cluster configuration. [More Info...](#)

**Unicast Port Auto Adjust** Specifies whether the unicast port will be automatically incremented if the port cannot be bound because it is already in use. [More Info...](#)

**Local Storage Enabled** Specifies whether or not this member will contribute storage to the Coherence cluster i.e. maintain partitions. This attribute is used only when the WebLogic Server is not part of a WLS Cluster. [More Info...](#)

**Coherence Web Local Storage Enabled** Specifies whether Local Storage is enabled for the Coherence Web cluster member. [More Info...](#)

## Create the Counter Web Application

The Counter web application is a simple counter implemented as a JSP. The counter is stored as an HTTP session attribute and increments each time the page is accessed.

To create the Counter web application:

1. Create a standard web application folder as follows:

```
/
/WEB-INF
```

2. Copy the following code to a text file and save it as a file named `web.xml` in the `/WEB-INF` folder.

```
<?xml version = '1.0' encoding = 'windows-1252'?>

<web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.
com/xml/ns/javaee/web-app_2_5.xsd"
  version="2.5">
  <description>Empty web.xml file for Web Application</description>
</web-app>
```

3. Create a `weblogic.xml` file in the `/WEB-INF` folder.
  - Reference `coherence-web` as the persistent store type for the session descriptor.
  - Reference the Coherence cluster in a `coherence-cluster-ref` stanza.

[Example 13-1](#) illustrates a sample `weblogic.xml` file.

#### **Example 13-1 Sample `weblogic.xml` File**

```
<weblogic-web-app>
  <session-descriptor>
    <persistent-store-type>coherence-web</persistent-store-type>
  </session-descriptor>
  <coherence-cluster-ref>
    <coherence-cluster-name>CoherenceCluster</coherence-cluster-name>
  </coherence-cluster-ref>
</weblogic-web-app>
```

4. Create a `lib` subfolder in the `WEB-INF` folder. Leave it empty.
5. Copy the following code for the counter JSP to a text file and save the file as `counter.jsp` in the root of the Web application folder.

```
<html>
  <body>
    <h3>
      Counter :
    <%
      Integer counter = new Integer(1);
      HttpSession httpsession = request.getSession(true);
      if (httpsession.isNew()) {
        httpsession.setAttribute("count", counter);
        out.println(counter);
      } else {
        int count = ((Integer) httpsession.getAttribute("count")).
intValue();
        httpsession.setAttribute("count", new Integer(++count));
        out.println(count);
      }
    %>
    </h3>
  </body>
</html>
```

6. Create a `MANIFEST.MF` file in the `META-INF` folder. [Example 13-2](#) illustrates a sample `MANIFEST.MF` file.

#### **Example 13-2 Sample `MANIFEST.MF` File**

```
Manifest-Version: 1.0
```

Created-By: 1.7.0\_03 (Oracle Corporation)

7. The structure of the web application folder should appear as follows:

```
/
/counter.jsp
/META-INF/MANIFEST.MF
/WEB-INF/lib/
/WEB-INF/web.xml
/WEB-INF/weblogic.xml
```

8. ZIP or JAR the Web application folder and save the file as `counter.war`.

## Start the WebLogic Servers

Start the managed WebLogic Servers `ServerA` and `ServerB`. Because of the tight integration between WebLogic and Coherence, the cache server will start with the managed servers. The managed servers can be started from the command line or from the WebLogic Server Administration Console.

- [To Start the WebLogic Servers from the Command Line](#)
- [To Start the WebLogic Servers from the WebLogic Server Administration Console](#)

### To Start the WebLogic Servers from the Command Line

The following instructions are for starting the servers from the command line.

1. Open a command prompt from the domain's `bin` folder. This example assumes that the `bin` folder is located in `c:\Oracle\user_projects\domains\base_domain\bin`.
2. Enter the following command to start `ServerA`:

```
startManagedWebLogic.cmd ServerA
```

Enter the server's user name and password when prompted.

3. Open a second command prompt from the domain's `bin` folder.
4. Enter the following command to start `ServerB`:

```
startManagedWebLogic.cmd ServerB
```

Enter the server's user name and password when prompted.

When the servers are running, you should see the response similar to [Figure 13-10](#) in the Summary of Servers page in the WebLogic Server Administration Console.

**Figure 13–10 Starting the Managed WebLogic Servers**

**Summary of Servers**

**Configuration** Control

A server is an instance of WebLogic Server that runs in its own Java Virtual Machine (JVM) and has its own configuration.

This page summarizes each server that has been configured in the current WebLogic Server domain.

**Customize this table**

**Servers (Filtered - More Columns Exist)**

New Clone Delete Showing 1 to 3 of 3 Previous | Next

<input type="checkbox"/>	Name	Type	Cluster	Machine	State	Health	Listen Port
<input type="checkbox"/>	AdminServer(admin)	Configured			RUNNING	✓ OK	7001
<input type="checkbox"/>	ServerA	Configured		Test	RUNNING	✓ OK	8081
<input type="checkbox"/>	ServerB	Configured		Test	RUNNING	✓ OK	8082

New Clone Delete Showing 1 to 3 of 3 Previous | Next

### To Start the WebLogic Servers from the WebLogic Server Administration Console

To start WebLogic Servers from WebLogic Server Administration Console, see "Start Managed Servers from the Administration Console" in the Oracle WebLogic Server Administration Console Online Help.

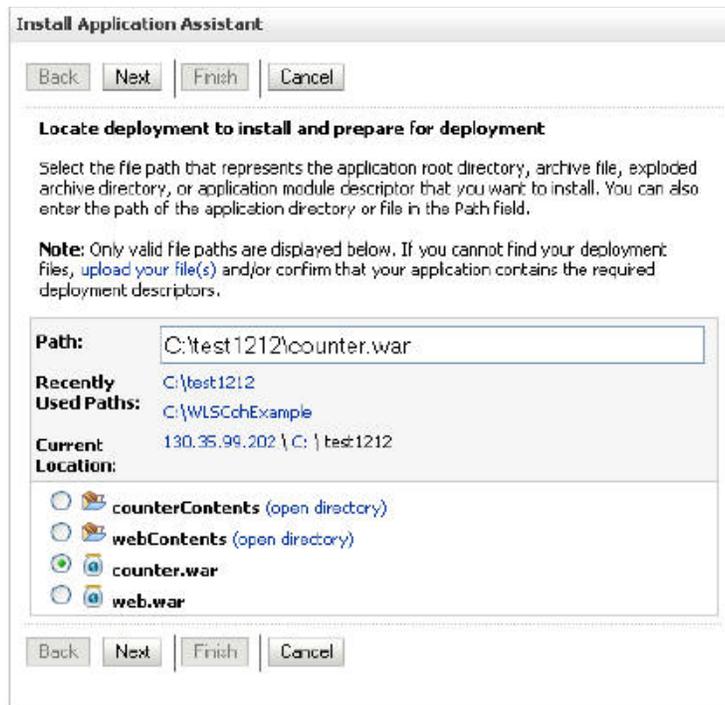
When the servers are running, you should see the response similar to [Figure 13–10](#) in the Summary of Servers page in the WebLogic Server Administration Console.

## Deploy the Application

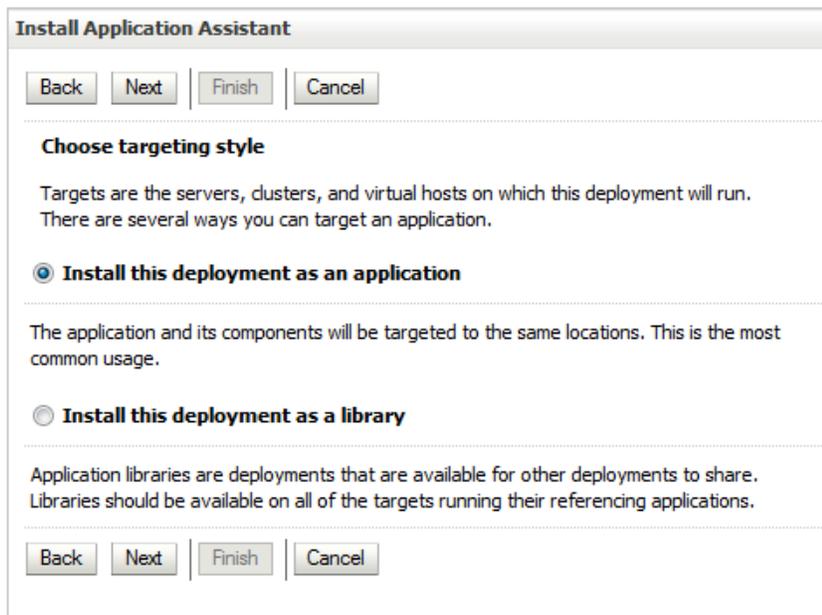
Deploy the `counter.war` application to the running managed servers.

To deploy the `counter.war` application:

1. Open the **Summary of Deployments** page by clicking **Deployments** in the **Domain Structure** menu in the Oracle WebLogic Server Administration Console.
2. Click **Install**. The **Install Application Assistant** wizard opens.
3. Navigate to the location of the `counter.war` file, as illustrated in [Figure 13–11](#). Click **Next**.

**Figure 13–11 Locating the Deployable Application**

4. Select **Install this deployment as an application** as illustrated in [Figure 13–12](#). Click **Next**.

**Figure 13–12 Installing the Deployment as an Application**

5. Target the deployment to `ServerA` and `ServerB`, as illustrated in [Figure 13–13](#).

**Figure 13–13 Targeting the Deployment**

The screenshot shows a dialog box titled "Install Application Assistant". At the top, there are four buttons: "Back", "Next", "Finish", and "Cancel". Below the buttons, the text reads "Select deployment targets" followed by "Select the servers and/or clusters to which you want to deploy this application. (You can reconfigure deployment targets later).". Underneath, it says "Available targets for wls\_test :". A table with a blue header "Servers" contains three rows: "AdminServer" with an unchecked checkbox, "ServerA" with a checked checkbox, and "ServerB" with a checked checkbox. At the bottom of the dialog, there are four buttons: "Back", "Next", "Finish", and "Cancel".

Servers
<input type="checkbox"/> AdminServer
<input checked="" type="checkbox"/> ServerA
<input checked="" type="checkbox"/> ServerB

6. In the **Optional Settings** page, accept the defaults, as illustrated in [Figure 13–14](#). Click **Finish**.

**Figure 13–14** *Optional Settings Page of the Installation Assistant*

**Install Application Assistant**

Back Next Finish Cancel

**Optional Settings**

You can modify these settings or accept the defaults

\* Indicates required fields

**General**

What do you want to name this deployment?

\* Name:

**Security**

What security model do you want to use with this application?

**DD Only:** Use only roles and policies that are defined in the deployment descriptors.

**Custom Roles:** Use roles that are defined in the Administration Console; use policies that are defined in the deployment descriptor.

**Custom Roles and Policies:** Use only roles and policies that are defined in the Administration Console.

**Advanced:** Use a custom model that you have configured on the realm's configuration page.

**Source Accessibility**

How should the source files be made accessible?

**Use the defaults defined by the deployment's targets**

Recommended selection.

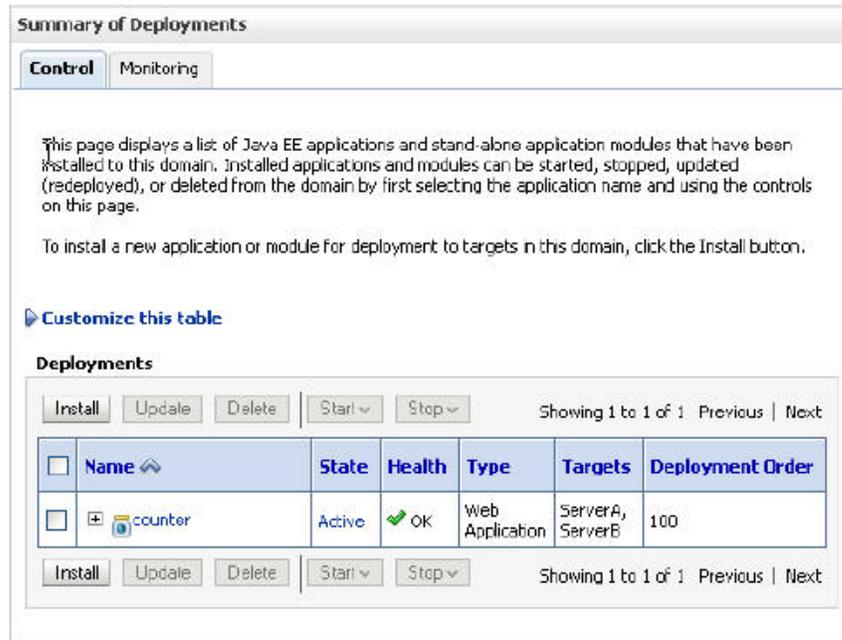
**Copy this application onto every target for me**

During deployment, the files will be copied automatically to the Managed Servers to which the application is targeted.

**I will make the deployment accessible from the following location**

Location:

The `counter.war` file appears in the table in the Summary of Deployments page. The table indicates that the application is active and running (since the servers are running) on `ServerA` and `ServerB`, as illustrated in [Figure 13–15](#).

**Figure 13–15 Deployments Window Showing the Deployed Application**

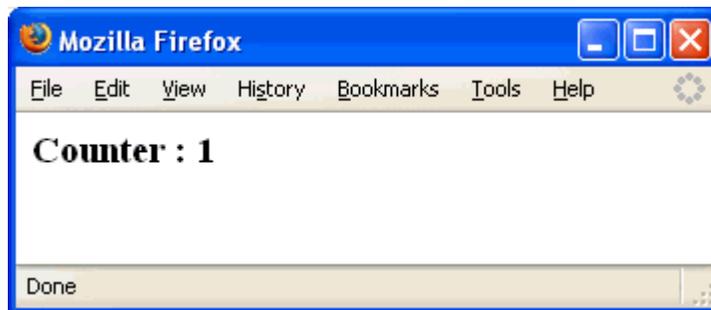
## Verify the Example

To verify the example:

1. Open a browser and access the ServerA counter instance using the following URL:

```
http://host:8081/counter/counter.jsp
```

The counter page displays and the counter is set to 1 as illustrated in [Figure 13–16](#).

**Figure 13–16 Counter Page with Counter Set to 1**

2. In a new browser (or new browser tab), access the ServerB counter instance using the following URL:

```
http://host:8082/counter/counter.jsp
```

The counter page displays and the counter increments to 2 based on the session data as illustrated in [Figure 13–17](#).

**Figure 13–17 Counter Page with Counter Set to 2**

3. If you refresh the page, the counter increments to 3. Return to the original browser (or browser tab), refresh the instance, and the counter displays 4.

## Working with Custom Session Cache Configuration Files

If you want to use a custom session cache configuration file, you must package it with your application. It must also be stored in a Grid Archive (GAR) file and deployed to the WebLogic Server cluster that is to act as the storage-enabled Coherence cluster members. A detailed discussion of working with custom session cache configuration files is beyond the scope of this document. For more information on creating and deploying a custom session cache configuration file, see "Using a Custom Session Cache Configuration File" in *Oracle Fusion Middleware Administering HTTP Session Management with Oracle Coherence\*Web*.

---

---

## Coherence Examples in the examples.zip File

The Coherence distribution provides a collection of example code in the `examples.zip` file. These examples demonstrate how to use basic Coherence functionality, security, and events features in all supported languages (Java, .NET, and C++). The examples are organized as collections of code that show how to use one or more features. They also provide a single common way (per language) to build and run all examples.

This appendix has the following sections:

- [Examples Provided in the examples.zip File](#)
- [Obtaining the examples.zip File](#)
- [How to Build the Examples](#)
- [How to Run the Examples](#)
- [Coherence Basic Features Examples](#)
- [Coherence Security Examples](#)
- [Coherence Live Events Examples](#)

There are a number of differences between the examples in the `examples.zip` file described in this appendix and the examples that are presented in the main body of the tutorial:

- The examples in the `examples.zip` must be built and run from the command line. The tutorial uses an IDE to compile and run the code.
- The examples in the `examples.zip` file demonstrate how to use basic Coherence functionality and security features in all supported languages (Java, .NET, and C++). The tutorial covers only Java implementations.
- The Java examples in the `examples.zip` file are only a subset of the Java examples presented in the tutorial.
- The Java code files in the `examples.zip` file are similar, *but not identical to*, the files used in the tutorial. In many instances, the code in the tutorial has been simplified for demonstration purposes.

### A.1 Examples Provided in the examples.zip File

The Coherence Basic Features Examples include the following:

**Table A–1 Coherence Basic Features Examples**

Example Name	Description
Basic Data Access	"Getting", "putting" and "removing" data from the Coherence Data Grid. See <a href="#">Section A.5.3, "Basic Data Access Example."</a>
Data Loading	Loading example data into the Coherence Data Grid. See <a href="#">Section A.5.4, "Loader Example."</a>
Parallel Querying	Querying the Coherence Data Grid including the use of indexes. See <a href="#">Section A.5.5, "Query Example."</a>
Observable	Listening for changes to data in the Coherence Data Grid. See <a href="#">Section A.5.6, "Observer Example."</a>
Processing	Co-locating data processing with the data itself in the Coherence Data Grid. See <a href="#">Section A.5.7, "Processor Example."</a>
Query Language	How to use the Coherence Query Language. See <a href="#">Section A.5.5, "Query Example."</a>

The Coherence Security Examples include the following:

**Table A–2 Coherence Security Examples**

Example Name	Description
Password Example	Requiring a password to access Coherence. See <a href="#">Section A.6.2, "Password Example."</a>
Access Control Example	Simplified role based access control. See <a href="#">Section A.6.3, "Access Control Example."</a>
Password Identity Transformer	Creates a custom security token that contains the required password and then adds a list of Principal names. See <a href="#">Section A.6.4, "Password Identity Transformer."</a>
Password Identity Asserter	Asserts that the security token contains the required password and then constructs a Subject based on a list of Principal names. See <a href="#">Section A.6.5, "Password Identity Asserter."</a>
Entitled Cache Service	Wraps a cache service for access control. See <a href="#">Section A.6.6, "Entitled Cache Service."</a>
Entitled Invocation Service	Wraps an invocation service for access control. See <a href="#">Section A.6.7, "Entitled Invocation Service."</a>
Entitled Named Cache	Wraps a named cache for access control. See <a href="#">Section A.6.8, "Entitled Named Cache."</a>

The Coherence Live Events Examples are available for the Java platform only. They include the following:

**Table A–3 Coherence Live Events Examples**

Example Name	Description
EventsExamples	Illustrates various features within Live Events, such as providing mean elapsed times split by event type, the different semantics in throwing exceptions in pre-events compared to post-events, and logging of partition movement when enabled. See <a href="#">Section A.7.2, "EventsExamples."</a>
TimedTraceInterceptor	Provides timings between pre- and post-commit events for different types of events. See <a href="#">Section A.7.3, "TimedTraceInterceptor."</a>

**Table A–3 (Cont.) Coherence Live Events Examples**

Example Name	Description
CantankerousInterceptor	Responds with runtime exceptions at either pre- or post-commit time, based on the type of key being inserted. See <a href="#">Section A.7.4</a> , "CantankerousInterceptor."
RedistributionInterceptor	Logs partition events when enabled. See <a href="#">Section A.7.5</a> , "RedistributionInterceptor."
RedistributionInvocable	Defines three actionable states that will be executed on various members of the cluster. The states are enable logging performed by the RedistributionInterceptor, disable logging, or terminate the JVM that the invocable (RedistributionInvocable) is executed on. See <a href="#">Section A.7.6</a> , "RedistributionInvocable."
LazyProcessor	Creates a superficial delay between the processing of events. See <a href="#">Section A.7.7</a> , "LazyProcessor."

## A.2 Obtaining the examples.zip File

You can obtain the `examples.zip` file by performing a full Coherence installation with the `coherence_version.jar` or `wls_version.jar` installer file. The Coherence examples appear as an installation option in the Oracle Universal Installer.

If you have already installed Coherence but without the examples, you can obtain the `examples.zip` file by running the `coherence_quick_supp_version.jar` supplemental installer file. The supplemental installer contains only API documentation and examples.

Note that the `coherence_quick_version.jar` quick installer file does not install the examples or API documentation.

Unzip the contents of the `examples.zip` file into an `examples` directory.

## A.3 How to Build the Examples

---

**Note:** You must build and run the Java example even for .NET and C++. This is because the cache server runs in Java.

---

This section contains the following information:

- [How to Build the Java Examples](#)
- [How to Build the .NET Examples](#)
- [How to Build the C++ Examples](#)

### A.3.1 How to Build the Java Examples

This section contains the following information:

- [Prerequisites for Java](#)
- [Directory Structure for Java](#)
- [Build Instructions for Java](#)

### A.3.1.1 Prerequisites for Java

To build the example, you must have Coherence version 3.7 or later and a Java development kit (JDK) 1.6 or later. Ensure that the following environment variables are set.

Environment Variable	Description
\$COHERENCE_HOME	Make sure that the COHERENCE_HOME environment variable points to the location of the unpacked Coherence 3.7 directory.
\$JAVA_HOME	Make sure that the JAVA_HOME environment variable points to the location of a 1.6 or greater JDK before building the example. A Java runtime 1.6 or greater is needed to run the example

### A.3.1.2 Directory Structure for Java

The directory structure described below is relative to the `examples` directory.

**Table A-4** Directory Structure for Java

Directory Name	Description
java/bin	Scripts for building and executing the example. There are two sets of scripts. Scripts with no file extension are bash scripts. Scripts with a <code>.cmd</code> file extension are Windows command scripts. The following description refers to the script names without specifying the file extension. <ul style="list-style-type: none"> <li>■ <code>build</code>—builds an example</li> </ul>
java/src	All example source. The examples are in the <code>com.tangosol.examples.&lt;example name&gt;</code> package. The classes for objects stored in the cache are in the <code>com.tangosol.examples.pof</code> package.
java/classes	The class files output from a build. This directory will not exist until the build script is executed.
java/resource/config	The common Coherence configuration files required by the examples.
java/resource/<example name>	If an example has configuration that is required instead of the common configuration, it will have its own directory. The security example uses configuration files from <code>java/resource/security</code> .
\$COHERENCE_HOME/lib	Coherence libraries used for compiling and running the example.

### A.3.1.3 Build Instructions for Java

Execute the build script with the name of the example collection, for example: `bin/build contacts`, `bin/build security`, or `bin/build events`.

The script will build the POF package files and then the files for the particular example. On Windows, change directories to the `/bin` directory then run the scripts.

## A.3.2 How to Build the .NET Examples

This section contains the following information:

- [Prerequisites for .NET](#)
- [Directory Structure for .NET](#)
- [Build Instructions for .NET](#)

### A.3.2.1 Prerequisites for .NET

To build the example, you must have Coherence version 3.7 or later for .NET and Visual Studio 2008 or later or Visual Studio 2008 Express or later.

To run the example, you will need the Java version of Coherence 3.7 or later and a Java development kit (JDK) 1.6 or greater. The Java version is required because the Coherence\*Extend proxy and cache servers require Java. Also, the examples depend on Java example classes that must be built before running the proxy and cache server. See the Java example `readme.txt` file for instructions on how to build and run. Note that the Java `run-proxy` script must be executed; the Java `run-cache-server` is optional because the proxy is storage enabled.

### A.3.2.2 Directory Structure for .NET

The directory structure described below is relative to the `examples` directory.

**Table A-5** Directory Structure for .NET

Directory Name	Description
<code>dotnet\src</code>	<p>All example source. The examples are in the <code>Tangosol.Examples.&lt;example name&gt;</code> namespace. The classes for objects stored in the cache are in the <code>Tangosol.Examples.Pof</code> namespace.</p> <p>The examples are in the Visual Studio 2008 examples solution. Each example has its own Visual Studio 2008 project in the <code>src</code> directory. For example, <code>src</code> contains projects for the <code>contacts</code> and <code>security</code> examples.</p> <p>The Coherence configuration files required by the example.</p>
<code>src\pof\config</code>	The common Coherence configuration files required by the examples.
<code>src&lt;example name&gt;\config</code>	If an example has configuration that is required instead of the common configuration, it will have its own directory. The security example uses configuration files from <code>security\config</code> .

### A.3.2.3 Build Instructions for .NET

Open the examples project from the `examples\dotnet\src\contacts.csproj` directory with Visual Studio

When installing Coherence 3.7 for the .NET Framework, the installer registers the `coherence.dll` library with the assembly registry. The included Visual Studio projects have a reference to `coherence.dll` in the default location. If another version of the library is desired, or it was not installed in the default location, the Coherence reference can be overridden when configuring the reference, be sure to set the `local` copy attribute to `true`. This setting will copy and register the correct `coherence.dll` in the `bin\debug` directory.

After the desired Coherence 3.7 for .NET is configured, in Visual Studio select **Build** then **Build Solution** from the menu, **Build Solution (F6)**, etc., to build the solution.

The build for the `contacts` example will copy `resource\contacts.csv` to the build output directory (`examples\dotnet\src\bin\Debug`).

## A.3.3 How to Build the C++ Examples

This section contains the following information:

- [Prerequisites for C++](#)

- [Directory Structure for C++](#)
- [Build Instructions for C++](#)

### A.3.3.1 Prerequisites for C++

To run the examples, you will need the Java version of Coherence 3.7 or later and a Java development kit (JDK) 1.6 or greater. The Java version is required because the Coherence\*Extend proxy and cache servers require Java. Also, the examples depend on Java example classes that must be built before running the proxy and cache server. See the Java examples readme.txt for instructions on how to build and run. Note that the Java run-proxy script must be executed; the Java run-cache-server is optional because the proxy is storage enabled.

Ensure that the following environment variables are set:

Environment Variable	Description
%COHERENCE_HOME%	Make sure that the COHERENCE_HOME environment variable points to the location of the unpacked Coherence 3.7 (or later) directory.
%JAVA_HOME%	Make sure that the JAVA_HOME environment variable points to the location of a 1.6 or greater JDK before building the examples. A Java runtime 1.6 or greater is needed to run the examples.
%COHERENCE_CPP_HOME%	Make sure that the COHERENCE_CPP_HOME environment variable points to the location of the unpacked C++ development environment. Compiler environments supported.

### A.3.3.2 Directory Structure for C++

The directory structure described below is relative to the examples directory.

**Table A-6 Directory Structure for C++**

Directory Name	Description
cpp\bin	Scripts for building and executing the examples. Scripts with no file extension are bash scripts. Scripts with a .cmd file extension are Windows command scripts. The following description refers to the script names without specifying any file extension.
cpp	All example source organized under the <example name> (such as contacts and security) and pof directories.
cpp\contacts	The contacts example source. The examples are in the coherence::examples namespace. The next level of the name after examples represents a related set of example classes. "Driver" in coherence::examples::LoaderExample is the Loader for the contacts example. In other words, the name of the example is the name after coherence::examples.
cpp\security	The security example source. The examples are in the coherence::examples namespace.
cpp\pof	The data model is represented in this directory plus any classes that are serialized. The rationale is to show how to utilize an already existing data model and expose it in Coherence. The model classes do not contain any Coherence-specific code to prove this point. However, there is a serializer that is associated with each model type. For example the Contact has a ContactSerializer class whose purpose is to register the model type with Coherence and serialization operations.  The generated output will be in the form of a dynamic library.

**Table A-6 (Cont.) Directory Structure for C++**

Directory Name	Description
cpp\config	The common Coherence configuration files required by the examples.
cpp\config\ <i>example name</i>	If an example has configuration that is required instead of the common configuration, it will have its own directory. The security example uses configuration files from config/security.
cpp\ <i>example name</i> \out	The object files output from a build. This directory will not exist until the build script is executed.
%COHERENCE_CPP_HOME%\include	Contains the Coherence header files.
%COHERENCE_CPP_HOME%\lib	Contains the Coherence library.

### A.3.3.3 Build Instructions for C++

This section contains the following information:

- [Build Instructions for C++ on Windows](#)
- [Build Instructions for C++ on Linux/Mac and Solaris](#)

#### Build Instructions for C++ on Windows

Open a development environment command prompt. This should have been installed with Visual Studio or the platform SDK. Go to the C++ examples directory and type `bin\build.cmd <example name>`. This will build both the pof (model) and the example executable. For example, `bin\build.cmd contacts` or `bin\build.cmd security`

The model will put the `pof.lib` and `pof.dll` file under `cpp\pof\out`. These are needed for building and running the `contacts` and `security` examples.

The executable `contacts.exe` will be generated in `cpp\contacts\out` directory. The executable `security.exe` will be generated in `cpp\security\out` directory.

To run the `contacts` example, type `bin\run.cmd contacts` after starting a proxy server, `java\bin\run-proxy`, and an additional cache server `java\bin\run-cache-server`.

As an alternative, in any command window you can `cd` to the C++ bin directory and run `vcvars32.bat` before trying to build the examples. With a default install of Visual Studio, the bin directory is `C:\Program Files\Microsoft Visual Studio 9.0\vc\bin`. Follow the previous instructions for running the build script.

#### Build Instructions for C++ on Linux/Mac and Solaris

Open a command shell. Go to the C++ examples directory and type `bin/build <example name>`. This will build both the pof (model) and the `contacts` examples executable.

The model dynamic library and lib file will be put in `cpp/pof/out`. These are needed for building and running the `contacts` and `security` examples.

The executable `contacts`, will be generated in `cpp/contacts/out` or `cpp/security/out`.

## A.4 How to Run the Examples

---



---

**Note:** The Coherence examples are distributed as source, so they must first be built. See [Section A.3, "How to Build the Examples."](#)

---



---

This section contains the following information:

- [How to Run the Java Examples](#)
- [How to Run the .NET Examples](#)
- [How to Run the C++ Examples](#)

### A.4.1 How to Run the Java Examples

This section contains the following information:

- [Prerequisites for Java](#)
- [Directory Structure for Java](#)
- [Instructions for Java](#)

#### A.4.1.1 Prerequisites for Java

To run the examples, you must have Coherence version 3.7 and a Java development kit (JDK) 1.6 or greater.

Environment Variable	Description
\$COHERENCE_HOME	Make sure that the COHERENCE_HOME environment variable points to the location of the unpacked Coherence 3.7 directory.
\$JAVA_HOME	Make sure that the JAVA_HOME environment variable points to the location of a 1.6 or greater JDK before building the examples. A Java runtime 1.6 or greater is needed to run the examples.

#### A.4.1.2 Directory Structure for Java

The directory structure described below is relative to the `examples` directory, the directory into which the examples were unzipped.

**Table A-7** Directory Structure for Java

Directory Name	Description
java/bin	Scripts for building and executing examples. There are two sets of scripts. Scripts with no file extension are bash scripts. Scripts with a <code>.cmd</code> file extension are Windows command scripts. The following description refers to the script names without specifying any file extension. <ul style="list-style-type: none"> <li>■ <code>run</code>—Runs an example collection</li> <li>■ <code>run-cache-server</code>—Runs the cache server used for examples</li> <li>■ <code>run-proxy</code>—Runs a proxy node. Optional for some Java examples; required for .NET and C++. This can only be done after the example has been built</li> </ul>
java/classes	The class files output from a build. This directory will not exist until the build script is executed.

**Table A-7 (Cont.) Directory Structure for Java**

Directory Name	Description
java/resource/config	The common Coherence configuration files required by the examples.
java/resource/<example name>	If an example has configuration that is required instead of the common configuration, it will have its own directory. The security example uses configuration files from java/resource/security.
COHERENCE_HOME/lib	Coherence libraries used for compiling and running the examples.
resource	The data file used for the contacts LoaderExample: contacts.csv.

### A.4.1.3 Instructions for Java

Execute the run script. There are two parts to running the example.

#### contacts example

1. Start one or more cache servers: `bin/run-cache-server`. Each execution will start a cache server cluster node. To add additional nodes, execute the command in a new command shell.
2. In a new command shell, run with the name of the example: `bin/run contacts`. The `Driver.main` method will run through the features of the example with output going to the command window (`stdout`).

Starting with Coherence 3.7, an example of the new Query Language feature was added. This example shows how to configure and use a simple helper class `FilterFactory` using the Coherence `InvocationService`.

#### security example

The security example requires Coherence\*Extend, which uses a proxy.

1. Start a proxy: `bin/run-proxy security`.  
Optionally, start one or more cache servers as described in the `contacts` example. The proxy is storage-enabled, so it will act as both a proxy and a cache server node.
2. In a new command shell, run with the name of the example: `bin/run security`. The `Driver.main` method will run through the features of the example with output going to the command window (`stdout`).

#### events example

1. Start one or more cache servers: `bin/run-cache-server`. Each execution will start a cache server cluster node. To add additional nodes, execute the command in a new command shell.
2. In a new command shell, run with the name of the example: `bin/run events`. The `Driver.main` method will run through the features of the example with output going to the command window (`stdout`).

## A.4.2 How to Run the .NET Examples

This section contains the following information:

- [Prerequisites for .NET](#)

- [Directory Structure for .NET](#)
- [Instructions for .NET](#)

#### A.4.2.1 Prerequisites for .NET

To run the examples, you must have Coherence version 3.7 or later for .NET and Visual Studio 2008 or later. To run the examples, you will also need to build the Java examples. The Java version is required because the Coherence\*Extend proxy and cache servers require Java.

Also, the examples depend on Java example classes that must be built before running the proxy and cache server.

#### A.4.2.2 Directory Structure for .NET

The directory structure described below is relative to the "examples" directory.

**Table A-8** *Directory Structure for .NET*

Directory Name	Description
resource	The data file used for the contacts LoaderExample: contacts.csv.

#### A.4.2.3 Instructions for .NET

The following sections contain instructions for running the contacts and security examples.

##### contacts

1. Following the Java instructions, start a proxy server (run-proxy) and zero or more cache servers (run-cache-server).
2. From Visual Studio, start the contacts project without debugging or execute the contacts.exe produced from the build in a command shell. The Driver.Main method will run through the features of the example with the output going to the command window (stdout).

Starting with Coherence 3.7, a new example of the new Query Language feature was integrated. This example shows how configure and use a simple helper class "FilterFactory" using the Coherence InvocationService.

##### security

1. Following the java readme.txt instructions, start a proxy server (java/bin/run-proxy security) and zero or more cache servers.
2. From Visual Studio, start the security project without debugging or execute the contacts.exe produced from the build in a command shell. The Driver.Main method will run through the features of the example with the output going to the command window (stdout).

### A.4.3 How to Run the C++ Examples

This section contains the following information:

- [Prerequisites for C++](#)
- [Directory Structure for C++](#)
- [Instructions for C++](#)

### A.4.3.1 Prerequisites for C++

To build the examples, you must have the appropriate C++ library of Coherence version 3.7. Also you must have a C++ development environment. To run the examples, you will also need to build the Java examples. The Java version is required because the Coherence\*Extend proxy and cache servers require Java. Also, the examples depend on Java example classes that must be built before running the proxy and cache server.

Environment Variable	Description
\$COHERENCE_CPP_HOME	Make sure that the COHERENCE_CPP_HOME environment variable points to the location of the unpacked Coherence 3.7 C++ installation (or later) directory.

The supported C++ compilers are:

- Windows —Microsoft Visual C++ Express/Studio 2008 or later or the equivalent Platform SDK.
- Linux—g++ 4.0
- Mac—g++ 4.0

### A.4.3.2 Directory Structure for C++

The directory structure described below is relative to the `examples` directory.

**Table A-9** Directory Structure for C++

Directory Name	Description
<code>cpp/bin</code>	Scripts for building and executing the examples. Scripts with no file extension are bash scripts. Scripts with a <code>.cmd</code> file extension are Windows command scripts. The following description refers to the script names without specifying any file extension. <ul style="list-style-type: none"> <li>■ <code>run</code>—Runs an example, requires that <code>java/bin/run-proxy</code> is started.</li> </ul>
<code>cpp</code>	All example source organized under the <code>contacts</code> and <code>model</code> directories.
<code>contact/out</code>	The object files output from a build. This directory will not exist until the build script is executed.
<code>resource</code>	The data file used for the contacts LoaderExample: <code>contacts.csv</code> .
<code>cpp/contacts</code>	Contains the <code>contacts</code> example sources.
<code>cpp/security</code>	Contains the <code>security</code> example sources.
<code>cpp/pof</code>	Contains the <code>datamodel</code> sources and any classes that require serialization.
<code>\$COHERENCE_CPP_HOME/include</code>	Contains the Coherence header files.
<code>\$COHERENCE_CPP_HOME/lib</code>	Contains the Coherence library.

### A.4.3.3 Instructions for C++

Execute the `run` scripts. There are two parts to running the example. From within new command shells:

**contacts example**

1. Start one proxy server: `java/bin/run-proxy contacts`.

Optionally, start one or more cache servers: `bin/run-cache-server`. Each execution will start a cache server cluster node. To add additional nodes, execute the command in a new command shell.

2. In a new command shell, execute `run` with the name of the example:

**Running the contacts Example on Windows:**

Type `bin\run.cmd contacts`

**Running the contacts Example on Linux/Mac and Solaris:**

Type `bin/run contacts`

The `Driver.main` method will run through the features of the example with output going to the command window (`stdout`).

Starting with Coherence 3.7, an example of the new Query Language feature was added. This example shows how to configure and use a simple helper class `FilterFactory` using the Coherence `InvocationService`.

**security example**

1. Start one proxy server: `java/bin/run-proxy security`.

Optionally, start one or more cache servers: `bin/run-cache-server`. Each execution will start a cache server cluster node. To add additional nodes, execute the command in a new command shell.

2. In a new command shell, execute `run` with the name of the example:

**Running the security Example on Windows:**

Type `bin\run.cmd security`

**Running the security Example on Linux/Mac and Solaris:**

Type `bin/run security`

The `Driver.main` method will run through the features of the example with output going to the command window (`stdout`).

## A.5 Coherence Basic Features Examples

The Coherence basic features examples are a collection of examples that show how to use the basic features of Coherence using a simplified contact information tracker and includes:

- Basic Data Access—"Getting", "putting" and "removing" data from the Coherence Data Grid. See [Section A.5.3, "Basic Data Access Example."](#)
- Data Loading—Loading example data into the Coherence Data Grid. See [Section A.5.4, "Loader Example."](#)
- Parallel Querying —Querying the Coherence Data Grid including the use of indexes. See [Section A.5.5, "Query Example."](#)
- Observable—Listening for changes to data in the Coherence Data Grid. See [Section A.5.6, "Observer Example."](#)
- Processing—Co-locating data processing with the data itself in the Coherence Data Grid. See [Section A.5.7, "Processor Example."](#)

- Query Language—How to use the new 3.6 Coherence Query Language. See [Section A.5.8, "Query Language."](#)

This example set uses example data represented by these Data Model classes.

**Table A-10 Data Model Classes for the Features Examples**

Name	Description
Address	Address information
Contact	Contact information (includes addresses and phone numbers)
ContactId	The key (contact name) to the contact information
PhoneNumber	Phone number

This example set also ships with a `contacts.csv` file which is a comma-delimited value file containing sample `Contacts` information.

### A.5.1 Running the Example Set

1. Review the following information:
  - [Section A.3, "How to Build the Examples"](#)
  - [Section A.4, "How to Run the Examples"](#)
2. Review the information on the Driver implementation found in [Section A.5.2, "Understanding the Features Driver File."](#)

### A.5.2 Understanding the Features Driver File

The Driver file has a static `main` method that executes all the `Contacts` examples in the following order:

1. `LoaderExample`
2. `QueryExample`
3. `QueryLanguageExample`
4. `ObserverExample`
5. `BasicExample`
6. `ProcessorExample`

The Driver file is implemented in each of the three programming languages supported by Coherence.

Language	Implementation Class
Java	<code>com.tangosol.examples.contacts.Driver</code> in <code>java/src</code>
.NET	<code>Driver</code> in namespace <code>Tangosol.Examples.Contacts</code> in <code>dotnet/src/contacts</code>
C++	<code>Driver</code> in namespace <code>coherence::examples</code> in <code>cpp/contacts</code>

Please refer to this example set's `examples.zip` file for more details on each of the examples outlined below.

## A.5.3 Basic Data Access Example

This example shows the most basic data access features of Coherence including getting, putting and removing data.

### Java

Implementation Class: `com.tangosol.examples.contacts.BasicExample` in `java/src`

- Associate a `ContactId` with a `Contact` in the cache:

```
cache.put(contactId, contact);
```

- Retrieve the `Contact` associated with a `ContactId` from the cache:

```
contact = (Contact) cache.get(contactId);
```

- Remove mapping of `ContactId` to `Contact` from the cache:

```
cache.remove(contactId);
```

### .NET

Implementation Class: `BasicExample` in namespace `Tangosol.Examples.Contacts` in `dotnet/src/contacts`

- Associate a `ContactId` with a `Contact` in the cache:

```
cache.Add(contactId, contact);
```

- Retrieve the `Contact` associated with a `ContactId` from the cache:

```
contact = (Contact)cache[contactId];
```

- Remove mapping of `ContactId` to `Contact` from the cache:

```
cache.Remove(contactId);
```

### C++

Implementation Class: `BasicExample` in namespace `coherence::examples` in `cpp/contacts`

- Associate a `ContactId` with a `Contact` in the cache:

```
hCache->put(vContactId, vContact);
```

- Retrieve the `Contact` associated with a `ContactId` from the cache:

```
vContact = cast<Managed<Contact>::View>(hCache->get(vContactId));
```

- Remove mapping of `ContactId` to `Contact` from the cache:

```
hCache->remove(vContactId);
```

### A.5.3.1 Example Output

The example output (due to ["Observer Example"](#)):

#### **Example A-1 Example Output of the Basic Data Access Example**

```
entry inserted:  
John Nocyefggqo  
Addresses  
Home: 1500 Boylston St.
```

```

null
Obopnof, NM 88824
US
Work: 8 Yawkey Way
null
Ssedhvmdeq, OR 84217
US
Phone Numbers
work: +11 0 707 3776578
Birth Date: 1971-12-31
entry deleted:
John Nocyefggqo
Addresses
Home: 1500 Boylston St.
null
Obopnof, NM 88824
US
Work: 8 Yawkey Way
null
Ssedhvmdeq, OR 84217
US
Phone Numbers
work: +11 0 707 3776578
Birth Date: 1971-12-31

```

## A.5.4 Loader Example

This example loads contacts into the cache from a file or stream.

It demonstrates the most effective way of inserting data into a cache using bulk inserts. This will allow for minimizing the number of network roundtrips between the application and the cache.

### Java

Implementation Class: `com.tangosol.examples.contacts.LoaderExample` in `java/src`  
`cache.putAll(mapBatch);`

### .NET

Implementation Class: `LoaderExample` in namespace `Tangosol.Examples.Contacts` in `dotnet/src/contacts`  
`cache.InsertAll(dictBatch);`

### C++

Implementation Class: `LoaderExample` in namespace `coherence::examples` in `cpp/contacts`  
`hCache->putAll(hMapBatch);`

### A.5.4.1 Example Output

#### *Example A-2 Example Output of the LoaderExample*

```
.....Added 10000 entries to cache
```

## A.5.5 Query Example

`QueryExample` runs sample queries for contacts.

The purpose of this example is to show how to create `Extractors` on cache data and how to create a `KeyExtractor` for the cache keys. It also illustrates how to use the indexes to filter the dataset to efficiently create a matching set. Finally, the example demonstrates how to use some of the built-in cache aggregators to do simple computational tasks on the cache data. A subset of the code is shown below.

## Java

Implementation Class: `com.tangosol.examples.contacts.QueryExample` in `java/src`

- Add an index to make queries more efficient.

```
cache.addIndex(new ChainedExtractor("getHomeAddress.getState"), /*fOrdered*/
false, /*comparator*/ null);
```

- Find all contacts who live in Massachusetts.

```
Set setResults = cache.entrySet(new EqualsFilter("getHomeAddress.getState",
"MA"));
```

- Count contacts who are older than `nAge` for the entire cache dataset.

```
System.out.println("count > " + nAge + ": " + cache.aggregate(new
GreaterFilter("getAge", nAge), new Count()));
```

## .NET

Implementation Class: `QueryExample` in namespace `Tangosol.Examples.Contacts` in `dotnet/src/contacts`

- Add an index to make queries more efficient.

```
cache.AddIndex(new ChainedExtractor("getHomeAddress.getState"), /*fOrdered*/
false, /*comparator*/ null);
```

- Find all contacts who live in Massachusetts.

```
ICacheEntry[] aCacheEntry = cache.GetEntries(new
EqualsFilter("getHomeAddress.getState", "MA"));
```

- Count contacts who are older than `nAge` for the entire cache dataset.

```
Console.WriteLine("count > " + nAge + ": " + cache.Aggregate(new
GreaterFilter("getAge", nAge), new
Count()));
```

## C++

Implementation Class: `QueryExample` in namespace `coherence::examples` in `cpp/contacts`

- Add an index to make queries more efficient.

```
ValueExtractor::View vHomeStateExtractor = ChainedExtractor::create(
ChainedExtractor::createExtractors("getHomeAddress.getState"));
```

- Find all contacts who live in Massachusetts.

```
Object::View voStateName = String::create("MA");
Set::View setResults = hCache->entrySet(
EqualsFilter::create(vHomeStateExtractor, voStateName));
```

- Count contacts who are older than `nAge` for the entire cache dataset.

```
Integer32::View nAge = Integer32::valueOf(58);
```

```

Object::View vResult = hCache->aggregate( (Filter::View)
GreaterFilter::create(vAgeExtractor, nAge), Count::create());
std::cout << "count > " << nAge->getValue() << ": " << vResult << std::endl;

```

### A.5.5.1 Example Output

The example output is large due to 10,000 contacts and several queries. A sample of the query for Massachusetts residents:

#### **Example A-3 Example Output of the Query Example**

```

MA Residents
ConverterEntry{Key="John Scqngqda", Value="John Scqngqda
Addresses
Home: 265 Beacon St.
Oaskxm, MA 88259
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, OK 95744
US
Phone Numbers
work: +11 88 903 8991283
home: +11 98 553 5878221
Birth Date: 1960-01-03"}

```

## A.5.6 Observer Example

ObserverExample demonstrates how to use a `MapListener` to monitor cache events such as when cache data has been inserted, updated, and removed. A subset of the code is shown below.

### Java

Implementation Class: `com.tangosol.examples.contacts.ObserverExample` in `java/src`

- `ContactChangeListener` is a class that implements the `MapListener` interface.

```
cache.addMapListener(new ContactChangeListener());
```

### .NET

Implementation Class: `ObserverExample` in namespace `Tangosol.Examples.Contacts` in `dotnet/src/contacts`

- `ContactChangeListener` is a class that implements the `ICacheListener` interface.

```
cache.AddCacheListener(new ContactChangeListener());
```

### C++

Implementation Class: `ObserverExample` in namespace `coherence::examples` in `cpp/contacts`

- `ContactChangeListener` is a class that extends the `MapListener` type using `Coherence extend macro`.

```
ContactChangeListener::Handle hListener = ContactChangeListener::create();
hCache->addFilterListener(hListener);
```
- Definition of `ContactChangeListener`:

```
class ContactChangeListener
```

```
: public class_spec<ContactChangeListener,  
    extends <MapListener> >
```

There is no immediate output when this example is run. The registered listener outputs the entry when it is inserted, updated, and deleted. For an update, it outputs both the old value and the new value. The changes to entries are caused by running the "[Basic Data Access Example](#)" and the "[Processor Example](#)", so the output happens when those examples are run.

## A.5.7 Processor Example

ProcessorExample demonstrates how to use a processor to modify a set of data in the cache. In the code sample that follows, all Contacts who live in MA will have their work address updated.

### Java

Implementation Class: `com.tangosol.examples.contacts.ProcessorExample` in `java/src`

Helper Class: `com.tangosol.examples.contacts.OfficeUpdater` in `java/src`

- Apply the `OfficeUpdater` on all contacts who live in MA. The `OfficeUpdater` is a class that implements the `InvocableMap.EntryProcessor` interface by extending `AbstractProcessor`.

```
cache.invokeAll(new EqualsFilter("getHomeAddress.getState", "MA"), new  
OfficeUpdater(addrWork));
```

### .NET

Implementation Class: `ProcessorExample` in namespace `Tangosol.Examples.Contacts` in `dotnet/src/contacts`

Helper Class: `OfficeUpdater` in namespace `Tangosol.Examples.Contacts` in `dotnet/src/contacts`

- Apply the `OfficeUpdater` on all contacts who live in MA. The `OfficeUpdater` is a class that implements the `IEntryProcessor` interface by extending `AbstractProcessor`.

```
cache.InvokeAll(new EqualsFilter("getHomeAddress.getState", "MA"), new  
OfficeUpdater(addrWork));
```

### C++

Implementation Class: `ProcessorExample` in namespace `coherence::examples` in `cpp/contacts`

Helper Class: `OfficeUpdater` in namespace `coherence::examples` in `cpp/contacts`

- The `OfficeUpdater` is a class that extends the `UpdaterProcessor` type.

```
class OfficeUpdater  
: public class_spec<OfficeUpdater,  
    extends<UpdaterProcessor>,  
    implements<PortableObject> >
```

- Apply the `OfficeUpdater` on all contacts who live in MA.

```
Filter::View vEqualsFilter = EqualsFilter::create(  
    ChainedExtractor::create(ChainedExtractor::createExtractors(  
        "getHomeAddress.getState")),  
    String::create("MA"));
```

```

    InvocableMap::EntryProcessor::Handle hOffice = OfficeUpdater::create(addrWork);
    Map::View vMap = hCache->invokeAll(vEqualsFilter, hOffice);

```

### A.5.7.1 Example Output

The example Output (due to "Observer Example") is large due to the number of contacts. A sample of output:

#### **Example A-4 Example Output of the Processor Example**

```

entry updated
old value:
John Keau
Addresses
Home: 443 Beacon St.
Ophvovvw, MA 06539
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, FL 86812
US
Phone Numbers
work: +11 8 919 9456102
home: +11 25 759 588823
Birth Date: 1968-12-31
new value:
John Keau
Addresses
Home: 443 Beacon St.
Ophvovvw, MA 06539
US
Work: 200 Newbury St.
Yoyodyne, Ltd.
Boston, MA 02116
US
Phone Numbers
work: +11 8 919 9456102
home: +11 25 759 588823
entry updated
old value:
John Lbggblkd
Addresses
Home: 929 Beacon St.
Trwylbmf, MA 50358
US
Work: Yoyodyne Propulsion Systems
330 Lectroid Rd.
Grover's Mill, AZ 19164
US
Phone Numbers
work: +11 60 699 203810
home: +11 34 149 5018157
Birth Date: 1964-01-02
new value:
John Lbggblkd
Addresses
Home: 929 Beacon St.
Trwylbmf, MA 50358
US
Work: 200 Newbury St.

```

```
Yoyodyne, Ltd.  
Boston, MA 02116  
US  
Phone Numbers  
work: +11 60 699 203810  
home: +11 34 149 5018157  
Birth Date: 1964-01-02  
Birth Date: 1968-12-31
```

## A.5.8 Query Language

This example shows how to run sample queries for contacts.

### Java

Implementation Class: `com.tangosol.examples.query.QueryExample` in `java/src`

- Add indexes to make queries more efficient.

```
cache.addIndex(ff.createExtractor("age"), /*fOrdered*/ true, /*comparator*/  
null);  
cache.addIndex(ff.createExtractor("homeAddress.state"), /*fOrdered*/ false,  
/*comparator*/ null);
```

- Find all contacts who live in Massachusetts.

```
Set setResults = cache.entrySet(ff.createFilter("homeAddress.state = 'MA'"));
```

- Count contacts who are older than `nAge` for the entire cache dataset.

```
final int nAge = 58;  
Object[] aEnv = new Object[] {new Integer(nAge)};  
System.out.println("count > " + nAge + ": " +  
cache.aggregate(ff.createFilter("age > ?1", aEnv), new  
Count()));
```

### .NET

Implementation Class: `SimpleQueryExample` in namespace `Tangosol.Examples.Query` in `dotnet/src/query`

- Add indexes to make queries more efficient.

```
cache.AddIndex(ff.CreateExtractor("age"), /*fOrdered*/ true, /*comparator*/  
null);  
cache.AddIndex(ff.CreateExtractor("homeAddress.state"), /*fOrdered*/ false,  
/*comparator*/ null);
```

- Find all contacts who live in Massachusetts.

```
ICollection results = cache.GetEntries(ff.CreateFilter("homeAddress.state =  
'MA'"));
```

- Count contacts who are older than `age` for the entire cache dataset.

```
const int age = 58;  
object[] env = new object[] { age };  
results = cache.GetEntries(ff.CreateFilter("age > ?1", env));
```

### C++

Implementation Class: `SimpleQueryExample` in namespace `coherence::examples` in `cpp/query`

- Add indexes to make queries more efficient.

```
hCache->addIndex(hff->createExtractor("age"), /*fOrdered*/ true,
/*vComparator*/ NULL);
hCache->addIndex(hff->createExtractor("homeAddress.state"), /*fOrdered*/ false,
/*vComparator*/ NULL);
```

- Find all contacts who live in Massachusetts.

```
Set::View setResults = hCache->entrySet(hff->createFilter("homeAddress.state is
'MA'));
s
```

- Count contacts who are older than nAge for the entire cache dataset.

```
Integer32::View nAge = Integer32::valueOf(58);
ObjectArray::Handle haEnv = ObjectArray::create(1);
haEnv[0] = nAge;
HashMap::Handle hbinds = HashMap::create();
hbinds->put(String::create("nAge"), nAge);
setResults = hCache->entrySet(hff->createFilter("age > ?1", haEnv));
```

### A.5.8.1 Example Output

The example output (due to ["Query Example"](#)):

#### **Example A-5 Example Output of the Query Language Example**

```
MA Residents
ConverterCacheEntry{Key="John Wmbltik", Value="John Wmbltik
Addresses
Home: 785 Beacon St.
Vpmji, MA 34400
US
Work: 200 Newbury St.
Yoyodyne, Ltd.
Boston, MA 02116
US
Phone Numbers
work: +11 62 133 6144503
home: +11 17 238 6189757
Birth Date: 1/1/1968 12:00:00 AM"}
ConverterCacheEntry{Key="John Dtpx", Value="John Dtpx
Addresses
Home: 673 Beacon St.
Mvblms, MA 25889
US
Work: 200 Newbury St.
Yoyodyne, Ltd.
Boston, MA 02116
US
Phone Numbers
work: +11 89 900 8436918
home: +11 32 686 9582798
Birth Date: 1/3/1960 12:00:00 AM"}
.
.
.
count > 58 : 496
```

## A.5.9 Data Generator

Implementation Class: `com.tangosol.examples.contacts.DataGenerator` in `java/src`

The `DataGenerator` has a static `main` method that generates random `Contact` information and stores the results in a comma separated value file. This class was used to generate the `contacts.csv` that is packaged with the `contacts` examples and is included in case more sample data is needed.

It is implemented only in Java.

## A.6 Coherence Security Examples

The Coherence security examples are a collection of examples that show how to use the security features of Coherence in order to provide access control.

These examples are simplified to show only the security features of Coherence. They are not examples of security best practices:

- ["Password Example"](#)—Shows how a Coherence Proxy can require a password to access a cache.
- ["Access Control Example"](#)—Shows simplified role based access control.
- ["Password Identity Transformer"](#)—Creates a custom security token that contains the required password and then adds a list of Principal names.
- ["Password Identity Asserter"](#)—Asserts that the security token contains the required password and then constructs a Subject based on a list of Principal names.
- ["Entitled Cache Service"](#)—Wraps a cache service for access control.
- ["Entitled Invocation Service"](#)—Wraps an invocation service for access control.
- ["Entitled Named Cache"](#)—Wraps a named cache for access control.

### A.6.1 This Example Set

- Gets a cache reference that requires a password.
- Attempts cache and invocation service operations that require different roles.

#### A.6.1.1 Running the Security Example Set

1. Review the following information:
  - [Section A.3, "How to Build the Examples"](#)
  - [Section A.4, "How to Run the Examples"](#)
2. Review the information on the security Driver implementation found in the next section.

#### A.6.1.2 Understanding the Security Driver File

Has a static `main` method that executes all the security examples in the following order:

1. `PasswordExample`
2. `AccessControlExample.accessCache()`
3. `AccessControlExample.accessInvocationService()`

Is implemented in each of the three programming languages supported by Coherence:

Language	Implementation Class
Java	<code>com.tangosol.examples.security.Driver</code> in <code>java/src</code>
.NET	<code>Driver</code> in namespace <code>Tangosol.Examples.Security</code> in <code>dotnet/src/security</code>
C++	<code>Driver</code> in namespace <code>coherence::examples</code> in <code>cpp/security</code>

Please refer to this example set's `example.zip` file for more details on each of the examples outlined below.

## A.6.2 Password Example

This example shows how a Coherence Proxy can require a password to get a reference to a cache.

### Java

Implementation Class: `com.tangosol.examples.security.PasswordExample` in `java/src`

The code logs in to get a `Subject`, and then tries to get a cache reference running in the context of the `Subject`.

The [Password Identity Transformer](#) will generate a security token that contains the password. The [Password Identity Asserter](#) (running in the proxy) will validate the security token to enforce the password. The token generation and validation occurs automatically when a connection to the proxy is made.

### .NET

Implementation Class: `PasswordExample` in namespace `Tangosol.Example.Security` in `dotnet/src/security`

The code logs in to get a `Principal`, and then tries to get a cache reference running in the context of the `Principal` by making the `Principal` the `Thread`'s current principal.

The [Password Identity Transformer](#) will generate a security token that contains the password. The [Password Identity Asserter](#) (running in the proxy) will validate the security token to enforce the password. The token generation and validation occurs automatically when a connection to the proxy is made.

### C++

Implementation Class: `AccessExample` in namespace `coherence::examples` in `cpp/security`

The code logs in to get a `Subject`, and then tries to get a cache reference running in the context of the `Subject`.

The [Password Identity Transformer](#) will generate a security token that contains the password. The [Password Identity Asserter](#) (running in the proxy) will validate the security token to enforce the password. The token generation and validation occurs automatically when a connection to the proxy is made.

The example Output:

### **Example A-6 Example Output of the Password Example**

```
-----password example begins-----
-----password example succeeded-----
```

-----password example completed-----

### A.6.3 Access Control Example

This example shows simplified role-based access control.

#### Java

Implementation Class: `com.tangosol.examples.security.AccessControlExample` in `java/src`

The code logs in to get a `Subject` with a user-id with a particular role, gets a cache reference running in the context of the `Subject`, and then tries cache operations. Depending on the role granted to the user, the cache operation is allowed or denied.

Someone with a `writer` role is allowed to put and get. Someone with a `reader` role can get but not put. Someone with a `writer` role cannot destroy a cache. Someone with an `admin` role can destroy a cache.

Then a user with a particular role tries to use the invocation service. A `reader` is not allowed to invoke, but a `writer` is allowed.

Note that once the cache or invocation service reference is created in the context of a `Subject`, that identity is permanently associated with that reference. Any use of that cache or service reference is on behalf of that identity.

The [Password Identity Transformer](#) will generate a security token that contains the password, the user-id, and the roles. The [Password Identity Asserter](#) (running in the proxy) will validate the security token to enforce the password, and construct a `Subject` with the proper user-id and roles.

The production and assertion of the security token happens automatically.

See the [Entitled Cache Service](#), [Entitled Invocation Service](#), and [Entitled Named Cache](#) code for the implementation of access control.

#### .NET

Implementation Class: `AccessControlExample` in namespace `Tangosol.Example.Security` in `dotnet/src/security`

The code logs in to get a `Principal` with a user-id with a particular role, gets a cache reference running in the context of the `Principal`, and then tries cache operations. Depending on the role granted to the user, the cache operation is allowed or denied.

Someone with a `writer` role is allowed to put and get. Someone with a `reader` role can get but not put. Someone with a `writer` role cannot destroy a cache. Someone with an `admin` role can destroy a cache.

Then a user with a particular role tries to use the invocation service. A `reader` is not allowed to invoke, but a `writer` is allowed.

Note that once the cache or invocation service reference is created in the context of a `Principal`, that identity is permanently associated with that reference. Any use of that cache or service reference is on behalf of that identity.

The [Password Identity Transformer](#) will generate a security token that contains the password, the user-id, and the roles. The [Password Identity Asserter](#) (running in the proxy) will validate the security token to enforce the password, and construct a `Subject` with the proper user-id and roles.

The production and assertion of the security token happens automatically.

See the [Entitled Cache Service](#), [Entitled Invocation Service](#), and [Entitled Named Cache](#) code for the implementation of access control.

## C++

Implementation Class: `AccessControlExample` in namespace `coherence::examples` in `cpp/security`

The code logs in to get a `Subject` with a user-id with a particular role, gets a cache reference running in the context of the `Subject`, and then tries cache operations. Depending on the role granted to the user, the cache operation is allowed or denied.

Someone with a `writer` role is allowed to put and get. Someone with a `reader` role can get but not put. Someone with a `writer` role cannot destroy a cache. Someone with an `admin` role can destroy a cache.

Then a user with a particular role tries to use the invocation service. A `reader` is not allowed to invoke, but a `writer` is allowed.

Note that once the cache or invocation service reference is created in the context of a `Subject`, that identity is permanently associated with that reference. Any use of that cache or service reference is on behalf of that identity.

The [Password Identity Transformer](#) will generate a security token that contains the password, the user-id, and the roles. The [Password Identity Asserter](#) (running in the proxy) will validate the security token to enforce the password, and construct a `Subject` with the proper user-id and roles.

The production and assertion of the security token happens automatically.

See the [Entitled Cache Service](#), [Entitled Invocation Service](#), and [Entitled Named Cache](#) code for the implementation of access control.

### A.6.3.1 Example Output

The example output:

#### *Example A-7 Example Output of the Access Control Example*

```
-----cache access control example begins-----
Success: read and write allowed
Success: read allowed
Success: Correctly cannot write
Success: Correctly cannot destroy the cache
Success: Correctly allowed to destroy the cache
-----cache access control example completed-----
-----InvocationService access control example begins-----
Success: Correctly allowed to use the invocation service
Success: Correctly unable to use the invocation service
-----InvocationService access control example completed-----
```

## A.6.4 Password Identity Transformer

This example shows how an `IdentityTransformer` produces a security token from an identity.

### Java

Implementation Class:  
`com.tangosol.examples.security.PasswordIdentityTransformer` in `java/src`

The code produces a security token that is an array of strings. The first string is the password. The second string is the user-id and subsequent strings are the user's roles. Arrays of strings will be serialized by Coherence\*Extend without requiring a custom serializer.

This class will be invoked automatically when the Extend client connects to a proxy or a channel is opened in an existing connection.

### **.NET**

Implementation Class: PasswordIdentityTransformer in namespace  
Tangosol.Example.Security in dotnet/src/security

The code produces a security token that is an array of strings. The first string is the password. The second string is the user-id and subsequent strings are the user's roles. Arrays of strings will be serialized by Coherence\*Extend without requiring a custom serializer.

This class will be invoked automatically when the Extend client connects to a proxy or a channel is opened in an existing connection.

### **C++**

Implementation Class: PasswordIdentityTranfromer in namespace  
coherence::examples in cpp/security

The code produces a security token that is an array of strings. The first string is the password. The second string is the user-id and subsequent strings are the user's roles. Arrays of strings will be serialized by Coherence\*Extend without requiring a custom serializer.

This class will be invoked automatically when the Extend client connects to a proxy or a channel is opened in an existing connection.

## **A.6.5 Password Identity Asserter**

This example shows how an IdentityAsserter validates a security token and produces a Subject from a list of principal names.

### **Java**

Implementation Class:  
com.tangosol.examples.security.PasswordIdentityAsserter in java/src

The code processes a security token that should be an array of strings. The first string must be the password. Subsequent strings are principals. Any failure processing the token results in a SecurityException that will deny access to the proxy.

### **.NET**

Implementation Class: none

The IdentityAsserter runs only on the proxy (in Java), so it does not run in the .NET client. Therefore, there is no PasswordIdentityAsserter for .NET.

### **C++**

Implementation Class: none

The PasswordIdentityAsserter runs only on the proxy (in Java), so it does not run in the C++ client. Therefore there is no PasswordIdentityAsserter for C++.

## A.6.6 Entitled Cache Service

This example shows how a remote cache service can be wrapped to provide access control.

### Java

Implementation Class: `com.tangosol.examples.security.EntitledCachService` in `java/src`

The code instantiates an [Entitled Named Cache](#) that provides access control for cache operations. The code also provides access control for the cache service methods `release` and `destroy`. The access control check is delegated to the [Security Example Helper](#).

This class will be instantiated automatically when the cache service is started on the proxy.

### .NET

There is no .NET implementation. The class runs only on the proxy in Java.

### C++

There is no C++ implementation. The class runs only on the proxy in Java.

## A.6.7 Entitled Invocation Service

This example shows how a remote invocation service can be wrapped to provide access control.

### Java

Implementation Class: `com.tangosol.examples.security.EntitledInvocationService` in `java/src`

The code provides access control for the invocation service methods. The access control check is delegated to the [Security Example Helper](#).

This class will be instantiated automatically when the invocation service is started on the proxy.

### .NET

There is no .NET implementation. The class runs only on the proxy in Java.

### C++

There is no C++ implementation. The class runs only on the proxy in Java.

## A.6.8 Entitled Named Cache

This example shows how a remote named cache can be wrapped to provide access control.

### Java

Implementation Class: `com.tangosol.examples.security.EntitledNamedCache` in `java/src`

The code provides access control for the `NamedCache` methods. The access control check is delegated to the [Security Example Helper](#).

This class will be instantiated automatically when the cache service is started on the proxy.

#### **.NET**

There is no .NET implementation. The class runs only on the proxy in Java.

#### **C++**

There is no C++ implementation. The class runs only on the proxy in Java.

### **A.6.9 Security Example Helper**

This example is a helper class for authentication and access control.

#### **Java**

Implementation Class: `com.tangosol.examples.security.SecurityExampleHelper` in `java/src`

The code simulates authentication. For the sake of simplicity, it creates a `Subject`. A real implementation would do platform- and company-specific authentication. The login also does simple mapping of user names to roles.

The `checkAccess` method checks that the operation is allowed by the user's role.

#### **.NET**

Implementation Class: `SecurityExampleHelper` in namespace `Tangosol.Example.Security` in `dotnet/src/security`

The code simulates authentication. For the sake of simplicity, it creates a `Principal`. A real implementation would do platform- and company-specific authentication. The login also does simple mapping of user names to roles.

#### **C++**

Implementation Class: `SecurityExampleHelper` in namespace `coherence::examples` in `cpp/security`

The code simulates authentication. For the sake of simplicity, it creates a `Subject`. A real implementation would do platform- and company-specific authentication. The login also does simple mapping of user names to roles.

## **A.7 Coherence Live Events Examples**

These examples illustrate the various event types in Coherence Live Events and how they can be consumed, including `EntryEvents`, `EntryProcessorEvents` and `TransferEvents`.

The Live Events Examples are available only in the Java programming language, as they are executed on the storage-enabled members of the partitioned service.

- ["EventsExamples"](#)—Illustrates various features within Live Events.
- ["TimedTraceInterceptor"](#)—Provides timings between pre- and post-commit events for different types of events.
- ["CantankerousInterceptor"](#)—Responds with runtime exceptions at either pre- or post-commit time, based on the type of key being inserted.
- ["RedistributionInterceptor"](#)—Logs partition events when enabled.

- ["RedistributionInvocable"](#)—Defines three actionable states that will be executed on various members of the cluster. The states are enable logging performed by the `RedistributionInterceptor`, disable logging, or terminate the JVM that the invocable (`RedistributionInvocable`) is executed on.
- ["LazyProcessor"](#)—Creates a superficial delay between the processing of events.

## A.7.1 This Example Set

- Illustrates how to measure the elapsed time between pre- and post-events which are inserted into a results cache.
- Illustrates the semantics of throwing exceptions in pre- and post-commit events.
- Illustrates how partition redistribution events can be logged.

### A.7.1.1 Running the Live Events Example Set

1. Review the following information:
  - [Section A.3, "How to Build the Examples"](#)
  - [Section A.4, "How to Run the Examples"](#)
2. Review the information on the Live Events Driver implementation found in the next section.

### A.7.1.2 Understanding the Live Events Driver File

Has a static `main` method that executes all the Live Events examples in the following order:

1. Timed Events Example
2. Veto Events Example
3. Partition Transfer Events Example

Is implemented only in the Java programming language:

Language	Implementation Class
Java	<code>com.tangosol.examples.events.Driver</code> in <code>java/src</code>

## A.7.2 EventsExamples

Implementation Class: `com.tangosol.examples.events.EventsExamples` in `java/src`

The `EventsExamples` class illustrates various features within Live Events. This includes:

- Providing mean elapsed times split by event type.
- Illustrating the different semantics in throwing exceptions in pre-events compared to post-events.
- Illustrating logging of partition movement when enabled.

The `EventsExamples` class defines these inner classes:

- [EventsTimingExample](#)
- [VetodEventsExample](#)
- [RedistributionEventsExample](#)

### A.7.2.1 EventsTimingExample

The `EventsTimingExample` inner class is a catalyst for action to be performed by [TimedTraceInterceptor](#). This illustrates how the elapsed time between pre- and post-events can be measured which are inserted into a results cache. The entries inserted into the results cache are displayed by using the `stdout` of the process executing this class.

The example output:

#### **Example A-8 Example Output of the EventsTimingExample**

```
Received stats [memberId=2, eventType=INSERTED, sample=1] = EventStats[name =
INSERTED, sampleMean = 0.294040ms, mean = 0.294040ms]
Received stats [memberId=3, eventType=INSERTED, sample=1] = EventStats[name =
INSERTED, sampleMean = 0.397855ms, mean = 0.397855ms]
Received stats [memberId=1, eventType=INSERTED, sample=1] = EventStats[name =
INSERTED, sampleMean = 0.373270ms, mean = 0.373270ms]
Received stats [memberId=3, eventType=UPDATED, sample=1] = EventStats[name =
UPDATED, sampleMean = 0.187132ms, mean = 0.187132ms]
Received stats [memberId=2, eventType=UPDATED, sample=1] = EventStats[name =
UPDATED, sampleMean = 0.234314ms, mean = 0.234314ms]
Received stats [memberId=1, eventType=UPDATED, sample=1] = EventStats[name =
UPDATED, sampleMean = 0.237622ms, mean = 0.237622ms]
```

### A.7.2.2 VetodEventsExample

The `VetodEventsExample` inner class is a catalyst for action to be performed by [CantankerousInterceptor](#). This illustrates the semantics of throwing exceptions in pre- and post-events. The exceptions that are expected to only be logged are inserted into a results cache. The entries inserted into the results cache are displayed by using the `stdout` of the process executing this class.

The example output:

#### **Example A-9 Example Output of the VetodEventsExample**

```
Received event [memberId=3, eventType=NON_VETO, count=1] = Objection falls on deaf
ears! value = value: 11
Received event [memberId=3, eventType=NON_VETO, count=2] = Objection falls on deaf
ears! value = value: 22
Received event [memberId=3, eventType=NON_VETO, count=3] = Objection falls on deaf
ears! value = value: 33
Received event [memberId=3, eventType=NON_VETO, count=4] = Objection falls on deaf
ears! value = value: 44
```

### A.7.2.3 RedistributionEventsExample

The `RedistributionEventsExample` inner class is a catalyst for action to be performed by the [RedistributionInterceptor](#) class. This illustrates how partition redistribution events can be logged, by enabling logging in the `RedistributionInterceptor` and killing a member thus inducing partition redistribution.

The example output:

#### **Example A-10 Output of the RedistributionEventsExample**

```
Choosing to kill member Member(Id=3, Timestamp=2014-01-02 16:38:17.942,
Address=10.159.154.103:8092, MachineId=47251,
Location=site:,machine:TPFAEFFL-LAP,process:8168, Role=CoherenceServer)
```

### A.7.3 TimedTraceInterceptor

Implementation Class: `com.tangosol.examples.events.TimedTraceInterceptor` in `java/src`

The `TimedTraceInterceptor` class provides timings between pre- and post-commit events for each type of event; that is, inserts, updates, removes, and entry processor execution. These timings are collected and averaged at a sample rate defined by parameter `cSample`. Additionally they are output to the log at the same time. This implementation does maintain a strong reference to the each binary key however this is removed upon receiving the post-commit event for the same key.

The interceptor implements the `EventInterceptor` interface. The `@Interceptor` annotation provides the unique name of the interceptor with the `identifier` attribute and the order in which it should be executed (`Order.HIGH`) with the `order` attribute.

The interceptor also contains a protected `EventTimer` inner-class. This class times the elapsed time for each event it is notified of. The interceptor tracks the time between a pre- and post-commit event for each entry and the respective event types (`INSERT`, `UPDATE`, `REMOVE`). The timings are sent to the Coherence log in batches displaying sample and cumulative statistics.

As the generic argument is `com.tangosol.net.events.partition.cache.Event`, you will only get events that are consumers of that event, that is, `EntryEvent` and `EntryProcessorEvent`, without specifying any filtering.

### A.7.4 CantankerousInterceptor

Implementation Class: `com.tangosol.examples.events.CantankerousInterceptor` in `java/src`

The `CantankerousInterceptor` class is an `EventInterceptor` implementation that is argumentative in nature, hence the event of inserting certain keys will result in runtime exceptions at either pre- or post-commit phases.

If the exception is thrown at pre-commit time, then a rollback occurs and the exception is propagated to the client. If the exception occurs at post-commit time, then a log event is recorded. The keys used for the exceptions are `VETO` and `NON-VETO`. `INSERTING` and `UPDATING` are events that can be vetoed, whereas `INSERTED` and `UPDATED` events cannot be vetoed.

### A.7.5 RedistributionInterceptor

Implementation Class: `com.tangosol.examples.events.RedistributionInterceptor` in `java/src`

The `RedistributionInterceptor` class is an `EventInterceptor` that logs partition activity when enabled. Logging can be enabled by using setting the `RedistributionInvocable.ENABLED` constant.

### A.7.6 RedistributionInvocable

Implementation Class: `com.tangosol.examples.pof.RedistributionInvocable` in `java/src`

The `RedistributionInvocable` class defines three actionable states that will be executed on various members of the cluster. For this example, define the states as follows:

- **DISABLE:** Disable the logging performed by the `RedistributionInterceptor` event interceptor.
- **ENABLE:** Enable the logging performed by the `RedistributionInterceptor` event interceptor.
- **KILL:** Terminate the JVM that this invocable (`RedistributionInvocable`) is executed on.

### A.7.7 LazyProcessor

Implementation Class: `com.tangosol.examples.pof.LazyProcessor` in `java/src`

The `LazyProcessor` class creates a superficial delay between the processing of events. The class specifies the number of milliseconds this processor should sleep between processing events. This class will be used by the [EventsTimingExample](#) subclass in the [EventsExamples](#) class.