Coherence Hands-On Lab

Workbook

Rev 4.4.1 – 25th February, 2008

Table of Contents

Table of Contents 2

Coherence Hands-On Lab 3
Coherence Overview 3
Labs Overview...4
Labs Files...4
Author..4

Lab 1: Install Oracle Coherence and Configure JDeveloper 11g...5

Lab 2: Run the sample Coherence Application 10

Lab 3 – Access the data grid from Java..15

Lab 4 - Leaving Strings behind - Working with complex Objects 25

Lab 5 – Querying & Aggregating Data in a cache...34

Lab 6 - Observing Data Changes..39

Lab 7 – In-Place processing of data 44

Lab 8 – Extending your objects with Cache Keys & Data Affinity...48

Lab 9 – Using JPA with Coherence...53

Where to next? 61

Lab 3 Solutions: Access the data grid from Java...63

Lab 4 Solutions: Leaving Strings behind - Working with complex Objects................................65

Lab 5 Solutions - Querying & Aggregating Data in a cache 71

Lab 6 – Solutions – Observing Data Changes 76

Lab 7 Solutions - In-Place processing of data 82

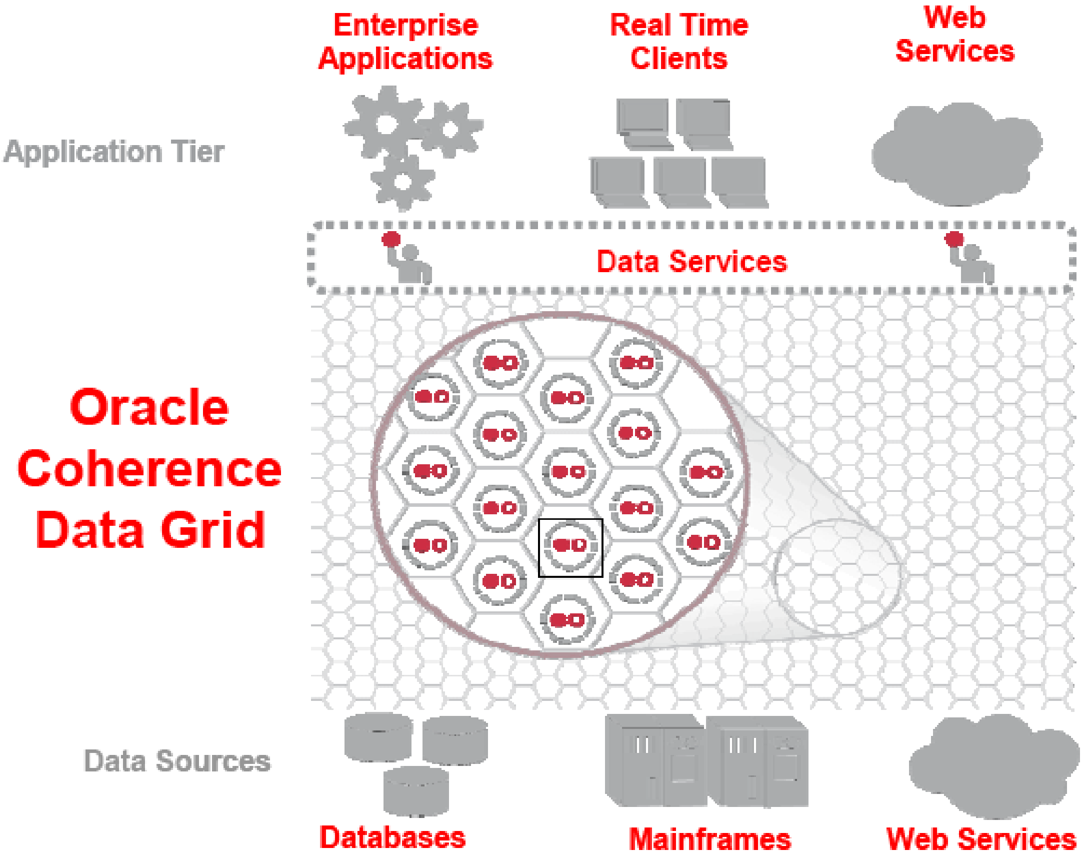
Lab 8 Solutions - Extending your objects with Cache Keys & Data Affinity..............................87

Lab 9 Solutions – Using JPA with Coherence..95

Coherence Overview

Oracle 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 middleware that reliably manages data objects in memory across many servers. By automatically and dynamically partitioning data, Oracle Coherence enables continuous data availability and transactional integrity, even in the event of a server failure. Oracle Coherence provides organizations with a robust, scale-out data abstraction layer.

For more information see http://www.oracle.com/products/middleware/coherence/index.html.



Labs Overview

This document contains hands on labs containing step by step instructions on how to install, configure and use Oracle Coherence (formerly Tangosol Coherence). The following labs are currently available

- Lab 1 – Install Oracle Coherence and Configure Jdeveloper 11g.
- Lab 2 – Run the sample coherence application.
- Lab 3 – Access the data grid from Java
- Lab 4 – Leaving Strings behind - Working with complex Objects.
- Lab 5 – Querying & Aggregating Data in a cache
- Lab 6 – Observing Data Changes
- Lab 7 – In-Place processing of data
- Lab 8 – Extending your objects with Cache Keys & data affinity
- Lab 9 – Using JPA with Coherence

Labs Files

The following files are required for this lab.

- OracleCoherenceLabs.zip – contains all lab application files
- coherence-331.zip– current coherence version 3.3.1
- jdevstudio1111.zip – Jdeveloper 11g
- OracleXe.exe – Only required for JPA Lab
- JDK 5.0 Update 12 or above

Author

This document was created by Tim Middleton (Oracle Corporation).

Thanks to the following people who have provided feedback & input: Ancil McBarnett, Brian Oliver, Raanan Dagan & Mark Randell.

Lab 1: Install Oracle Coherence and Configure JDeveloper 11g

The purpose of this lab is to install & setup the environment for running Coherence and building the lab examples.

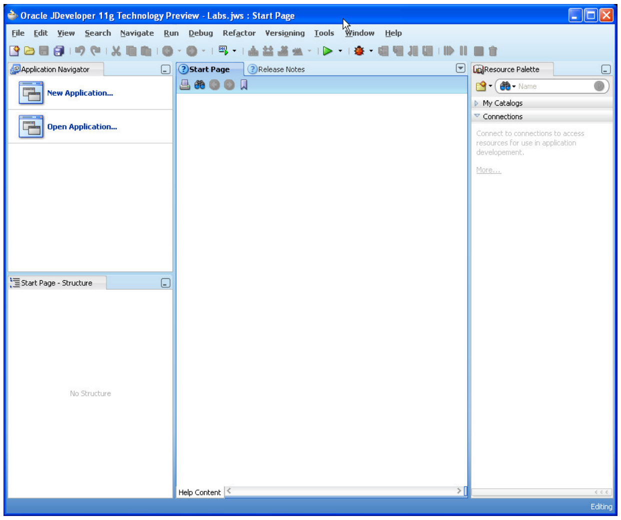
Unzip the required files

- Unzip the file OracleCoherenceLabs.zip to the root directory where you want to install the lab files. E.g. D:\ . This should create the directory D:\OracleCoherenceLabs. We will refer to this as [LABS_HOME]. It should contain the following directories:
  - LabsSolns – solutions for the labs
  - Labs – directory to create your solutions
- Unzip coherence331.zip into [LABS_HOME]. This will create the directory [LABS_HOME]\coherence with the following sub-directories:
  - bin – command scripts
  - lib – required library files
  - examples – example code
  - doc – javadoc
- Unzip jdevstudio1111.zip into a directory of your choosing. **Ensure that the directory name you unzip into does not have any spaces.** We will refer to this directory as [JDEV_HOME]

Startup and Configure Jdeveloper 11g

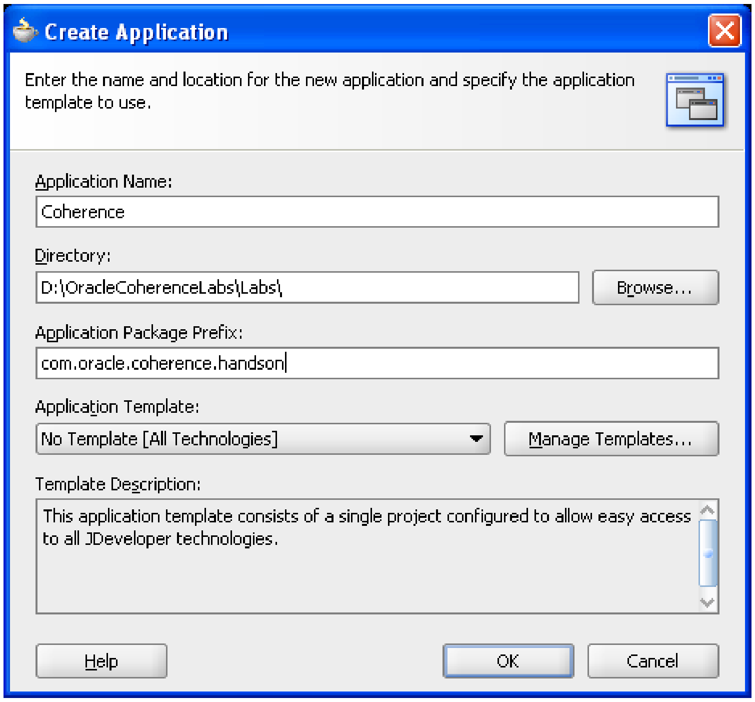
- Run the jdeveloper.exe file in the [JDEV_HOME]
- If you get a message asking to migrate from previous versions of JDeveloper, answer no.

- Once JDeveloper starts up you should see a screen similar to the following.



- Click on *New Application*, to create an application. An Application is a way of grouping projects or source code. This new application that will hold the Coherence Projects.

- In the dialog, change the application name to *Coherence* , the directory to [LABS_HOME]\labs .e g. D:\OracleCoherenceLabs\Labs\ , and the application package prefix to *com.oracle.coherence.handson*.



Enter the name and location for the new application and specify the application template to use.

Application Name: Coherence

Directory: D:\OracleCoherenceLabs\Labs\

Application Package Prefix: com.oracle.coherence.handson

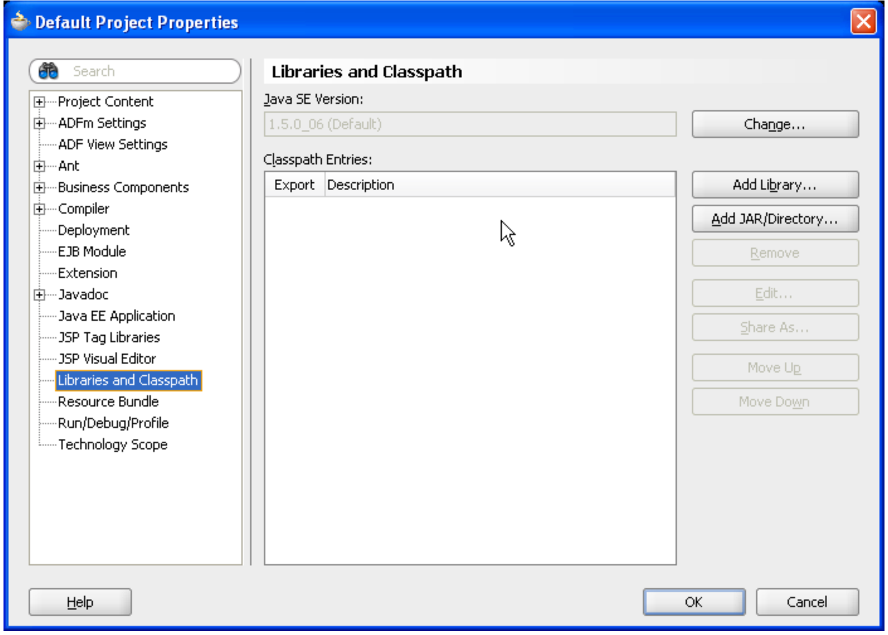
Application Template: No Template [All Technologies]

Template Description: This application template consists of a single project configured to allow easy access to all JDeveloper technologies.

- Click on *OK* and then click on *Cancel* when prompted to create a new project. We will do this later on.

- We will now add the Coherence jars to the default project properties. We will also define two run profiles to simplify running of cache servers.

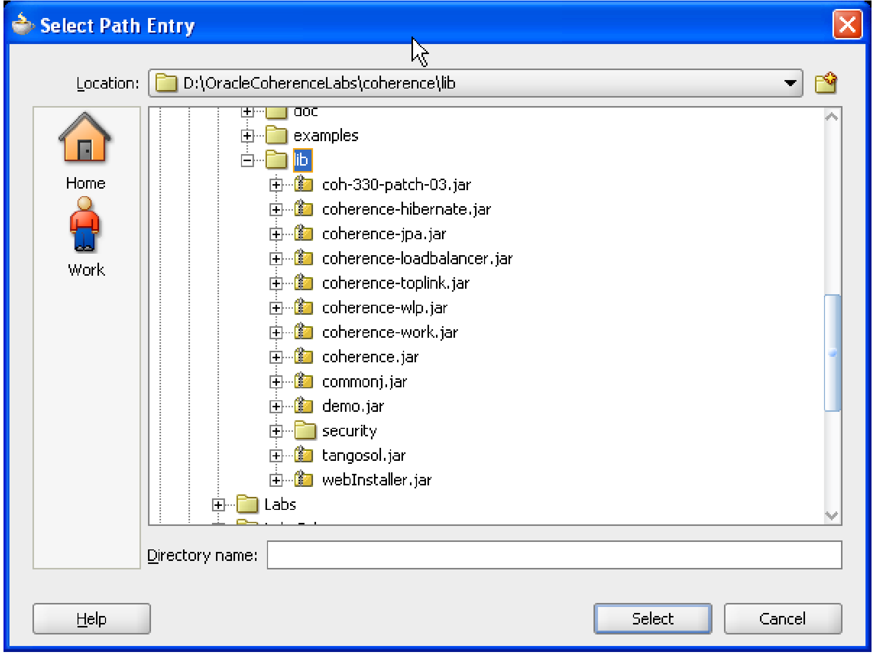
- From the *Tools* menu click on *Default Project Properties*. The following window will be displayed.



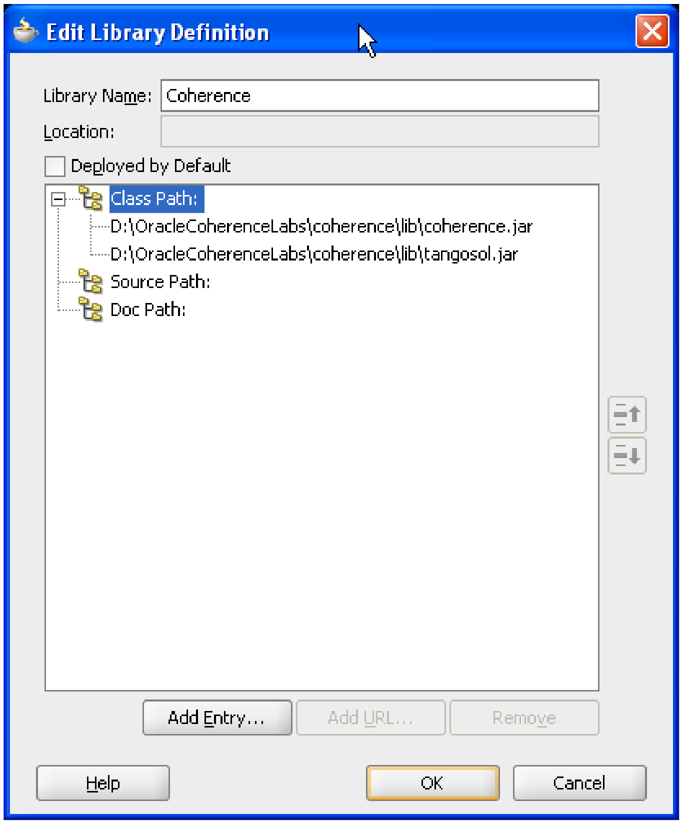
- Click on *Libraries and Classpath* and click the *Add Library* button.

- In the *add Library* dialog, click on *New*.

- Click on *Add Entry* and add find and expand the coherence\lib directory which is under the [LABS_HOME] directory.



- Highlight and select the following files in order to add them to the classpath:
    - coherence.jar
    - tangosol.jar
- Once you change the library name to *Coherence* the dialog should look like this:



Edit Library Definition

Library Name: Coherence

Location:

☐ Deployed by Default

Class Path:
  D:\OracleCoherenceLabs\coherence\lib\coherence.jar
  D:\OracleCoherenceLabs\coherence\lib\tangosol.jar
Source Path:
Doc Path:

- Click on *OK*, *OK* and *OK* to save your changes.
- Oracle JDeveloper is now setup with the correct Coherence Libraries. In Lab 3 we will do some coding within JDeveloper.

Create a run profile to run a cache server.

- From the *Tools* menu click on *Default Project Properties*. Choose *Run/Debug/Profile*.
- Click on *New* to create a new Profile.
- Name this *DefaultCacheServer* and click on *OK*.
- Click on *Edit* to modify the newly created run profile.
- Set the following values:

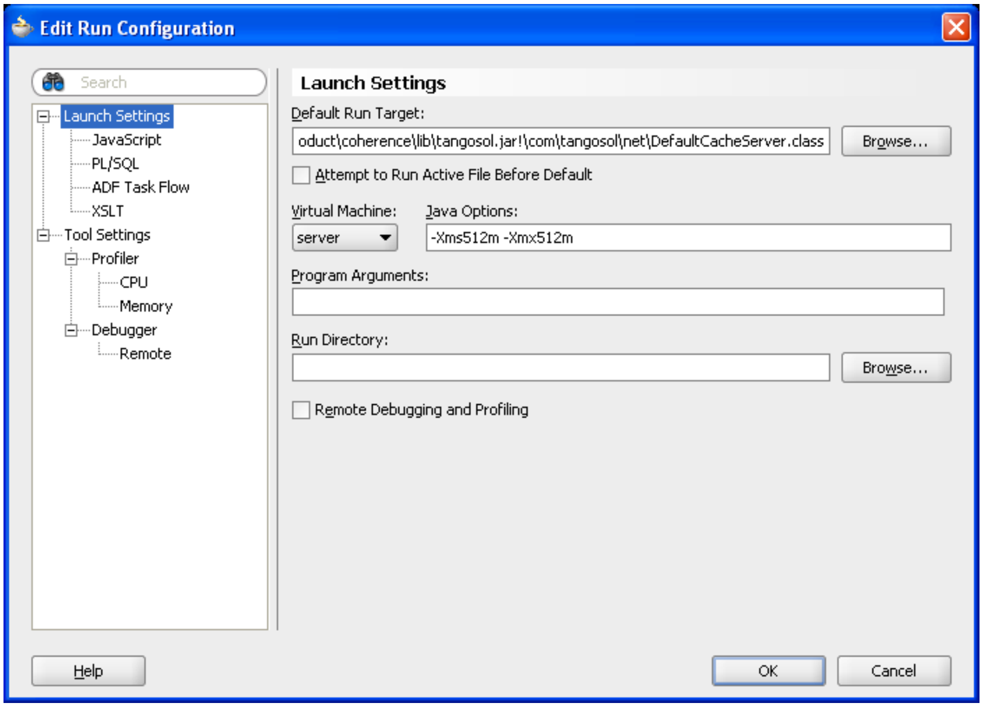
- Change *Virtual Machine* to *Server*

- Uncheck *Attempt to run active file…*

- In the *Default Run Target Field* browse to your Coherence lib directory as before and browse through the *tangosol.jar* file and select the following: com.tangosol.net.DefaultCacheServer.class. The default run target should now be similar to the following: D:\product\coherence\lib\tangosol.jar!\com\tangosol\net\DefaultCacheServer.class

- Put the following in the *Java Options* field:
  -Xms512m -Xmx512m

- Your dialog should now look like this:



Launch Settings

Default Run Target: oduct\coherence\lib\tangosol.jar!\com\tangosol\net\DefaultCacheServer.class

☐ Attempt to Run Active File Before Default

Virtual Machine: server

Java Options: -Xms512m -Xmx512m

Program Arguments:

Run Directory:

☐ Remote Debugging and Profiling

- From now on you can start a Cache Server using this run profile.

Lab 2: Run the sample Coherence Application

In this lab we run the sample Coherence cache servers and Map application to get a feel for some of the features and functionality of Coherence.

1. Setup JAVA_HOME for Oracle Coherence

    - In the coherence.cmd and cache-server.cmd (located in [LABS_HOME\coherence\bin set the JAVA_HOME variable to point to your JSDK 1.5 environment. If you don't have a JSDK you can install one from the DVD provided.

      SET JAVA_HOME=C:\PROGRA~1\Java\jdk1.5.0_11

    - In Windows explorer change to the [LABS_HOME]\coherence\bin directory.

    - In the file coherence.cmd (or coherence.sh), find the line with set storage_enabled and change the value **false** to **true**.

      set storage_enabled=**true**

      This option specifies if a process is a "Storage Enabled Member" when joining the cluster. Storage enabled members can store data as part of a Coherence cluster. Client processes that join and leave a cluster to carry out processing regularly should not be storage enabled. **What do you think the reason is for this?**

    - Save the files and exit.

Run the samples

- From the [LABS_HOME\coherence\bin directory run the following file: cache-server.cmd

- This will start up a storage enabled member in the cluster. When you start the first cache-server up you will notice a slight delay as the cache server looks for an existing cluster. Once it determines there are no clusters to join, it will start one.

  Note: by default Coherence uses multicast to search for cluster members only. Multicast is not used for data transfer. If you cannot use or do not want to use multicast then this can be configured.

- You should see a number of messages displayed. Some of the key messages are highlighted below.

```
** Starting storage enabled console **
java version "1.5.0_11"  [Java Version]
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_11-b03)
Java HotSpot(TM) Server VM (build 1.5.0_11-b03, mixed mode)

[Information about loading of config files. Default is to load from JAR]
2007-08-23 15:16:51.359 Oracle Coherence 3.3/387 <Info> (thread=main,
member=n/a
): Loaded operational configuration from resource
"jar:file:/D:/OracleCoherenceLabs/coherence/lib/coherence.jar!/tangosol-
coherence.xml"
2007-08-23 15:16:51.359 Oracle Coherence 3.3/387 <Info> (thread=main,
member=n/a
): Loaded operational overrides from resource
"jar:file:/D:/OracleCoherenceLabs/coherence/lib/coherence.jar!/tangosol-
coherence-override-dev.xml"
2007-08-23 15:16:51.375 Oracle Coherence 3.3/387 <D5> (thread=main,
member=n/a):
 Optional configuration override "/tangosol-coherence-override.xml" is not
specified

[Coherence Version]
Oracle Coherence Version 3.3/387
 Grid Edition: Development mode
Copyright (c) 2000-2007 Oracle. All rights reserved.

2007-08-23 15:16:52.265 Oracle Coherence GE 3.3/387 <D5> (thread=Cluster,
member
=n/a): Service Cluster joined the cluster with senior service member n/a
2007-08-23 15:16:55.515 Oracle Coherence GE 3.3/387 <Info>
(thread=Cluster, member=n/a): Created a new cluster with Member(Id=1,
Timestamp=2007-08-23 15:16:52.0
15, Address=192.168.200.1:8088, MachineId=40961,
Location=process:3668@tmiddlet-
au, Edition=Grid Edition, Mode=Development, CpuCount=2, SocketCount=1)
UID=0xC0A8C801000001149194A6AFA0011F98 SafeCluster: Name=n/a

[Mutli cast address - the default changes with each version. Note the
.3.3.0 for version 3.3.]
Group{Address=224.3.3.0, Port=33387, TTL=4}

[Member set]
MasterMemberSet
  (
  ThisMember=Member(Id=1, Timestamp=2007-08-23 15:16:52.015,
Address=192.168.200.1:8088, MachineId=40961,
Location=process:3668@tmiddlet-au)
  OldestMember=Member(Id=1, Timestamp=2007-08-23 15:16:52.015,
Address=192.168.200.1:8088, MachineId=40961,
Location=process:3668@tmiddlet-au)
  ActualMemberSet=MemberSet(Size=1, BitSetCount=2
    Member(Id=1, Timestamp=2007-08-23 15:16:52.015,
Address=192.168.200.1:8088,
MachineId=40961, Location=process:3668@tmiddlet-au)
    )
  RecycleMillis=120000
  RecycleSet=MemberSet(Size=0, BitSetCount=0
    )
  )

[Services configured for this member]
Services
  (
  TcpRing{TcpSocketAccepter{State=STATE_OPEN,
ServerSocket=192.168.200.1:8088}, Connections=[]}
  [Hand holding TCP connection]
  ClusterService{Name=Cluster, State=(SERVICE_STARTED, STATE_JOINED),
Id=0, Version=3.3, OldestMemberId=1}
  [Manages cluster membership]
  )
```

- Run coherence.cmd;

- You should now see the following prompt:
  Map (?):

- This application shows you the basic distributed cache functionality build within Coherence.

- Type in the following to create a new named cache called *test*.
  cache test

  (Each named cache can be thought of as a table. A cluster can have many named caches. Each named cache holds one type of object. It can be a simple object such as a String, or a complex object that you define)

- You should now see something similar to the following:

```
2007-08-23 15:32:52.875 Oracle Coherence GE 3.3/387 <Info> (thread=main,
member=
1): Loaded cache configuration from resource
"jar:file:/D:/OracleCoherenceLabs/c
oherence/lib/coherence.jar!/coherence-cache-config.xml"
2007-08-23 15:32:53.062 Oracle Coherence GE 3.3/387 <D5>
(thread=DistributedCach
e, member=1): Service DistributedCache joined the cluster with senior
service member 1
<distributed-scheme>
  <scheme-name>example-distributed</scheme-name>
  <service-name>DistributedCache</service-name>
  <backing-map-scheme>
    <local-scheme>
      <scheme-ref>example-backing-map</scheme-ref>
    </local-scheme>
  </backing-map-scheme>
  <autostart>true</autostart>
</distributed-scheme>

Map (test):
```

- The above text is showing that using the default config files within the supplied jar files, a named cache called *test* was created using the distributed scheme.

- Type in the command *help* to see the list of commands available. The following commands are the ones you will use most:

    - put – puts a name value pair (key and value) into the cache. Always returns the last value that the key had; (Examples: *put 1 value1 put 2 value2)*

    - get – retrieves the value for the given key from the named cache; (e.g. *get 2)*

    - list – shows the contents of the named cache;

    - remove – removes a key value from the cache

- Use the commands above and put and get some values.

- Start a second instance of coherence.cmd. What happens when you start this up?

- Use the command *cache test* in the second window to connect to the named cache called test. Notice the following message which shows the distribution of data taking place.

```
2007-08-23 16:02:44.453 Oracle Coherence GE 3.3/387 <D4>
(thread=DistributedCache, member=2): Asking member 1 for 128 out of 128
primary partitions
```

- Try to get and put values in different sessions. What happens?

- Close down one of the coherence.cmd consoles and note the message on the other console, indicating the backups of partitions have been restored.

```
2007-08-23 16:05:53.437 Oracle Coherence GE 3.3/387 <Info>
(thread=DistributedCache, member=1): Restored from backup 128 partitions
2007-08-23 16:05:53.437 Oracle Coherence GE 3.3/387 <D4>
(thread=DistributedCache, member=1): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11,
12, 13, 14, 15, 16, 17, 18,
19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36,
37, 38,
39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56,
57, 58,
59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76,
77, 78,
79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96,
97, 98,
99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113,
114, 1
15, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127,
```

- Shut both coherence.cmd consoles down and the cache-server.cmd.

- In explorer change to the [LABS_HOME]\coherence\bin directory.

- In the file coherence.cmd (or coherence.sh), find the line with set storage_enabled and change the value **true** to **false.**

  set storage_enabled=**false**

- Start the coherence.cmd file again. Create a named cache called test using *cache test*. Try to put a value in as before. What happens and why?

```
Map (test): put name tim
2007-08-23 16:10:16.343 Oracle Coherence GE 3.3/387 <Error> (thread=main,
member
=1):
java.lang.RuntimeException: Storage is not configured
        at
com.tangosol.coherence.component.util.daemon.queueProcessor.service.D
istributedCache$BinaryMap.onMissingStorage(DistributedCache.CDB:9)
        at
com.tangosol.coherence.component.util.daemon.queueProcessor.service.D
istributedCache$BinaryMap.put(DistributedCache.CDB:16)
        at
com.tangosol.coherence.component.util.daemon.queueProcessor.service.D
istributedCache$BinaryMap.put(DistributedCache.CDB:1)
        at
com.tangosol.util.ConverterCollections$ConverterMap.put(ConverterColl
```

- Startup a cache server by running the cache-server.cmd file from the bin directory. Try the put again. What happened and why? Shutdown all cache-servers.

Lab 3 – Access the data grid from Java

The purpose of this lab is to start to get you familiar with using the Coherence Java API's. We will use JDeveloper to do the following:

- Create a new project;
- Create a new named cache;
- Put and get information to/from it;
- Retrieve information about the cache.

All Coherence caches are named, have a lifetime scoped by the cluster instance in which they exist and implement the NamedCache interface.

The NamedCache interface is an extension of java.util.Map and holds data/resources that are shared amongst the cluster members. Each NamedCache 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 & synchronization, storage integration, queries, events aggregations, and transactions.

The cache topology (or implementation) can be swapped/ changed out without code changes!

Some of the methods within the NamedCache Interface:

- `void clear()` – removes all entries from the named cache
- `boolean containsKey(Object key)` – returns true if the named cache contains a entry for the key
- `booelan containsValue(Object value)` – returns true if there is at least one entry with this value in the named cache
- `Object get(Object key)` – get the entry from the named cache for that key
- `Object put(Object key, Object value)` – put an object in the named cache. (Blocking call)
- `Object remove(Object key)`

- `Set entrySet()` – returns a set of key, value pairs
- `Collection values()` – gets all values back as a collection
- `CacheService getCacheService()`

For a full list of methods refer to the javadoc in [LABS_HOME]\coherence\doc.

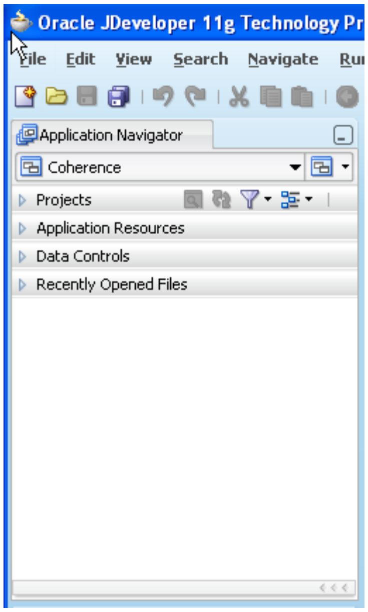
The CacheFactory is used mainly to obtain an instance of a named cache. The methods of interest here are:

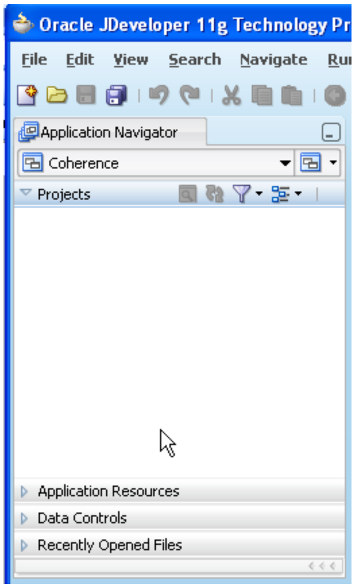
- `static Cluster ensureCluster()` – ensures you are in a cluster
- `static void shutdown()`
- `static NamedCache getCache(String cache)` – returns an instance of a cache. Either joins an existing cache in the cluster or creates the cache if this is the first member.

For more information on java.util.Map see http://java.sun.com/j2se/1.5.0/docs/api/java/util/Map.html

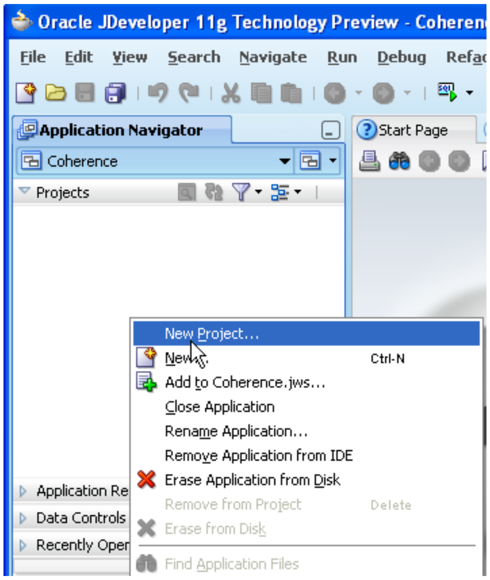
1. Create a new project in JDeveloper

    - If the Projects pane appears as below, click on Projects to expand so that it looks like the second image.



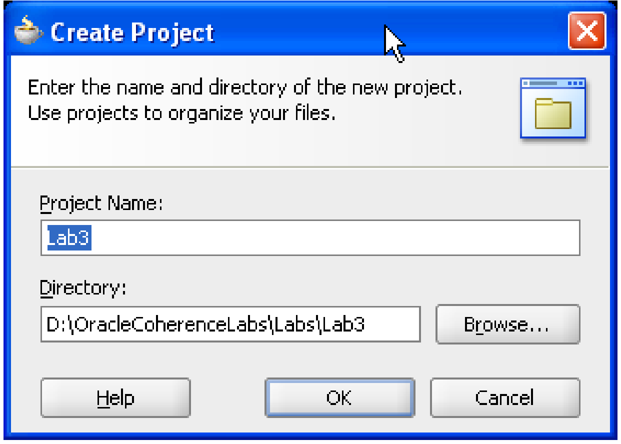


    - Right click in the projects area and click *New Project.*



    - Choose *Empty Project* and click *OK*.

- Name the project *Lab3* and click on *OK*.



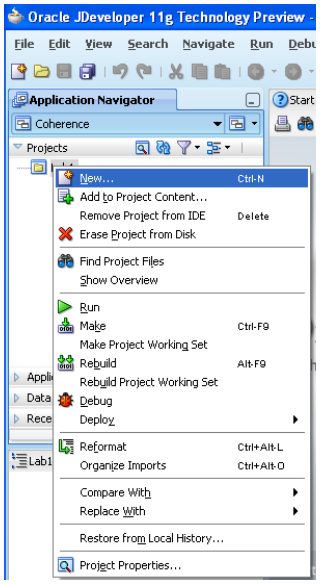
Enter the name and directory of the new project. Use projects to organize your files.

Project Name: Lab3

Directory: D:\OracleCoherenceLabs\Labs\Lab3

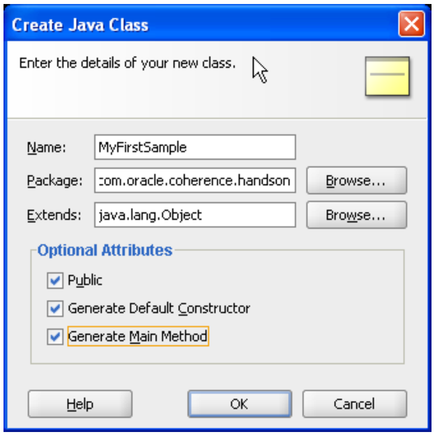
Create your first Coherence Java program

- Right click on the Lab3 project entry and choose *New*.



- Choose *General* category and choose *Java Class*. Click on *OK*.

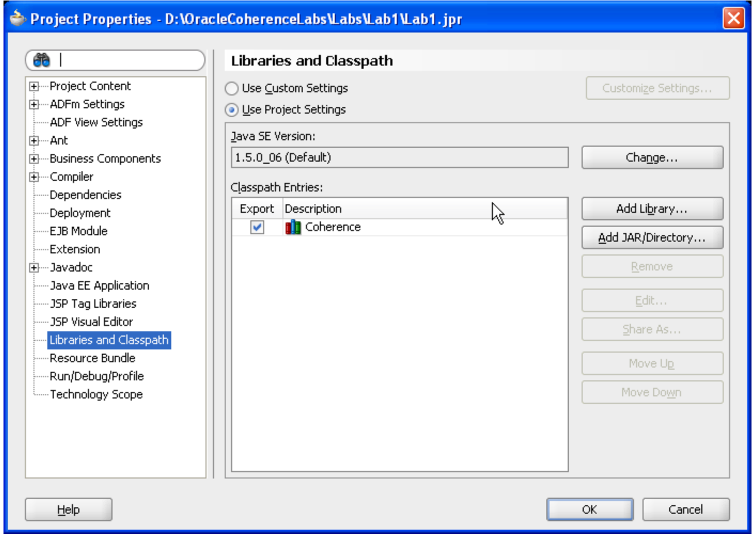
- Enter MyFirstSample as the class name and check *Generate Main Method*. You create dialog should look like this. Click on *OK*.



Name: MyFirstSample
Package: :om.oracle.coherence.handson
Extends: java.lang.Object

Optional Attributes
☑ Public
☑ Generate Default Constructor
☑ Generate Main Method

- If you right click on the Labs3 project and choose *Project Properties* and select *Libraries & Classpath*, you should see that the Coherence library we defined earlier is there by default.



Libraries and Classpath
○ Use Custom Settings
● Use Project Settings
Java SE Version: 1.5.0_06 (Default)
Classpath Entries:

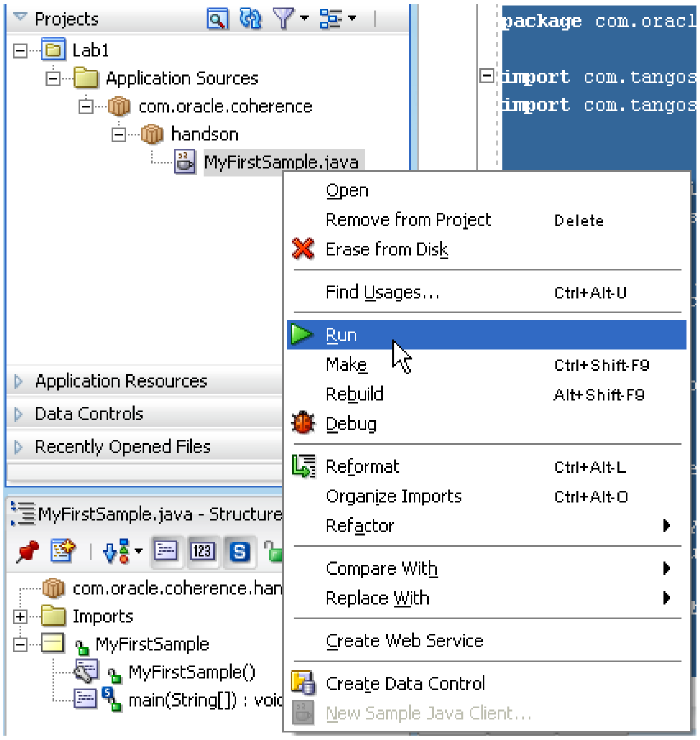
Export	Description
☑	Coherence

- Back in the editor for MyFirstSample you must import the following components in your Java class:
import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;

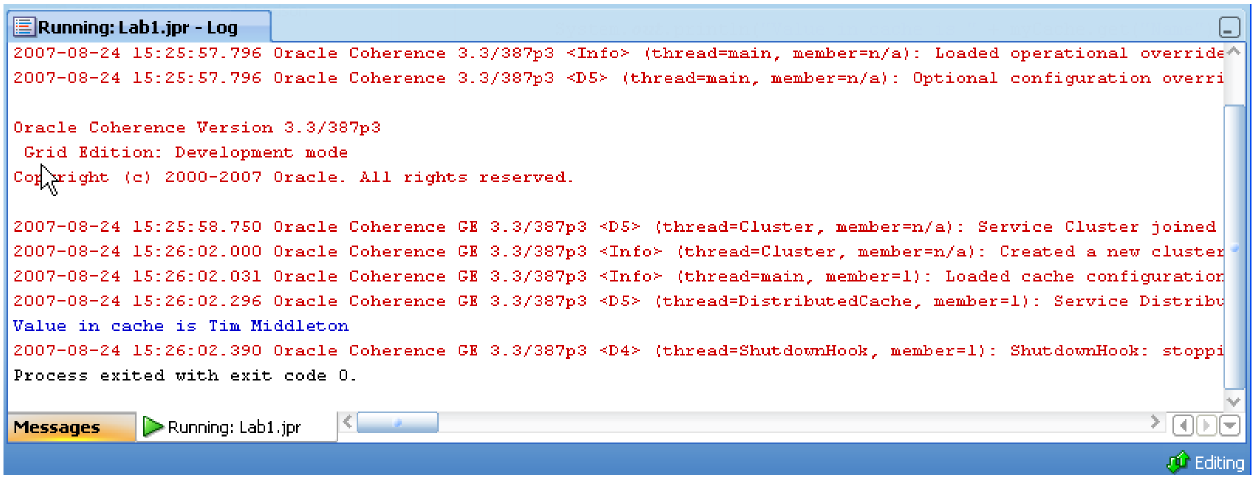
- Create the code to create a named cache, put a value in and then verify that value you put in. Use the following code fragment as a guide.
Example 1.

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

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

public class MyFirstSample {
    public MyFirstSample() {
    }

    public static void main(String[] args) {

       // ensure we are in a cluser
       CacheFactory.????

       // create or get a named cache called mycache
       NamedCache myCache = CacheFactory.??????

       // put key, value pair into the cache.
       myCache.put(????);

       System.out.println("Value in cache is " + ???? );
     }
}
```

- Once completed, save the file. To run this in JDeveloper, right click on the MyFirstSample.java class and choose *Run*.



- You should see messages similar to the following.



```
2007-08-24 15:25:57.796 Oracle Coherence 3.3/387p3 <Info> (thread=main, member=n/a): Loaded operational override
2007-08-24 15:25:57.796 Oracle Coherence 3.3/387p3 <D5> (thread=main, member=n/a): Optional configuration overri

Oracle Coherence Version 3.3/387p3
 Grid Edition: Development mode
Copyright (c) 2000-2007 Oracle. All rights reserved.

2007-08-24 15:25:58.750 Oracle Coherence GE 3.3/387p3 <D5> (thread=Cluster, member=n/a): Service Cluster joined
2007-08-24 15:26:02.000 Oracle Coherence GE 3.3/387p3 <Info> (thread=Cluster, member=n/a): Created a new cluster
2007-08-24 15:26:02.031 Oracle Coherence GE 3.3/387p3 <Info> (thread=main, member=1): Loaded cache configuration
2007-08-24 15:26:02.296 Oracle Coherence GE 3.3/387p3 <D5> (thread=DistributedCache, member=1): Service Distribu
Value in cache is Tim Middleton
2007-08-24 15:26:02.390 Oracle Coherence GE 3.3/387p3 <D4> (thread=ShutdownHook, member=1): ShutdownHook: stoppi
Process exited with exit code 0.
```

- There are a lot of messages above. We will sort that out soon.

- Now create another Java class to just get the value of the name from your cache, rather than doing a put then a get. Use the following code fragment as a guide.
  Hint: Right click on the ***Labs3*** project and choose N*ew*. Call this MyFirstSampleReader. Ensure that you have selected the option to create the main method.

  Example 2.

```
package com.oracle.coherence.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 cluser
        CacheFactory.????

        // create or get a named cache called mycache
        NamedCache myCache = CacheFactory.????

        System.out.println("Value in cache is " + ????);
    }
}
```

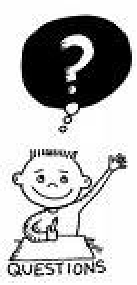
- Run the MyFirstSampleReader class. What happens? Why did you get a null value returned?

  Hint: When the MyFirstSample class ran, it is storage enabled by default unless

specified, so it created a named cache when it started; put the value in, and then when it completed the named cache disappeared.

- Startup the cache-server.cmd used in the previous exercise and then run MyFirstSample to put the value in the named cache and then MyFirstSampleReader to read the value from the cache.
  Note: You can also use the DefaultCacheServer run profile you created in the first lab.

- What happened and why?



By default, unless specified, all processes will startup as storage enabled. This means the process can store data as part of the cluster.

This should not be the case for a process that just joins the cluster to perform an operation such as put a value and then leave, like MyFirstSample.

What we need to do is to modify the process so that it will not be storage enabled. This is done by the following Java command line parameter.

`-Dtangosol.coherence.distributed.localstorage`

`-Dtangosol.coherence.distributed.localstorage=true` specifies that the process is storage enabled.

`-Dtangosol.coherence.distributed.localstorage=false` specifies that is it not storage enabled.

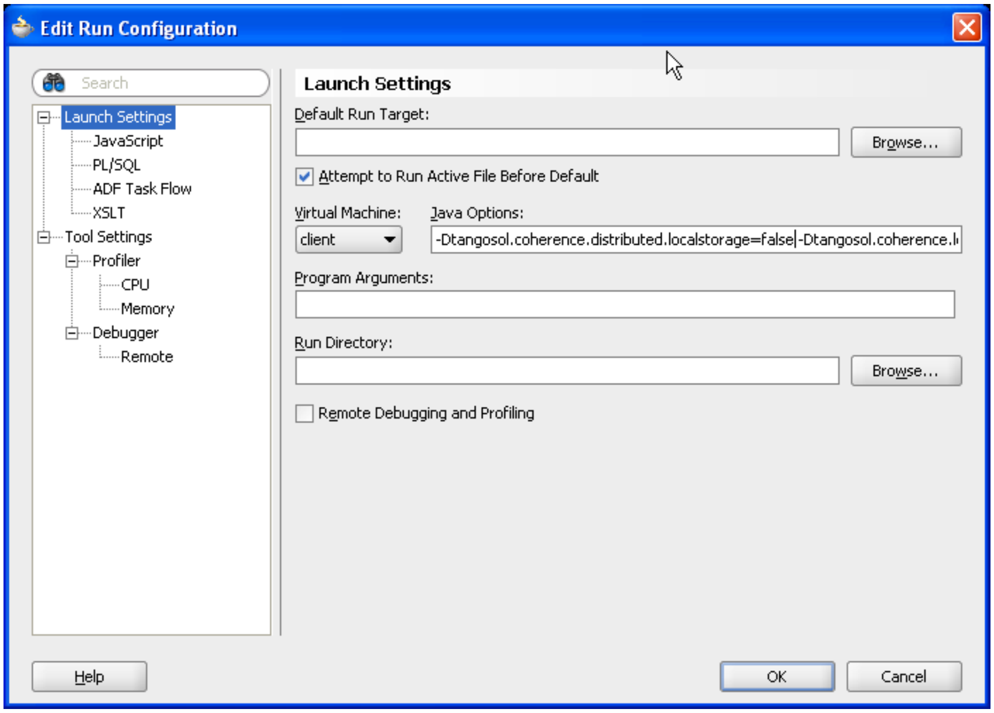
- Right click on the *Lab3* project and select *Project Properties*.

- Choose *Run/Debug Profile,* highlight *Default* and click on the *Edit* button.

- Enter the following value in the Java Options field:

```
-Dtangosol.coherence.distributed.localstorage=false
-Dtangosol.coherence.log.level=3
```

- The screen should look similar to the following:



Launch Settings

Default Run Target:

☑ Attempt to Run Active File Before Default

Virtual Machine: client

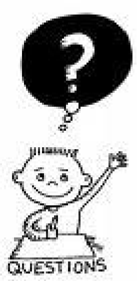
Java Options: -Dtangosol.coherence.distributed.localstorage=false -Dtangosol.coherence.l

Program Arguments:

Run Directory:

☐ Remote Debugging and Profiling

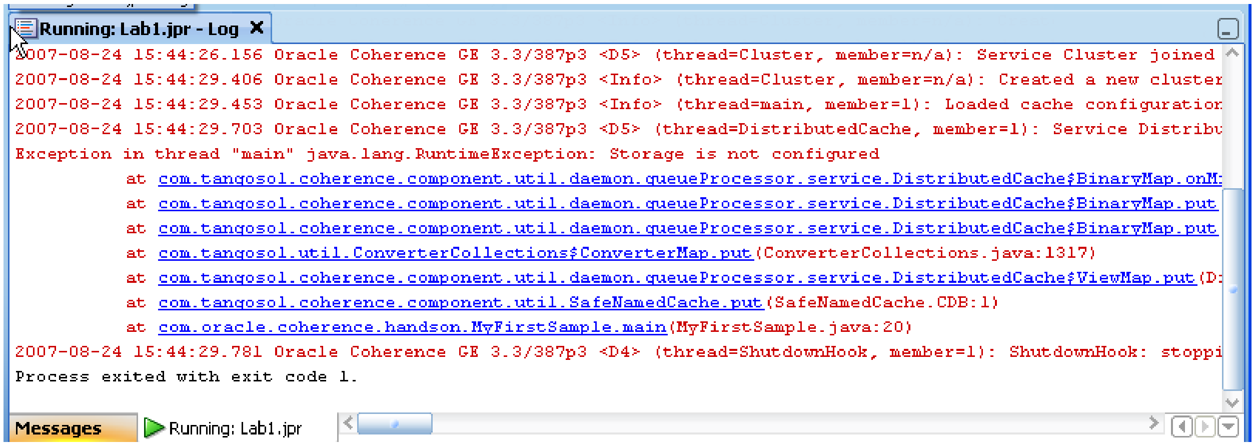
- Click on *Ok* and then *OK* to save.



Note:

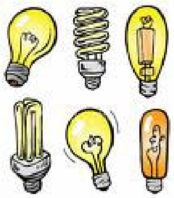
You could also create a default run profile called 'Storage Disabled' and set the values above. (See the next section for details of how to do this)

- Shutdown any running cache servers and re-run your class MyFirstSample. What happens?
- You will receive the following message indicating that there is no storage enabled on the cluster, as we have set this member to be storage disabled.



```
2007-08-24 15:44:26.156 Oracle Coherence GE 3.3/387p3 <D5> (thread=Cluster, member=n/a): Service Cluster joined
2007-08-24 15:44:29.406 Oracle Coherence GE 3.3/387p3 <Info> (thread=Cluster, member=n/a): Created a new cluster
2007-08-24 15:44:29.453 Oracle Coherence GE 3.3/387p3 <Info> (thread=main, member=1): Loaded cache configuration
2007-08-24 15:44:29.703 Oracle Coherence GE 3.3/387p3 <D5> (thread=DistributedCache, member=1): Service Distribu
Exception in thread "main" java.lang.RuntimeException: Storage is not configured
          at com.tangosol.coherence.component.util.daemon.queueProcessor.service.DistributedCache$BinaryMap.onM
          at com.tangosol.coherence.component.util.daemon.queueProcessor.service.DistributedCache$BinaryMap.put
          at com.tangosol.coherence.component.util.daemon.queueProcessor.service.DistributedCache$BinaryMap.put
          at com.tangosol.util.ConverterCollections$ConverterMap.put(ConverterCollections.java:1317)
          at com.tangosol.coherence.component.util.daemon.queueProcessor.service.DistributedCache$ViewMap.put(D
          at com.tangosol.coherence.component.util.SafeNamedCache.put(SafeNamedCache.CDB:1)
          at com.oracle.coherence.handson.MyFirstSample.main(MyFirstSample.java:20)
2007-08-24 15:44:29.781 Oracle Coherence GE 3.3/387p3 <D4> (thread=ShutdownHook, member=1): ShutdownHook: stoppi
Process exited with exit code 1.
```

- Start the cache-server again and re-test.
- You should now see that the data is persisted between running the two Java examples.



Bonus example.

Write some code to find the information about the local cluster member (the running Java process) and then list all other members in a cluster.

Hints:
The getCluster() method in the CacheFactory returns Cluster information.
The getMemberSet() for a Cluster returns a Set of Member objects for a cluster.

See the solutions for one way of achieving this.

Lab 4 - Leaving Strings behind - Working with complex Objects

The purpose of this lab is to work with complex objects within the cache. Within JDeveloper we will do the following.

- Create a new Person class which is serializable.
- Store and retrieve these Person objects in the cache;
- Serialize your own objects.

Up until now we have just been putting and getting String objects as the value in a named cache. Based upon the implementaion of the get mrthod we can see that the value (and the key) can be of type Object. i.e. any object,

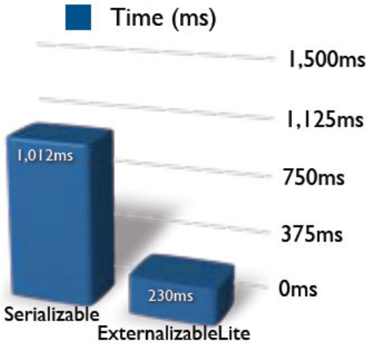
```
Object put(Object key, Object value)
```

This then allows us to store complex objects as the value in the cache. (Same as the java.util.Map interface.) Because Coherence may need to send the object across the wire, the object store must be Serializable. E.g. it implements java.io.Serializable.

Object serialization is the process of saving an object's state to a sequence of bytes, as well as the process of rebuilding those bytes into a live object at some future time.

Coherence supplies a class for doing high-performance serialization. This is com.tangosol.io.ExternalizableLite.

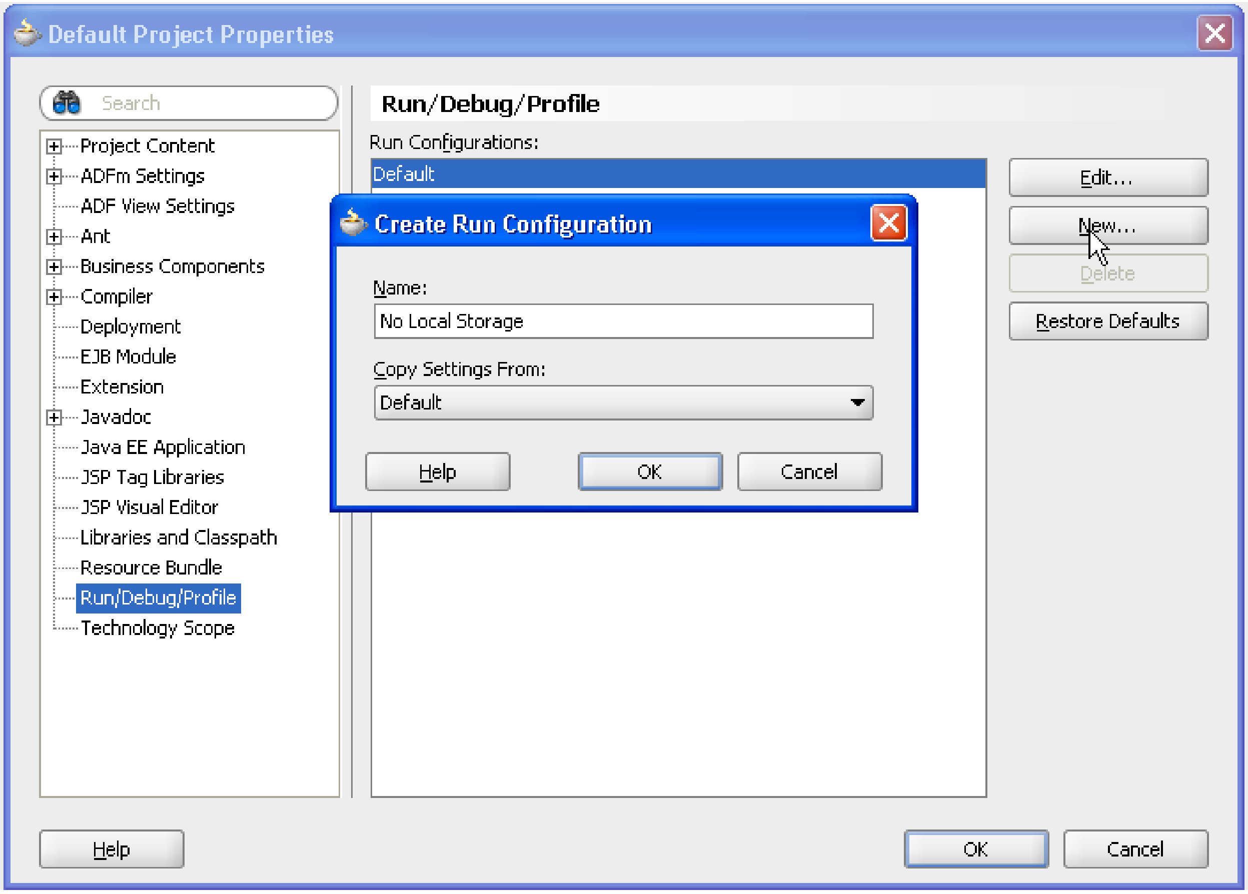
This is up to 6 times faster than standard Serializable as well as results in the serialization being smaller.



Add a Default Run Configuration to set the Java process as storage disabled, for this and future projects.

- Go to *Tools -> Default Project Properties*, on the Menu. Ensure that *Run/Debug/Profile* is selected from the properties list on the left

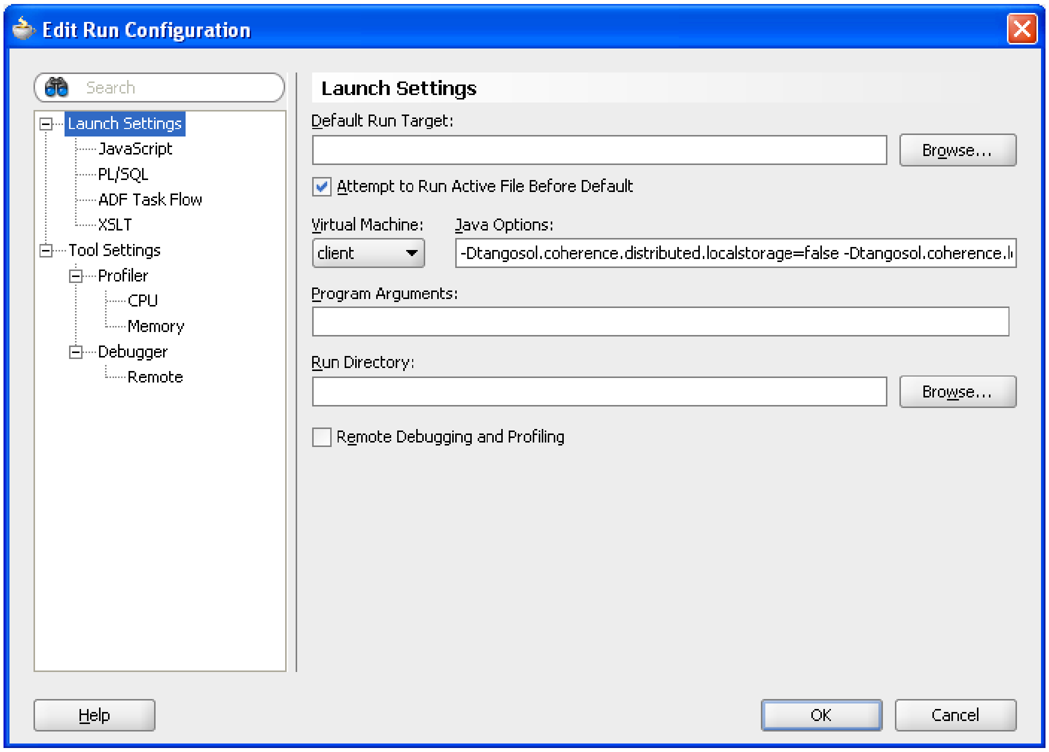
- Create a New Run Configuration called "*No Local Storage*". Click on *OK*.



Name: No Local Storage

Copy Settings From: Default

- Edit the configuration and enter as Java Options: -`Dtangosol.coherence.distributed.localstorage=false -Dtangosol.coherence.log.level=3`



Launch Settings

Default Run Target:

☑ Attempt to Run Active File Before Default

Virtual Machine: client

Java Options: -Dtangosol.coherence.distributed.localstorage=false -Dtangosol.coherence.l

Program Arguments:

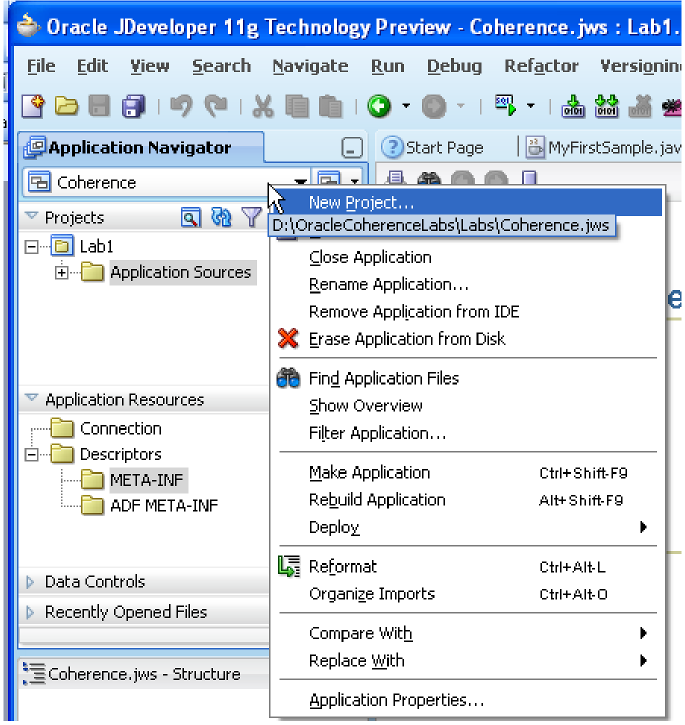
Run Directory:

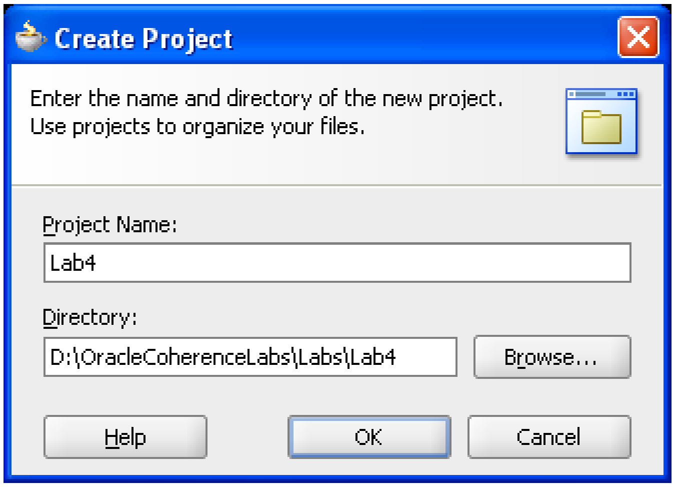
☐ Remote Debugging and Profiling

- Click on *OK* and then *OK* to save.

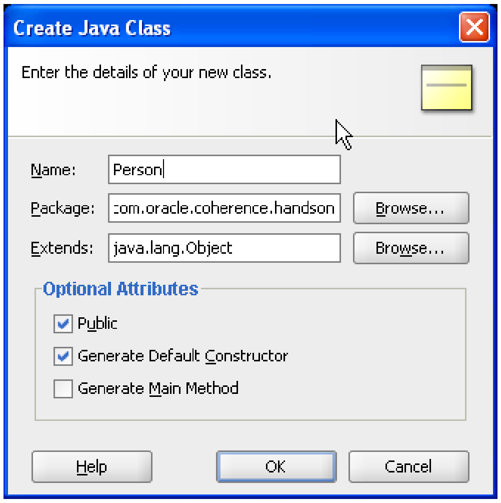
Create a Person class to store in the cache

- Create a new project in JDeveloper called *Lab4*. Hint: right click on C*oherence* application and choose *New Project*.




Create Project
Enter the name and directory of the new project.
Use projects to organize your files.
Project Name:
Lab4
Directory:
D:\OracleCoherenceLabs\Labs\Lab4
Browse...
Help
OK
Cancel

- Create a new Java class called *Person*. Do not select *Generate Main Method*.



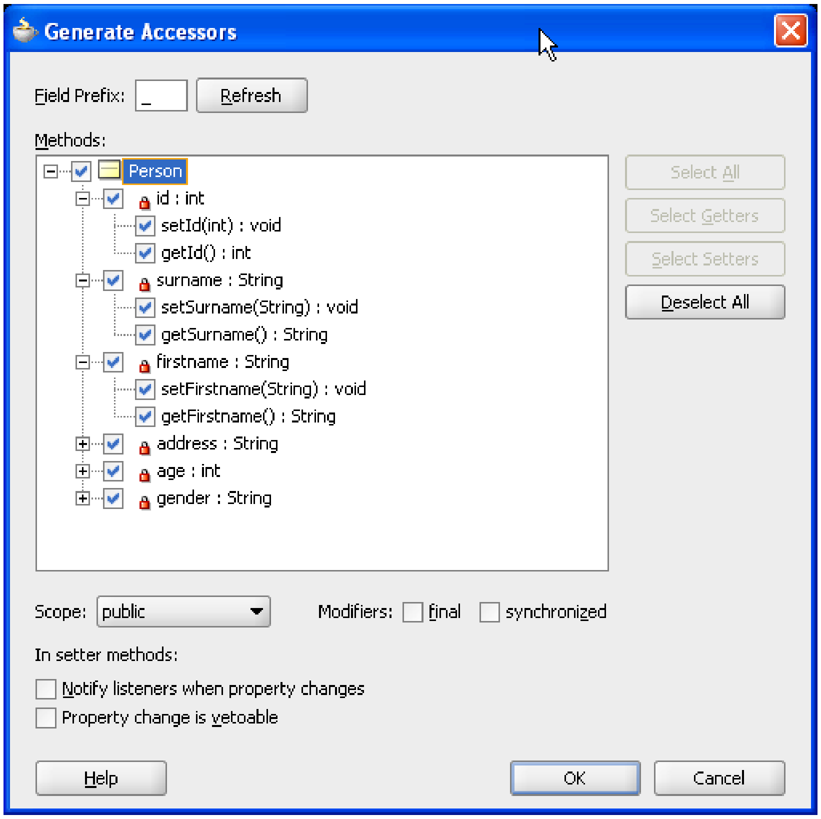
- Change your class the implement java.io.Serializable.
  Hint: use "implements java.io.Serializable"

- Enter the following private attributes for your *Person* class. (You can add others if you like)
  - int id
  - String surname
  - String firstname
  - String address
  - int age;

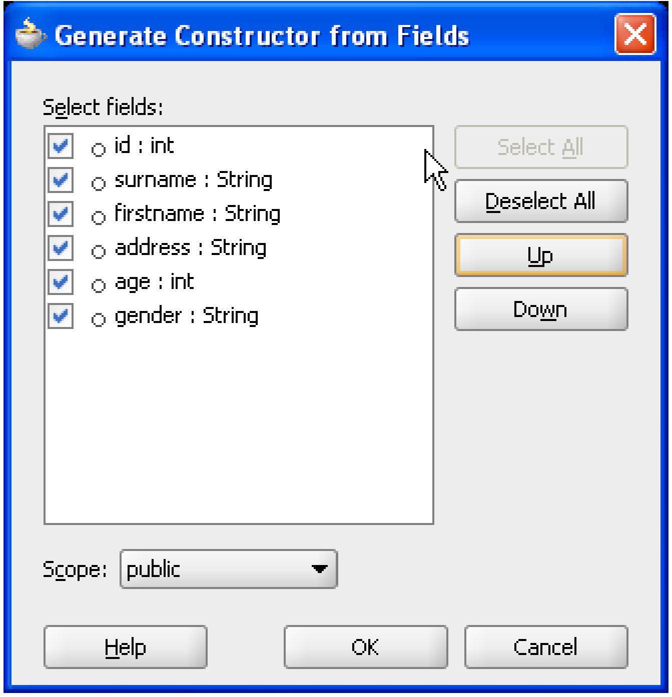
  - String gender

- JDeveloper can generate the default get/set methods for your attributes. From the *Source* menu, click on *Generate Accessors*.

- Select the check-box next to the Person class at the top. It will automatically select all the attributes. Click on *OK* to continue.



- You can also generate the default constructor and equals methods automatically. From the *source* menu, click *Generate Constructor from fields*, select all then *OK*.



- From the *Source* menu click on *Generate equals and hashcode*, and select all and press *OK*.

- Add 2 static variables under the gender variable, but above the public Person () method.
  - *public static String MALE = "M";*
  - *public static String FEMALE = "F";*

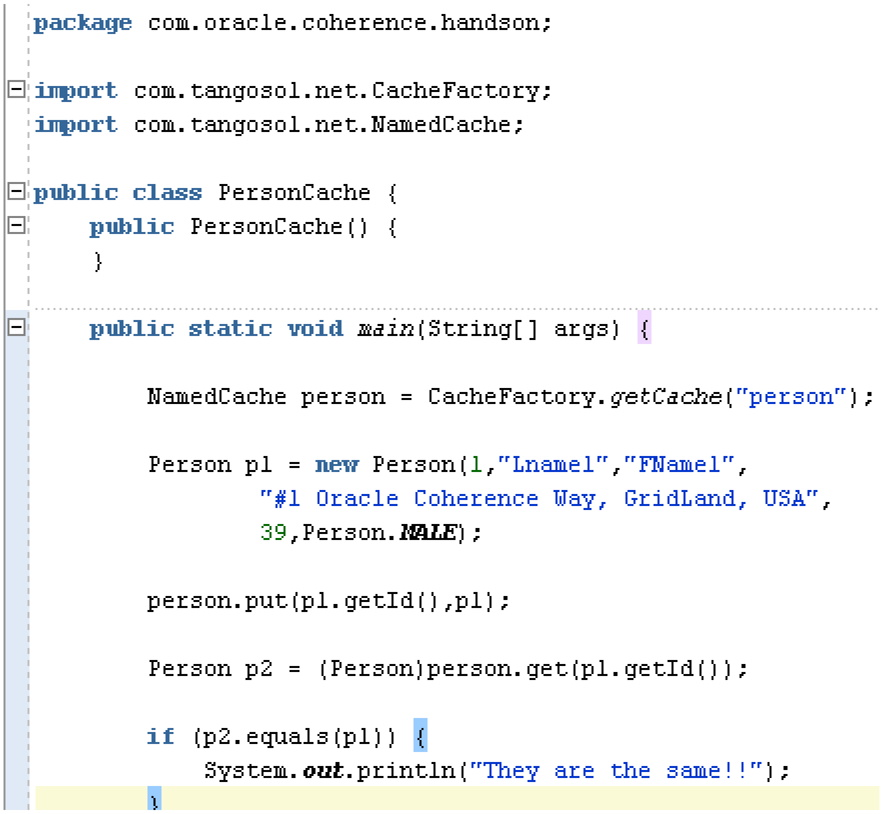
Create a driver class to put and get your Person class

- Create a new Java class called **PersonCache** in your lab4 project. Ensure it has a main method.

- Create a new named cache called person and put a new instance of the Person class in there. (see the lab solutions for full examples)

- Get the person from the cache and ensure that they are identical.

- Your code should look like the following:



```java
package com.oracle.coherence.handson;

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

public class PersonCache {
    public PersonCache() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");

        Person p1 = new Person(1,"Lname1","FName1",
                "#1 Oracle Coherence Way, GridLand, USA",
                39,Person.MALE);

        person.put(p1.getId(),p1);

        Person p2 = (Person)person.get(p1.getId());

        if (p2.equals(p1)) {
            System.out.println("They are the same!!");
        }
```

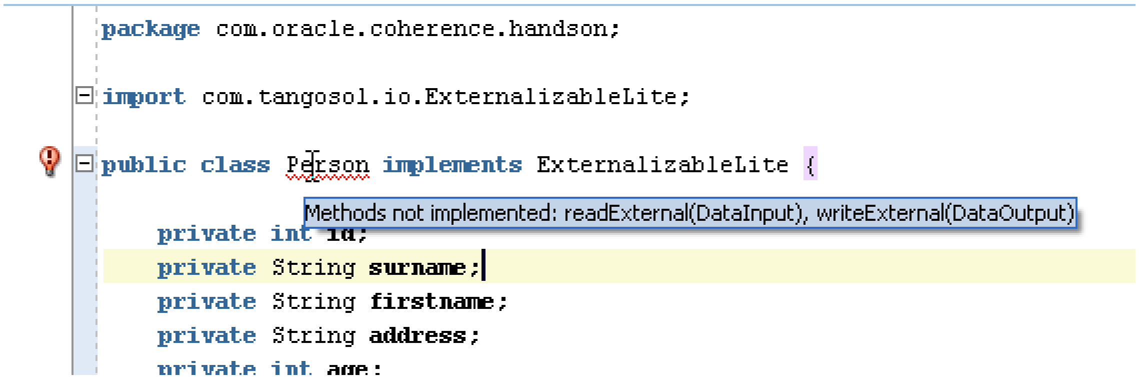
- Run Coherence Server within Jdeveloper via the DefaultCacheServer run profile

Implement faster serialization

- Highlight the Person.java tab. Go to ***File -> Save As*** in your menu, and save the file as PersonSerializable.java. Go to the new PersonSerializable.java tab, go to ***Search -> Replace*** from the menu, and replace globally all text from***"Person"*** to ***"PersonSerializable".*** *Save and close the file.*

- We will now change the Person class to implement the Coherence Externalizable Lite. Re-open the Person class in JDeveloper and change the implements ***Serializable*** to implements ***ExternalizableLite***;

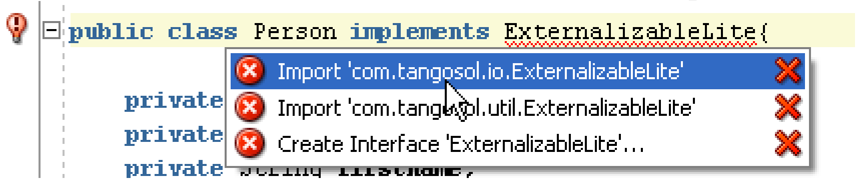
- You will notice that the Person class now appears with a read underline. If you hover the mouse over this, JDeveloper will tell you what the problem is.



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

import com.tangosol.io.ExternalizableLite;

public class Person implements ExternalizableLite {

    private int id;
    private String surname;
    private String firstname;
    private String address;
    private int age;
```

Methods not implemented: readExternal(DataInput), writeExternal(DataOutput)

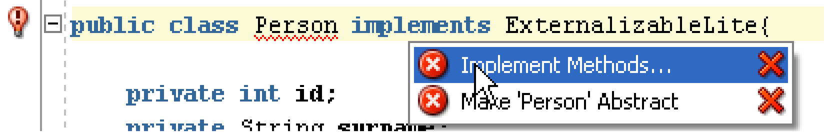
- Right click on *Person* and choose *Code Assist.* Select *Implement Methods.* The following window should be displayed. Choose to import `com.tangosol.io.ExternalizableLite`. Accept the defaults and click *OK.*



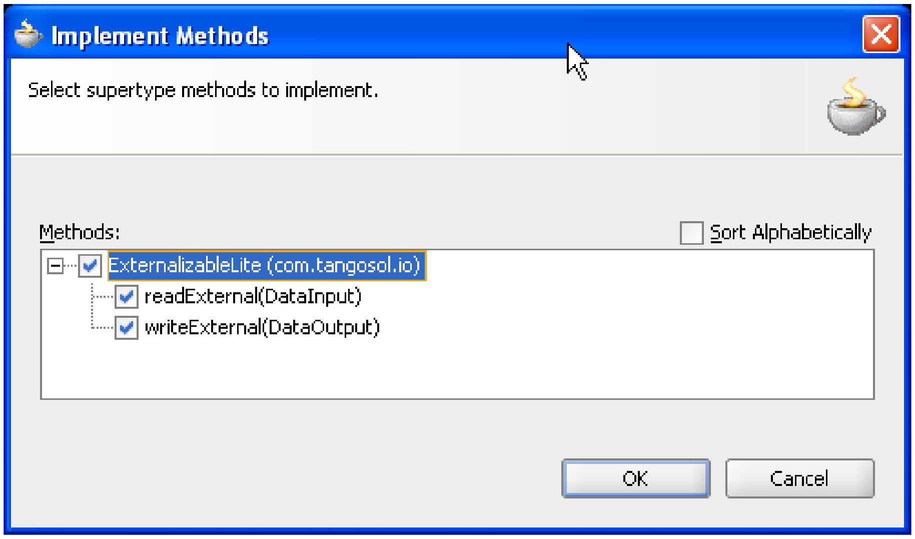
```
public class Person implements ExternalizableLite{

    private
    private
    private String firstname;
```

Import 'com.tangosol.io.ExternalizableLite'
Import 'com.tangosol.util.ExternalizableLite'
Create Interface 'ExternalizableLite'...

- Right click again, and choose to "Implement Methods…"



```
public class Person implements ExternalizableLite{

    private int id;
    private String surname;
```

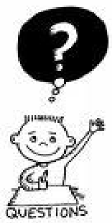

Implement Methods
Select supertype methods to implement.
Methods:
☐ Sort Alphabetically
☑ ExternalizableLite (com.tangosol.io)
☑ readExternal(DataInput)
☑ writeExternal(DataOutput)
OK
Cancel

- This will generate the readExternal and writeExternal methods required

- Make the following changes to the *Person* class. (You may need to import java.util.IOException & com.tangosol.util.ExternalizableHelper.)

```
public void readExternal(DataInput dataInput) throws IOException {
    this.id = ExternalizableHelper.readInt(dataInput);
    this.surname = ExternalizableHelper.readSafeUTF(dataInput);
    this.firstname = ExternalizableHelper.readSafeUTF(dataInput);
    this.address = ExternalizableHelper.readSafeUTF(dataInput);
    this.gender = ExternalizableHelper.readSafeUTF(dataInput);
    this.age = ExternalizableHelper.readInt(dataInput);
}

public void writeExternal(DataOutput dataOutput) throws IOException {
    ExternalizableHelper.writeInt(dataOutput, this.id);
    ExternalizableHelper.writeSafeUTF(dataOutput, this.surname);
    ExternalizableHelper.writeSafeUTF(dataOutput, this.firstname);
    ExternalizableHelper.writeSafeUTF(dataOutput, this.address);
    ExternalizableHelper.writeSafeUTF(dataOutput, this.gender);
    ExternalizableHelper.writeInt(dataOutput, this.age);
}
```

- Now re-test your code to ensure it works. (Remember to use run configurations “DefaultCacheServer” for DefaultCoherenceServer and “No Local Storage” for PersonCache)



What is overriding the readExternal and writeExternal achieving?

This implements the optimized Coherence serialization methods using the ExternaliableHelper class to read and write objects to and from the data output stream more efficiently.

Lab 5 – Querying & Aggregating Data in a cache

The purpose of this lab is to introduce the concept of querying and aggregating data in a cache. We will do the following:

- Create a java class to populate our cache with 10,000 Person objects;
- Query the cache for specific data; and
- Aggregate information within the cache and observe the changes in query times when we add addition cache members. (On dual core machines only)

Now that we are able to put complex objects in our named caches, we are going to look at querying and aggregating information within the grid.

The *com.tangosol.util.QueryMap* interface is used to for values or keys within a cache. You can use filters to restrict your results. You can also define indexes to optimize your queries.

Tip: Since Coherence serializes information when storing, you will have the overhead of de-serializing when querying. When indexes are added, the values identified in the index are not serialized and therefore offer quicker access time.

Some of the more useful methods in the QueryMap interface are:

- Set entrySet(Filter filter) – Returns a Set of entries that are contained in the map that satisfy the Filter;
- addIndex(ValueExtractor extractor, boolean fOrdered, Comparator comparator) – adds and index
- Set keySet(Filter filter) – Similar to entrySet but returns Keys, not values.

It is important to note that Filtering occurs at Cache Entry Owner .i.e.: In Partitioned Topology, Primary Partitions do the filtering and so can be done in parallel.

The QueryMap interface makes use of the Filter classes some of which are below.

AllFilter, AlwaysFilter, AndFilter, AnyFilter, ArrayFilter, BetweenFilter, ClassFilter, ComparisonFilter, ContainsAllFilter, ContainsAnyFilter, ContainsFilter, EqualsFilter, ExtractorFilter, GreaterEqualsFilter, GreaterFilter, InFilter, InKeySetFilter, IsNotNullFilter, IsNullFilter, KeyAssociatedFilter, KeyFilter, LessEqualsFilter, LessFilter, LikeFilter, LimitFilter, MapEventFilter, NeverFilter, NotEqualsFilter, NotFilter, NullFilter, OrFilter, PresentFilter, ValueChangeEventFilter, XorFilter..

A couple of examples will hopefully make things clearer!

Return a Set of people beginning with “Sm”:

```
Set macPeople = people.entrySet(
                 new LikeFilter("getLastName", “Sm”));
```

A set containing all of the open trades:

```
Set openTrades = trades.entrySet(
                   new EqualsFilter("isOpen", BOOLEAN.TRUE));
```

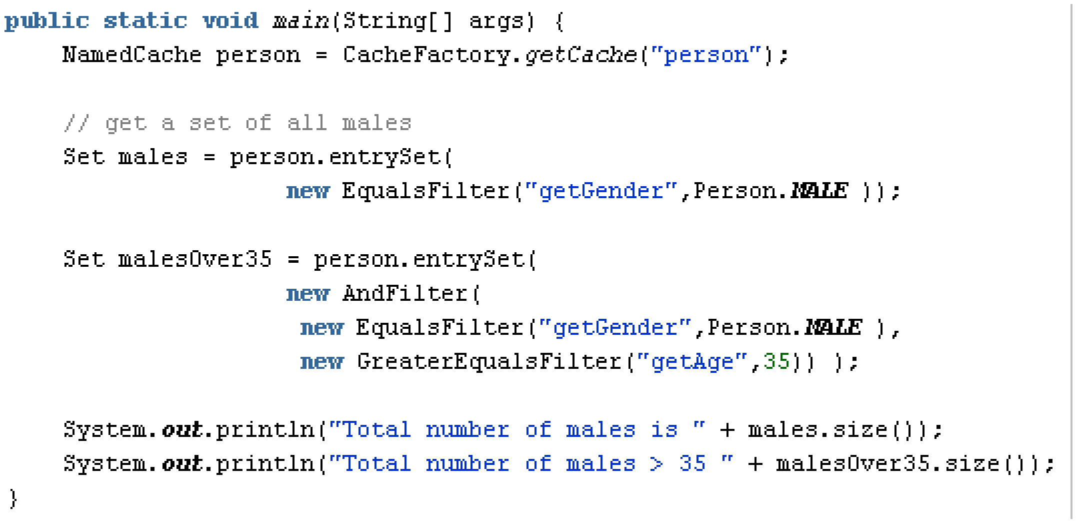
1. Create a Java class to populate the cache with 10,000 random *People* objects.

- Create a new project in JDeveloper and call it *Lab5*.
- We need to use the Person class we created in the last lab. Right click in *Lab5* and choose *Project Properties*. Click on *Libraries and Classpath* and then click on *Add Jar/Directory*.
- Select the following directory [LABS_HOME]\Labs\Lab4\classes.
- Create a new Java class called PopulatePeople. Ensure it has a main method.
- Create the code to connect to the cache and create 10,000 random Person objects, **Hint:** Use java.util.Random and to generate some random ages.
  Random generator = new Random();
  int age = generator.nextInt(100);
- Can you think of a more efficient way of doing this that 10,000 puts?
- **Hint:** see [LABS_HOME]\coherence\doc\user-guide.htm and search for “Bulk Loading and Processing with Coherence”

Create a class to perform your queries

- Create a new class called QueryExample with a main method;
- Use the entrySet method to get the number of males and the number of males 35 and over. Use the size() method to get the number of records returned. We will use the aggregation functions to do this a better way later.

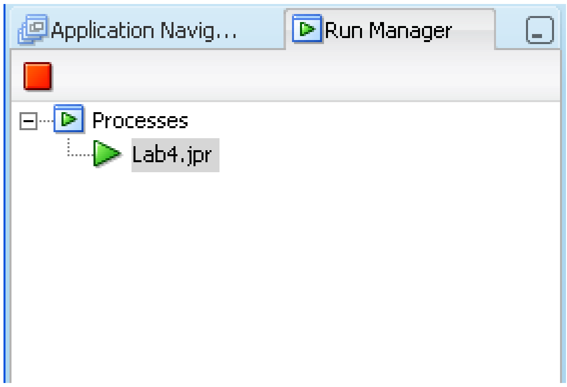
- Your code should look similar to the following:

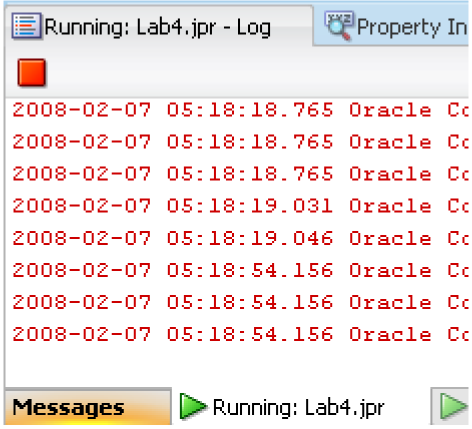


```
public static void main(String[] args) {
    NamedCache person = CacheFactory.getCache("person");

    // get a set of all males
    Set males = person.entrySet(
                    new EqualsFilter("getGender",Person.MALE ));

    Set malesOver35 = person.entrySet(
                    new AndFilter(
                     new EqualsFilter("getGender",Person.MALE ),
                     new GreaterEqualsFilter("getAge",35)) );

    System.out.println("Total number of males is " + males.size());
    System.out.println("Total number of males > 35 " + malesOver35.size());
}
```

Run PopulatePerson and QueryExample and observe the behavior -

- Go to *Run Manager*, next to *Application Navigator* to your left, and ensure that your Coherence Server is still running from last lab. Alternatively, look at the logs at the bottom pane for the server. If it is running, you would see a red square.





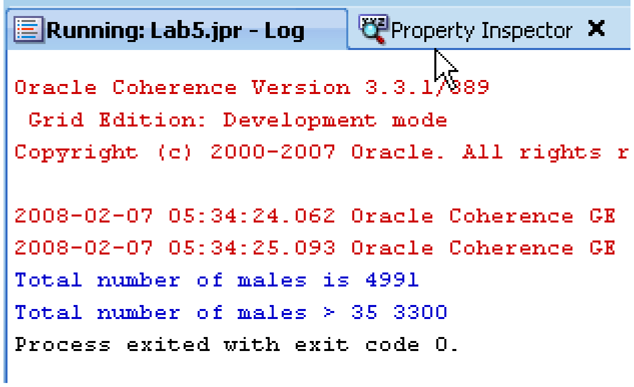
```
2008-02-07 05:18:18.765 Oracle Co
2008-02-07 05:18:18.765 Oracle Co
2008-02-07 05:18:18.765 Oracle Co
2008-02-07 05:18:19.031 Oracle Co
2008-02-07 05:18:19.046 Oracle Co
2008-02-07 05:18:54.156 Oracle Co
2008-02-07 05:18:54.156 Oracle Co
2008-02-07 05:18:54.156 Oracle Co
```

- If not, choose ▶ ▾ from the tool bar, and select the “DefaultCacheServer” run configuration.

- After the server has started, go to PopulatePeople.java, choose ▶ ▾ from the tool bar, and select the “No Local Storage” run configuration.

- Go to QueryExample.java, choose ▶ ▾ from the tool bar, and select the “No Local Storage” run configuration.

- You should receive output similar to the following:



```
Oracle Coherence Version 3.3.1/389
 Grid Edition: Development mode
Copyright (c) 2000-2007 Oracle. All rights r

2008-02-07 05:34:24.062 Oracle Coherence GE
2008-02-07 05:34:25.093 Oracle Coherence GE
Total number of males is 4991
Total number of males > 35 3300
Process exited with exit code 0.
```

Create a class to carry out aggregations on the data in the cache

An EntryAggregator allows you to carry out operations on all or a specific set of objects and get an aggregation as a result. They are effectively agents that execute services in parallel against the data within the cluster.

The aggregations are done in parallel and can benefit greatly from the addition of cluster members.

There are two ways of aggregating. Aggregate over a collection of keys or via specifying a filter.

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

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

We will use the Filter in our example below.

- Create a new class in Lab5 project called **AggregationExample**. Ensure it has a main method.
- Write the code to get the following from the cache
  - average age of all males
  - average age of all females;
  - average age;
  - Maximum age

- **Hint 1:** The following code will get the average age for all males.

```
        Double averageAgeMales = (Double)person.aggregate(

                          new EqualsFilter("getGender",Person.MALE ),

                          new DoubleAverage("getAge")

                          );
```

- **Hint 2:** to query all objects in a named cache you can either use (Filter)null or a new AlwaysFilter()

Observe what happens when we add new cache members.

- Modify the AggregationExample to put in timing information. Run the queries in a loop for 100 times and get an average time. This will ensure that gc's or normal machine spikes will not influence the results.

- **Hint:** you can get the current time using the following code:

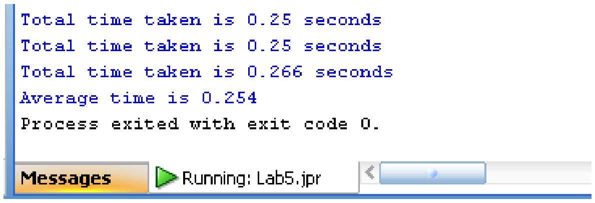
  long timeStart = System.currentTimeMillis();
  You can also use System.nanotime() which is a new feature introduced in Java5 to give sub-millisecond measurements.

- Run your code with 1 cache server, note the average time. Start a second cache server up, allow for the data to be re-distributed and re-run the code. What happens?

  Note: If you do not have a dual core machine you may not see an improvement in aggregation times. Why do you think this is the case?

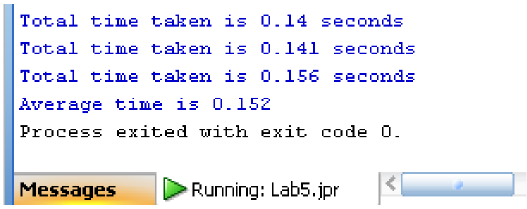
- Some sample timings are below

1 cache server.



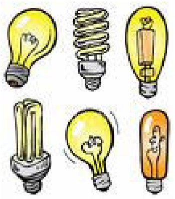
```
Total time taken is 0.25 seconds
Total time taken is 0.25 seconds
Total time taken is 0.266 seconds
Average time is 0.254
Process exited with exit code 0.
```

2 cache servers:



```
Total time taken is 0.14 seconds
Total time taken is 0.141 seconds
Total time taken is 0.156 seconds
Average time is 0.152
Process exited with exit code 0.
```

- What happens if you go to 3 cache servers?



Bonus example.

Add indexes to the *PopulatePeople* class to improve performance.

Hints:
Look up the *QueryMap* interface in the JavaDoc and find the *addIndex* method.

Lab 6 - Observing Data Changes

The purpose of this lab is to introduce the concept of observing data changes within a named cache. You will do the following in JDeveloper:

- Create a Java class that will respond to changes within cache entries in a named cache;
- Create a simple chat program.

The ObservableMap interface allows you observe and take action on changes made to cache entries. It extends the java.util.EventListener and uses the standard bean event model within Java.

Every type of named caches implements this interface. To listen for an event you register a MapListener on the cache.

There are three main ways to listen for events:

- listen for all events;
- listen for all events that satisfy a Filter; or
- listen for events on a particular object key.

The following methods (which implement the above) are available against a named cache.:

```
void addMapListener(MapListener listener)

void addMapListener(MapListener listener, Filter filter, boolean fLite)

void addMapListener(MapListener listener, Object oKey, boolean fLite)
```

The 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 may not be present.
An example of how to register this against a named cache is below. This has been done as an anonymous class.

```
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
    }
});
```

You can use the getOldValue() or getNewValue() methods in the mapEvent class above to get the entry for which the entry fired.

See
http://wiki.tangosol.com/display/COH33UG/Deliver+events+for+changes+as+they+occur

for more details.

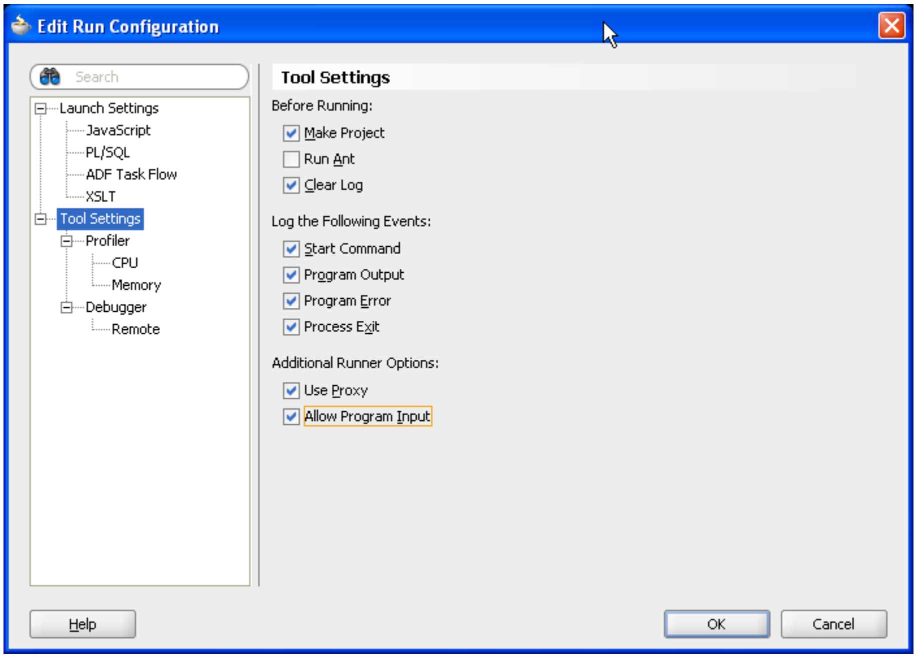
1. Create a new class that listens for a new Person object entry.

- Create a new project called *Lab6*. Ensure that the run confirguration “No Local Storage” was also created by default.
- We need to use the Person class we created in *Lab 4*. Right click in *Lab5* and choose *Project Properties*. Click on *Libraries and Classpath* and then click on *Add Jar/Directory*.
- Select the following directory [LABS_HOME]\Labs\Lab4\classes.
- Create a new Java class called *ListenForNewPerson.* Ensure that it has a main method.
- Within this class, add a listener to print out a message whenever a new Person is added to the cache.
- **Hint:** Use the following code to keep the Java process running until you read from the console, otherwise your program will exit immediately.

```
BufferedReader console = new BufferedReader(new InputStreamReader(System.in));

String text = console.readLine();
```

- To enable the console input you must do the following:
  - Right click on the Lab6 project and choose *Project Properties*.
  - Select *Run/Debug/Profile* on the left
  - Choose the “No Local Storage” run configuration, click on the *Edit* button on the right and then click *Tool Settings*. Ensure the *Allow Program Input* check-box is

checked.



Tool Settings

Before Running:

☑ Make Project
☐ Run Ant
☑ Clear Log

Log the Following Events:

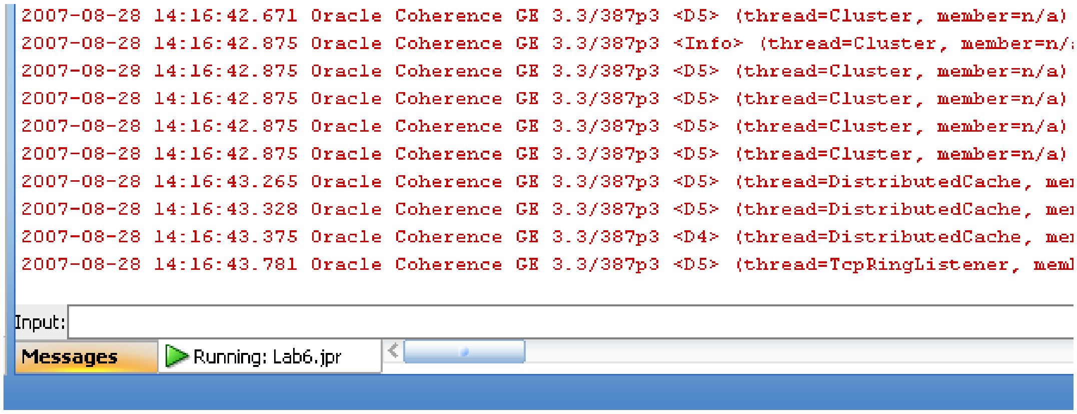
☑ Start Command
☑ Program Output
☑ Program Error
☑ Process Exit

Additional Runner Options:

☑ Use Proxy
☑ Allow Program Input

- Click on *OK* twice to save.

- When you run your Java class with "No Local Storage" run configuration, you should see the Input area down the bottom of the messages window. This is where you can input information from the console.



```
2007-08-28 14:16:42.671 Oracle Coherence GE 3.3/387p3 <D5> (thread=Cluster, member=n/a)
2007-08-28 14:16:42.875 Oracle Coherence GE 3.3/387p3 <Info> (thread=Cluster, member=n/
2007-08-28 14:16:42.875 Oracle Coherence GE 3.3/387p3 <D5> (thread=Cluster, member=n/a)
2007-08-28 14:16:42.875 Oracle Coherence GE 3.3/387p3 <D5> (thread=Cluster, member=n/a)
2007-08-28 14:16:42.875 Oracle Coherence GE 3.3/387p3 <D5> (thread=Cluster, member=n/a)
2007-08-28 14:16:42.875 Oracle Coherence GE 3.3/387p3 <D5> (thread=Cluster, member=n/a)
2007-08-28 14:16:43.265 Oracle Coherence GE 3.3/387p3 <D5> (thread=DistributedCache, me
2007-08-28 14:16:43.328 Oracle Coherence GE 3.3/387p3 <D5> (thread=DistributedCache, me
2007-08-28 14:16:43.375 Oracle Coherence GE 3.3/387p3 <D4> (thread=DistributedCache, me
2007-08-28 14:16:43.781 Oracle Coherence GE 3.3/387p3 <D5> (thread=TcpRingListener, mem
```

Create a class that will add and remove entries in a cache

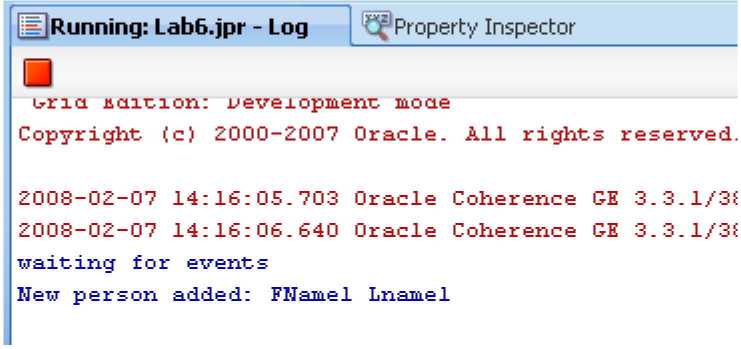
- Create a class called *PersonEventTester* that will put a new Person object in the cache. You can use similar code from PersonCache.java in *Lab4*

- Once this is complete, do the following:

  - Restart your cache server, with "DefaultCacheServer" run configuration.

  - Run your *ListenForNewPerson* class, with "No Local Storage" run configuration.

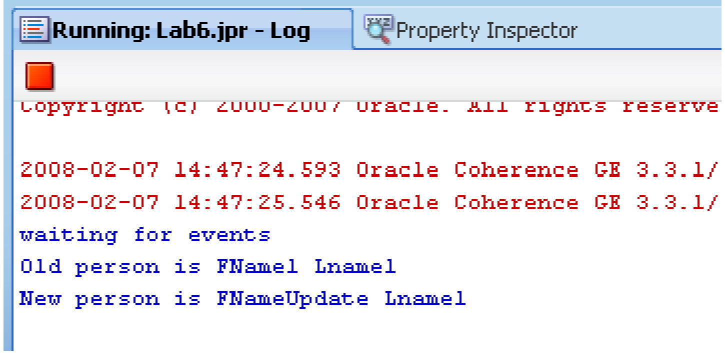
- Run the *PersonEventTester* to create a new record in the cache. What happens?
- You should see a message in the windows of the *ListenForNewPerson* messages window indicating a new record has been added.



```
Grid Edition: Development mode
Copyright (c) 2000-2007 Oracle. All rights reserved.

2008-02-07 14:16:05.703 Oracle Coherence GE 3.3.1/3
2008-02-07 14:16:06.640 Oracle Coherence GE 3.3.1/3
waiting for events
New person added: FName1 Lname1
```

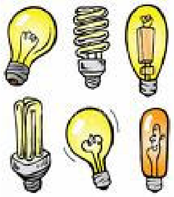
Create a class that will listen for an update to the Person object

- Update the class *PersonEventTester* to update the Person object in the cache
- Create a new class called *ListenForUpdatedPerson*. Ensure it has a main method.
- In the class, add a map listener to the person cache that will print out the new and old values of a male *Person* when that person object is updated.
- **Hint:** you need to supply two new parameters to the addListener method:
  - A new MapEventFilter();   and a parameter indicating if you want a lite event.
- Once this is complete, do the following:
  - Restart your cache server
  - Run the *ListenForNewPerson* class;
  - Run the *ListenForUpdatedPerson* class;
  - Run the *PersonEventTester* class;
- You should see the appropriate messages in the correct windows.



```
Copyright (c) 2000-2007 Oracle. All rights reserve

2008-02-07 14:47:24.593 Oracle Coherence GE 3.3.1/
2008-02-07 14:47:25.546 Oracle Coherence GE 3.3.1/
waiting for events
Old person is FName1 Lname1
New person is FNameUpdate Lname1
```

Bonus: Create your own scalable & resilient chat program



You should now have all the elements for a basic chat program.

Here are some hints on how to create this.

- Create a Message object to store the chat messages
- Create a ChatClient class that will do the following:
  - Get a persons name;
  - Setup a new MapListener to receive messages that are posted in the chat. Ensure that you don’t get messages posted by yourself; and
  - Loop and read a message from the command line and post this to the ‘messages’ named cache. (Exit when the user types exit.
- Run multiple copies of this and observe the behavior.
- Some extra features to add if you have time:
  - Add a facility to list all users in the chat. (You will need a second named cache)
  - Send a private message to a named individual.
  - Many more…

Lab 7 – In-Place processing of data

The purpose of this lab is to show how Entry Processors can be used to modify and carry out processing on entries. In JDeveloper we will do the following:

- Create an Employee class to hold our employees;
- Create a RaiseSalary class to be invoked on all entries;
- Create a InvokerTest class to insert employees and run the RaiseSalary on;
- Create a class that will show where data is held in a cluster;

Up to now, to perform actions on entries in a cache we have been using put and get operations. If we want to control concurrency to the data we have the option to lock and unlock keys.

Although we can do this, there is a better way to perform operations on data that ensures consistent behavior when concurrent data access is required.

Entry Processes are agents that perform processing against entries, and will carry this out directly where the data is being held. The sort of processing you can carry out may change the data, e.g. create, update or remove or may just perform calculations on the data.

Entry processors that work against the same key will be logically queued. This means that you can achieve lock-free (high performance) processing.

In Lab 5 we used and Entry Aggregator which is a type of Entry Processor, to aggregate data across the grid.

The InvocableMap interface (of which the NamedCache implements) has the following methods for operating on data.

- Object invoke(Object oKey, InvocableMap.EntryProcessor processor)
  Invoke the passed entry processor against an individual object, returns the result of the invocation.

- Map invokeAll(Collection keys,
  InvocableMap.EntryProcessor processor)
  Invoke the entry processor against the collection of keys and return the result for each for each invocation

- Map invokeAll(Filter filter,
  InvocableMap.EntryProcessor processor)
  Invoke the entry processor against the of entries that match the filter and

return the result for each for each invocation

To create an entry process you can extend com.tangosol.util.processes.AbstractProcessor and implement the process method.

E.g. We are creating an entry processor to increase the salary of all employees by 10%.

```
Class RaiseSalary extents AbstractProcessor {
    ...
   public Object process (Entry entry) {
     Employee emp = (Employee)entry.getValue();
     emp.setSalary(emp.getSalary() * 1.10);
     entry.setValue(emp);
     return null;
}
```

To invoke this you then do the following:

```
empCache.invokeAll(AlwaysFilter.INSTANCE, new RaiseSalary());
```

1. Create a new class to hold employees

   - Create a new project called *Lab7*. Ensure that the “No Local Storage” run configuration is also available for this project.
   - Create an *Employee* class (with no main method), with the following attributes:
     - private int empId;
     - private String surname;
     - private String firstname;
     - private double salary;
   - Ensure that you implement all of the methods required. (Use the *Source* menu to do this).
     - Generate Accessors…
     - Generate equals() and hashcode()…
     - Generate Constructor from Fields()…

Create a class to increase the salary of employees by 10%

- Create a class called *RaiseSalary* which extends AbstractProcessor

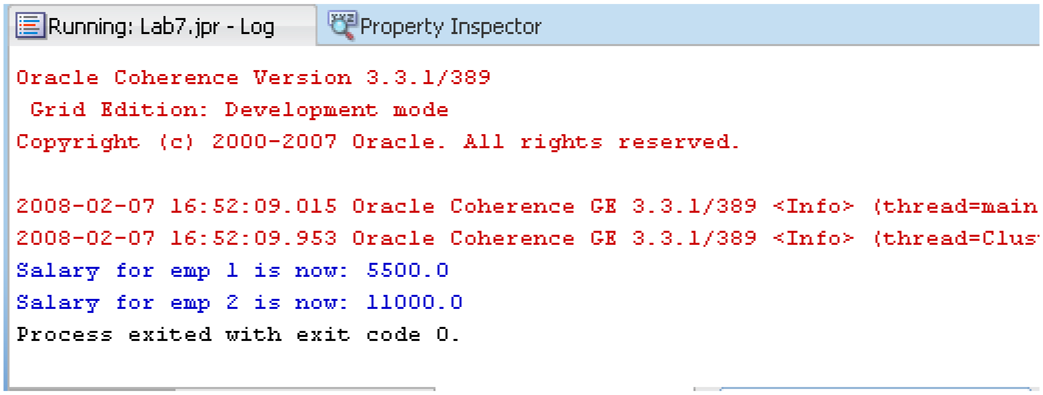
- Implement the process() method to raise the salary of an employee by 10%.

Create a class to test the above

- Create a class called *InvokeTest* (also Generate Main Method) that does the following:
    - Create a number of *Employee* objects and put them in the *employees* cache;
    - Invoke the *RaiseSlary* method on them all
    - Print out the new salaries to confirm the changes have been made.

Test Your Entry Processor *InvokeTest*

- Ensure that you Coherence Server is running. If not, run "**DefaultCacheServer**" under run profiles
- Right click on *Employee.java* and click on "Make" to compile this class.
- Right click on *RaiseSalary.java* and click on "Make" to compile this class.
- Click on *InvokeTest.java*, choose the "**No Local Storage**" run configuration to run this class.
- Your output should look similar to the following:



```
Oracle Coherence Version 3.3.1/389
 Grid Edition: Development mode
Copyright (c) 2000-2007 Oracle. All rights reserved.

2008-02-07 16:52:09.015 Oracle Coherence GE 3.3.1/389 <Info> (thread=main
2008-02-07 16:52:09.953 Oracle Coherence GE 3.3.1/389 <Info> (thread=Clus
Salary for emp 1 is now: 5500.0
Salary for emp 2 is now: 11000.0
Process exited with exit code 0.
```

Create a class to show where data is held within a cluster

- Create a class called *SayHelloProcessor* which extends AbstractProcessor
- Implement the process method to say hello.
- Create a new class called *WhereAreMyEmployees*. Ensure this class has a main method.
- Invoke the *SayHelloProcessor* on all entries in the cache.

- Add some more Employees and a second cache server. Run the *WhereAreMyEmployees* class again. Observe the messages on the cache server console.

Lab 8 – Extending your objects with Cache Keys & Data Affinity

The purpose of this lab is to show how you can extend objects to have composite keys, rather than a single value. In JDeveloper we will do the following:

- Create a new Person class an implement a Person.Key inner class;
- Use this new Person class to put and get values;
- Enable data affinity – E.g. keep related data together in the same partition.

In all the examples so far, we have been using single attributes for the key. E.g. person id or name, etc. If we want to store different versions of objects or have composite keys, there are a number of (bad) ways you "could" do this such as storing version as "Fred-1","Fred-2", or strings such as "Fred-Jones".

Using this method, to get a key for a particular individual would require some string mangling which will get messy and potentially give us sub-optimal performance.

Rather than using a single key, as above, you can create a class that encapsulates the business logic of that key and implement this as your key. This helps to future-proof your objects against changes. E.g. We can create an inner class that provides us a key.

Person.Key(int id, int version)

An interesting side-effect of this is that we can then make our key implement the KeyAssociation interface, and use the getAssociatedKey() method to provide us with partition affinity on an attribute of a key.

E.g. we could potentially store all of the people with the same surnames or potentially any common field within the same partition.

E.g. The following code is an example of defining an inner class called *Key* for the Person class based upon the person id and a version number.

```
public static class Key implements ExternalizableLite {

    // lets define a key of id and version
    private int id;
    private int version;

    public Key() {
            //for serializble
    }

    public Key(int id, int version) {
       this.id = id;
       this.version = version;
```

```
        }

        public Key(Person p) {
            this.id = p.getId();
            this.version = 1;  // default to version 1
        }

        public void writeExternal(DataOutput dataOutput) throws IOException {
            ExternalizableHelper.writeInt(dataOutput, this.id);
            ExternalizableHelper.writeInt(dataOutput, this.version);
        }

        public void readExternal(DataInput dataInput) throws IOException {
            this.id = ExternalizableHelper.readInt(dataInput);
            this.version = ExternalizableHelper.readInt(dataInput);
        }

        @Override
        public boolean equals(Object object) {
             ...
        }

        @Override
        public int hashCode() {
            final int PRIME = 37;
            int result = 1;
            return result;
        }
    }
```

To create and use this within Coherence:

```
NamedCache person = CacheFactory.getCache("person");

Person p = new Person(1,"Middleton","Tim","Address",29,Person.MALE);

person.put(p.getKey(), p);

Person p2 = (Person)person.get(new Person.Key(1,1));

System.out.println("Person is " + p2.getSurname());
```

We can then modify the inner class to implement KeyAssociation and then provide an implementation of the *getAssociatedKey* method. If the *getAssociatedKey()* returned surname, this would ensure all people with the same surname would reside in the same key partition and hence the same member.

1. Copy the existing Person class from lab 4.
    - Create a new project called *Lab8*.
    - Create a new class called *Person*.
    - Cut and paste the source from the *Person.java* file from lab 4 into this class

- Implement the Key class using the code from above.
- Ensure you implement the getKey method in the Person class.

```
public Key getKey() {

      return new Key(this);

}
```

- Compile your class.

Create a *KeyTester* class to test your new key.

- Create a new class called *KeyTester* in your project and implement the code above to create a new *Person* object and put and get this from the cache.

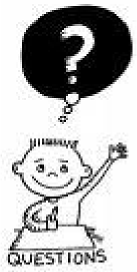
Implement KeyAssocation class for data affinity

- Modify your *Person.Key* class to implement the KeyAssocation interface. You will need to do a number of things:
    - Implement the getAssocatedKey method within the *Person.Key* class;
    - Add a new attribute to the *Key* class called surname;
    - Initialize surname in the constructor.
- Compile your new *Person* class
- You will need to restart your cache servers as the definition of the *Person* object has changed.

- Create a new class called *KeyTester2* with a main method and add some new *Person* objects.
  Hint: You could use the following code to all entries

```
        LinkedList peopleList = new LinkedList();

        // build the list
        peopleList.add( new Person(1,"Flinstone","Fred","Address",29,Person.MALE));
        peopleList.add( new Person(2,"Flinstone","Wilma","Address",29,Person.FEMALE));
        peopleList.add( new Person(3,"Rubble","Barney","Address",44,Person.MALE));
        peopleList.add( new Person(4,"Rubble","Betty","Address",44,Person.FEMALE));

        for (java.util.Iterator iterator = peopleList.iterator(); iterator.hasNext();) {
                Person p = (Person)iterator.next();
                person.put(p.getKey(), p);
        }
```

- Now that we have added the entries how do you think we can see if our data affinity has worked?



There are a number of ways of doing this.

1. We could use an EntryProcessor (as we did in Lab 7), to do a System.out.println on all the *Person* entries.

2. We can also ask the CacheService who owns the entry. The CacheService has a method called getKeyOwner() that will return the member that owns the key. Interesting to note that the entry does not actually have to exist. E.g. you can find out which member will own a key before you create it. You can use the following:

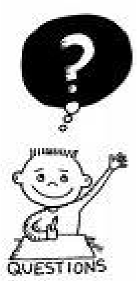
```
Member m =
((DistributedCacheService)namedCache.getCacheService()).getKeyOwner(pp.getKey(
));
```

- Add the following code to get the member that owns each of the keys. Once you get the member you can use various methods to get informaiton about the member.

- Run *KeyTester2* with one cache server running. You should get output similar to the following.

```
Person Fred Flinstone is owned by member 1
Person Wilma Flinstone is owned by member 1
Person Barney Rubble is owned by member 1
Person Betty Rubble is owned by member 1
```

- Startup a second cache server and re-run. What happens?

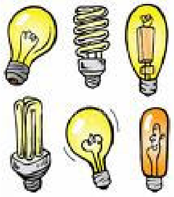


You should see that all people of the same surname will be owned by the same members.

If you still see all of the people in the same member, add another cache server (and optionally more person objects) and that should distribute the data, and you should see something like this.

```
Person Fred Flinstone is owned by member 3
Person Wilma Flinstone is owned by member 3
Person Barney Rubble is owned by member 4
Person Betty Rubble is owned by member 4
Person Dino Rubble is owned by member 4
```

Bonus – Use an EntryProcessor to display the location of the entries



Create a new EntryProcessor (ref to Lab 7), to display the location of each *Person* object in the cache.

Remember to add the classpath of you Person class to the cache-server.cmd file.

Lab 9 – Using JPA with Coherence

The purpose of this lab is to show you how to use JPA (Java Persistence API), to enable you to carry out object-relational mapping under the covers. In JDeveloper we will do the following:

- Install OracleXE database;
- Create a connection to the HR schema in XE;
- Automatically generate the JPA Objects for the EMPLOYEE table;
- Modify the cache-server.cmd to point to the sample JPA cache-config.xml

From the Sun site (http://java.sun.com/developer/technicalArticles/J2EE/jpa/)

“A major enhancement in EJB technology is the addition of the new Java Persistence API, which simplifies the entity persistence model and adds capabilities that were not in EJB 2.1 technology.

The Java Persistence API deals with the way 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 Java Persistence API 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 a number of backing map implementations.

The JPA implementation is the one we are going to use within this lab. This implementation provides O-R mapping from the Java world to the Database world, allowing us to use standard Coherence get/put and have this translated into database calls using the JPA API via TopLink.

1. Install OracleXE

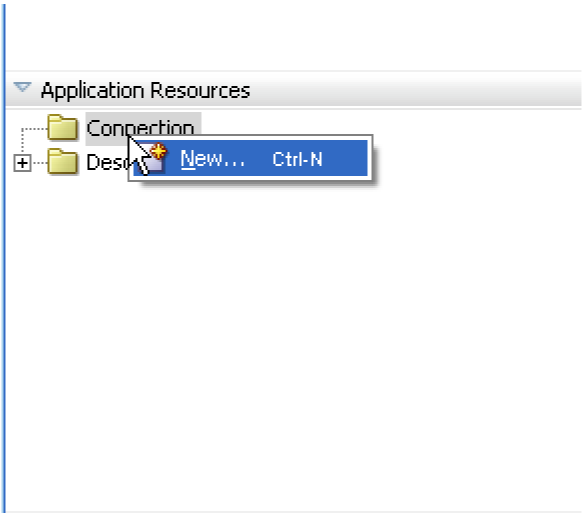
   - If you do not have an Oracle database that has the same HR schema, install OracleXE.

Create a new project called *Lab9*.

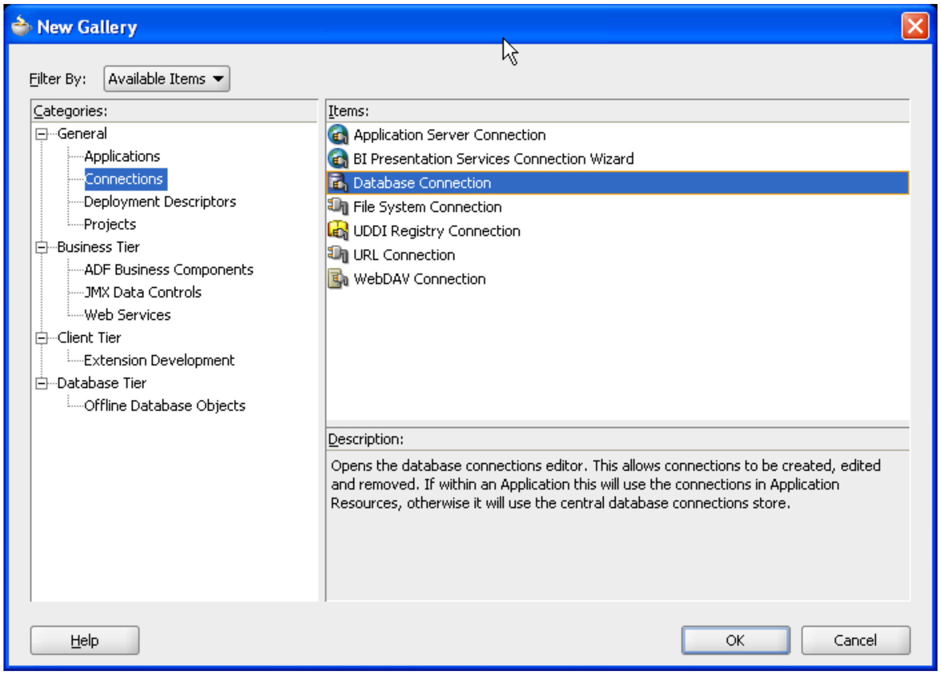
- Ensure you set the default java args to make your client not storage enabled.

Create a new database connection to the HR schema

- In the *Application Resources* section of the navigator, right-click on *Connection* and click *New*.

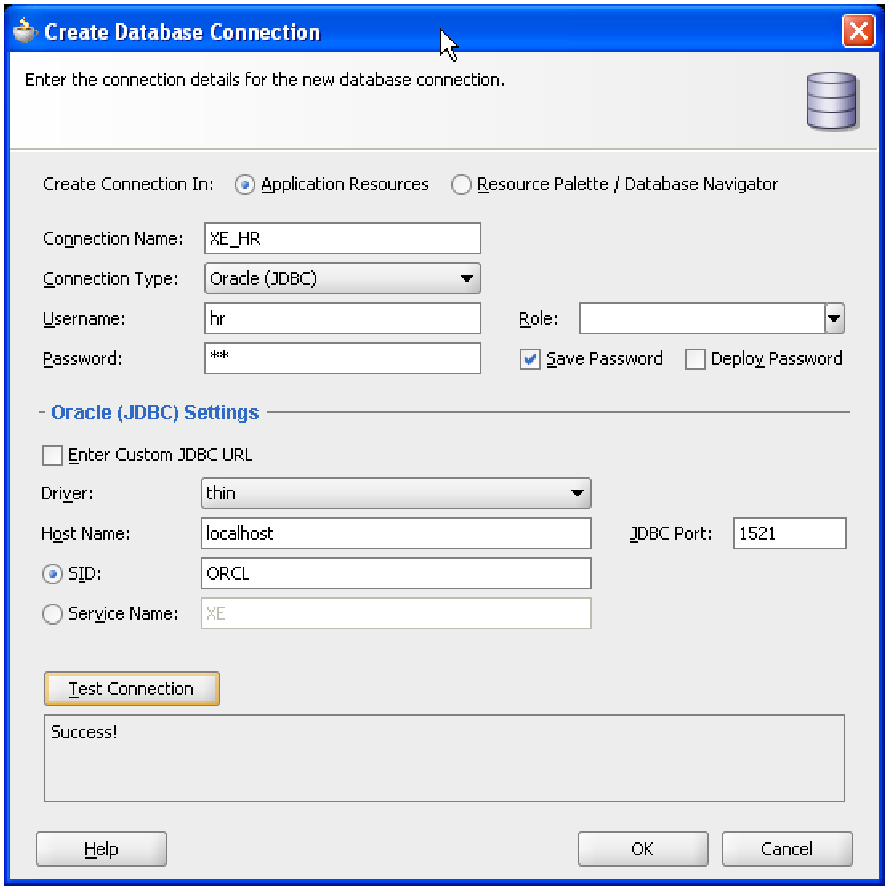


- Choose *Connections* and then *Database Connection*.



Opens the database connections editor. This allows connections to be created, edited and removed. If within an Application this will use the connections in Application Resources, otherwise it will use the central database connections store.

- Enter the details to connect to your HR schema and click *OK*.



Enter the connection details for the new database connection.

Create Connection In: ● Application Resources ○ Resource Palette / Database Navigator

Connection Name: XE_HR
Connection Type: Oracle (JDBC)
Username: hr
Role:
Password: **
☑ Save Password ☐ Deploy Password

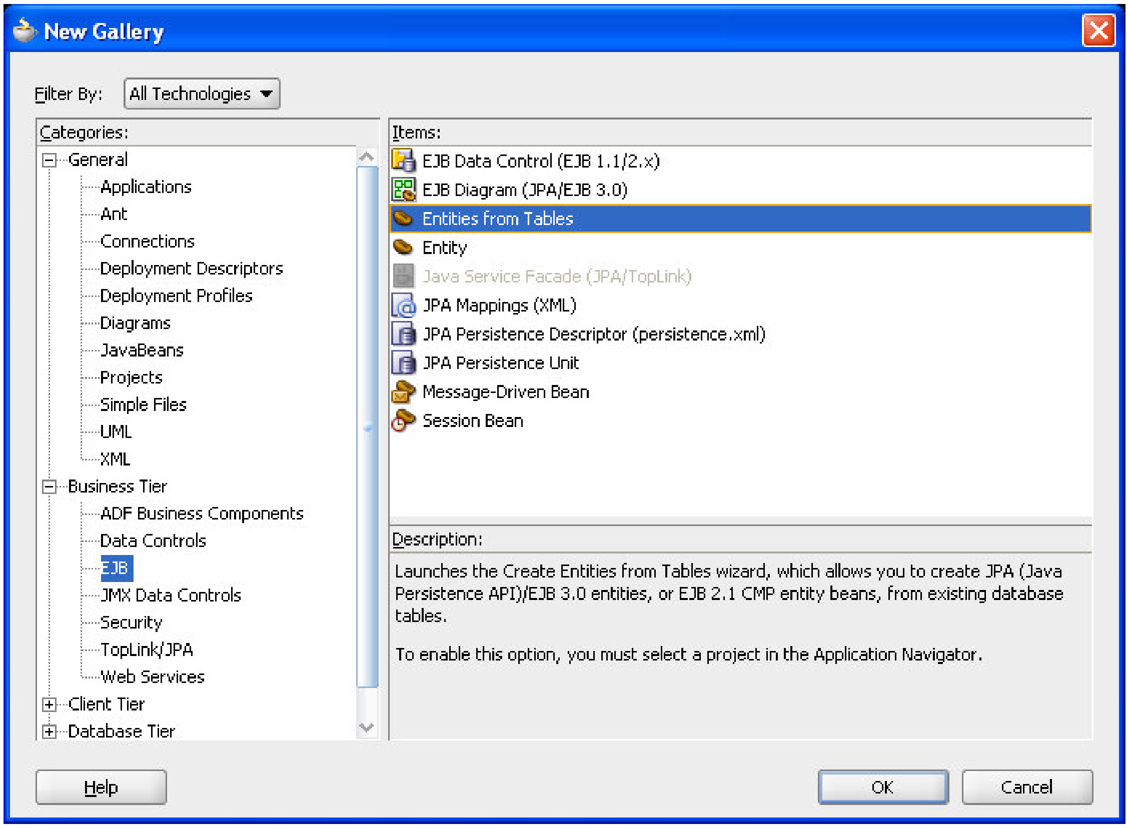
Oracle (JDBC) Settings

☐ Enter Custom JDBC URL
Driver: thin
Host Name: localhost
JDBC Port: 1521
● SID: ORCL
○ Service Name: XE

Success!

- Right click on the *Lab9* project and choose *New*.

- Under *Business Tier* select *EJB* and then select *Entities From Tables*.

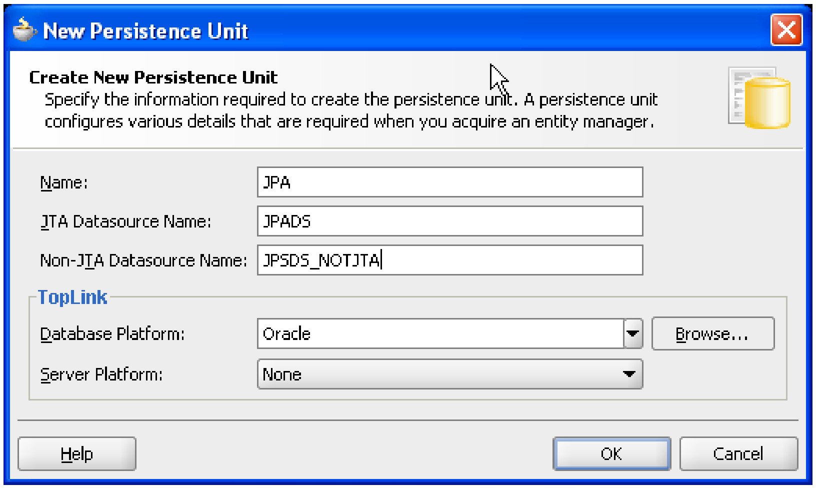


Launches the Create Entities from Tables wizard, which allows you to create JPA (Java Persistence API)/EJB 3.0 entities, or EJB 2.1 CMP entity beans, from existing database tables.

To enable this option, you must select a project in the Application Navigator.

- Click *Next* to continue.
- Choose *EJB 3.0 JPA Entities* and then click *Next*.

- Click on *New* to create a new persistence unit. (Each persistence-unit defines a set of classes and their mapping characteristics when persisting them). Enter the details and click *OK*. Once the screen returns, click on *Next*.



New Persistence Unit

Create New Persistence Unit

Specify the information required to create the persistence unit. A persistence unit configures various details that are required when you acquire an entity manager.

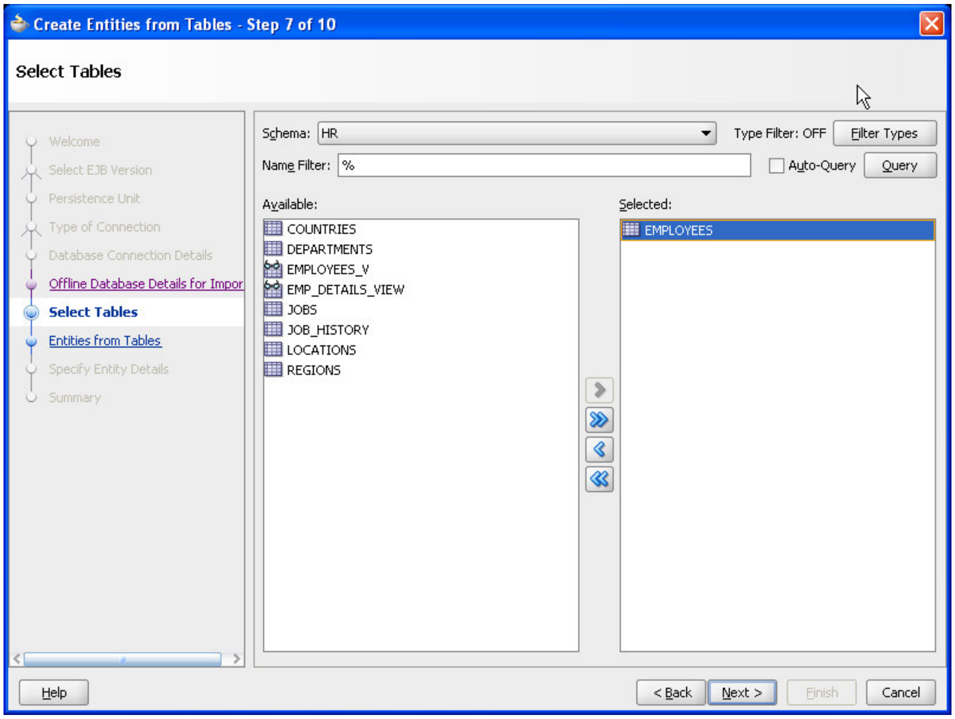
Name: JPA
JTA Datasource Name: JPADS
Non-JTA Datasource Name: JPSDS_NOTJTA

TopLink
Database Platform: Oracle
Server Platform: None

- Choose *Online Database Connection* and click *Next*.

- Ensure *Lab9* project is selected and click *Next*.

- Query for the EMPLOYEES table and select is as in the screen-shot below and click *Next*.



Create Entities from Tables - Step 7 of 10

Select Tables

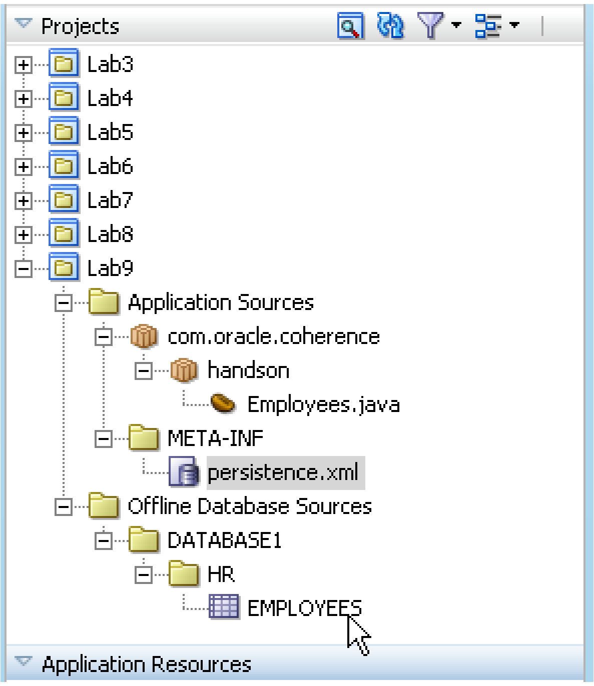
Schema: HR
Type Filter: OFF
Name Filter: %
☐ Auto-Query

Available:
COUNTRIES
DEPARTMENTS
EMPLOYEES_V
EMP_DETAILS_VIEW
JOBS
JOB_HISTORY
LOCATIONS
REGIONS

Selected:
EMPLOYEES

- On Leave the Entity Class Options as they are and click *Next*.

- Then click on *Finish* to complete the creation. You should now see the following in the navigator.



- Replace the contents of the persistence.xml with the following. (You can find this in the [LABS_HOME] directory. Ensure that the connection details match your database connection details.

```
<?xml version="1.0" encoding="windows-1252" ?>
<persistence xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd"
             version="1.0" xmlns="http://java.sun.com/xml/ns/persistence">
  <persistence-unit name="JPA">
    <provider>oracle.toplink.essentials.PersistenceProvider</provider>

    <class>
      com.oracle.coherence.handson.Employees
    </class>
    <properties>
      <property name="toplink.jdbc.driver" value="oracle.jdbc.OracleDriver"/>
      <property name="toplink.jdbc.url" value="jdbc:oracle:thin:@localhost:1521:ORCL"/>
      <property name="toplink.jdbc.user" value="hr"/>
      <property name="toplink.jdbc.password" value="hr"/>
    </properties>

   </persistence-unit>

</persistence>
```

Copy the cache-server.cmd file

- In the [LABS_HOME]\bin directory, copy the file cache-server.cmd to jpa-cache-server.cmd.

- Add the following switch to the java command. (Replacing labs home with the appropriate value).

```
-Dtangosol.coherence.cacheconfig=[LABS_HOME]\jpa-cache-config.xml
```

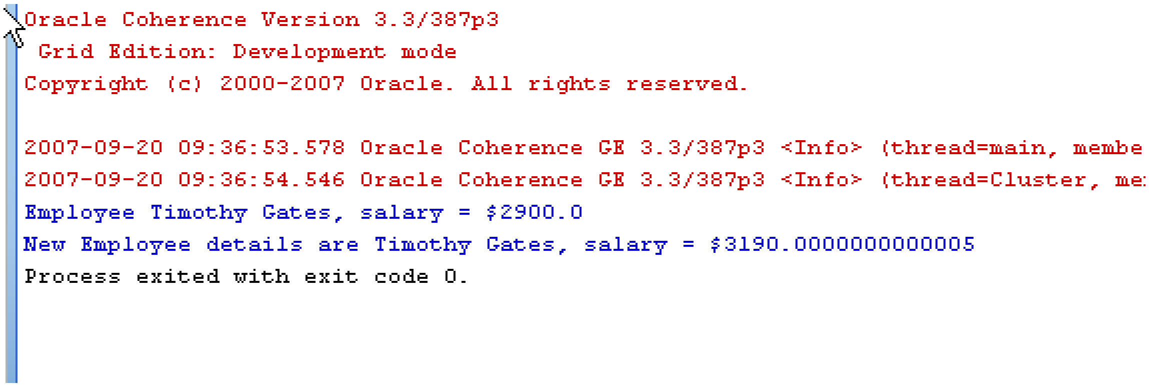
- Add the following classpath to the –cp argument.

```
[LABS_HOME]\Labs\Lab9\classes
```

- We must also add the following jars into the classpath:
  - [LABS_HOME]\coherence\lib\coherence-jpa.jar – Coherence JPA libraries
  - [JDEV_HOME]\jdbc\lib\ojdbc14.jar – JDBC libraries
  - [JDEV_HOME]\lib\java\internal\toplink-essentials.jar – Toplink Libraries
- Once this is complete, run the jpa-cache-server.cmd

Create a new class to interact with the *Employee* Object

- Create a new class called *RunEmployeeExample*. Ensure this has a main method.
- Create the code to do the following:
  - Get en Employee using the EMPLOYEE_ID. The EMPLOYEE_ID will need to be a long.
  - Display the salary
  - Give them a 10% pay rise.
  - Get the value again to confirm the pay rise.
- Run the code and you should see output similar to the following.



```
Oracle Coherence Version 3.3/387p3
 Grid Edition: Development mode
Copyright (c) 2000-2007 Oracle. All rights reserved.

2007-09-20 09:36:53.578 Oracle Coherence GE 3.3/387p3 <Info> (thread=main, membe
2007-09-20 09:36:54.546 Oracle Coherence GE 3.3/387p3 <Info> (thread=Cluster, me
Employee Timothy Gates, salary = $2900.0
New Employee details are Timothy Gates, salary = $3190.0000000000005
Process exited with exit code 0.
```

- What has happened?

Now that the Employees class has been annotated to persist to the database using JPA, and we have included the persistence.xml file to tell JPA where our database is, Coherence will use a CacheStore implementation that uses JPA to load and store objects to the database.

When we use the get(Object key) method the following happens:

- Coherence will look for the entry with the key *key*.
- If the entry has not already been cached, or it has been expired from the cache, then it will ask the backing map, which uses JPA and toplink to retrieve the data.
- If the entry is in the cache then it will be return directly to the application without going through TopLink

When we use the put(Object Key, Object Value), Coherence then uses JPA via Toplink to persist any changes to the database.

Where to next?

Hopefully this has given you a taste of how easy it is to work with data in a Coherence data-grid, the increased processing power you automatically get when you add cache servers, and the some of the features available.

There are many more features and abilities in Coherence, for more information a few good places to start are:

- http://www.oracle.com/technology/products/coherence/index.html
- http://wiki.tangosol.com/display/COH/Tangosol+Coherence+Knowledge+Base+Home

Lab Solutions

Lab 3 Solutions: Access the data grid from Java

1. Example 1 - MyFirstSample

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

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

public class MyFirstSample {
    public MyFirstSample() {
    }

    public static void main(String[] args) {

       // ensure we are in a cluser
       CacheFactory.ensureCluster();

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

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

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

Example 2 - MyFirstSampleReader

```
package com.oracle.coherence.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 cluser
        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"));
    }
}
```

Example 3. Bonus

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

import com.tangosol.net.CacheFactory;
import com.tangosol.net.CacheService;
import com.tangosol.net.Cluster;
import com.tangosol.net.Member;
import com.tangosol.net.NamedCache;

import java.util.Iterator;


public class CacheInfo {
    public CacheInfo() {
    }

    public static void main(String[] args) {

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

        // get the information about the cluster and local member
        Cluster cluster    = CacheFactory.getCluster();
        Member  memberThis = cluster.getLocalMember();

        System.out.println("Local member info: " + memberThis);

        // list all the members - highlighting the curret one
        for (Iterator iter = cluster.getMemberSet().iterator();
             iter.hasNext(); )  {
            Member member = (Member) iter.next();

            System.out.println(member +
                (member.equals(memberThis) ? " <-- this node" : ""));
        }


    }
}
```

Lab 4 Solutions: Leaving Strings behind - Working with complex Objects.

1. Example 1. Person Class

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

public class Person implements java.io.Serializable {

    private int id;
    private String surname;
    private String firstname;
    private String address;
    private int age;
    private String gender;

    public Person() {
    }

    public Person(int id1, String surname1, String firstname1,
                  String address1,
                  int age1, String gender1) {
        super();
        this.id = id1;
        this.surname = surname1;
        this.firstname = firstname1;
        this.address = address1;
        this.age = age1;
        this.gender = gender1;
    }

    public void setId(int param) {
        this.id = param;
    }

    public int getId() {
        return id;
    }

    public void setSurname(String param) {
        this.surname = param;
    }

    public String getSurname() {
        return surname;
    }

    public void setFirstname(String param) {
        this.firstname = param;
    }

    public String getFirstname() {
        return firstname;
    }
    public void setAddress(String param) {
        this.address = param;
    }

    public String getAddress() {
        return address;
    }

    public void setAge(int param) {
        this.age = param;
    }

    public int getAge() {
        return age;
    }

    public void setGender(String param) {
        this.gender = param;
    }

    public String getGender() {
        return gender;
    }


    @Override
    public boolean equals(Object object) {
        if (this == object) {
            return true;
        }
        if (!(object instanceof Person)) {
            return false;
        }
        final Person other = (Person)object;
        if (id != other.id) {
            return false;
        }
        if (!(surname == null ? other.surname == null :
            surname.equals(other.surname))) {
            return false;
        }
        if (!(firstname == null ? other.firstname == null :
            firstname.equals(other.firstname))) {
            return false;
        }
        if (!(address == null ? other.address == null :
            address.equals(other.address))) {
            return false;
        }
        if (age != other.age) {
            return false;
        }
        if (!(gender == null ? other.gender == null :
            gender.equals(other.gender))) {
            return false;
        }
        return true;
    }

    @Override
    public int hashCode() {
        final int PRIME = 37;
        int result = 1;
        result = PRIME * result + ((surname == null) ? 0 :
                 surname.hashCode());
        result = PRIME * result + ((firstname == null) ? 0 :
                 firstname.hashCode());
        result = PRIME * result + ((address == null) ? 0 :
                 address.hashCode());
        result = PRIME * result + ((gender == null) ? 0 :
                 gender.hashCode());
        return result;
    }
}
```

Example 2 - PersonCache

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

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

public class PersonCache {
    public PersonCache() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");

        Person p1 = new Person(1,"Middleton","Tim",
                "Level 2, 66 Kings Park Road, West Perth",
                39,Person.MALE);

        person.put(p1.getId(),p1);

        Person p2 = (Person)person.get(p1.getId());

        if (p2.equals(p1)) {
            System.out.println("They are the same!!");
        }
    }
}
```

Example 3 – Person class using ExternalizableLite

```
package com.oracle.coherence.handson;
import com.tangosol.io.ExternalizableLite;
import com.tangosol.util.ExternalizableHelper;
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;

public class Person implements ExternalizableLite {

    private int id;
    private String surname;
    private String firstname;
    private String address;
    private int age;
    private String gender;

    public static String MALE = "M";
    public static String FEMALE = "F";

    public Person() {
    }

    public Person(int id1, String surname1, String firstname1,
                  String address1,
                  int age1, String gender1) {
        super();
        this.id = id1;
        this.surname = surname1;
        this.firstname = firstname1;
        this.address = address1;
        this.age = age1;
        this.gender = gender1;
    }

    public void setId(int param) {
        this.id = param;
    }

    public int getId() {
        return id;
    }

    public void setSurname(String param) {
        this.surname = param;
    }

    public String getSurname() {
        return surname;
    }

    public void setFirstname(String param) {
        this.firstname = param;
    }
    public String getFirstname() {
        return firstname;
    }

    public void setAddress(String param) {
        this.address = param;
    }

    public String getAddress() {
        return address;
    }

    public void setAge(int param) {
        this.age = param;
    }

    public int getAge() {
        return age;
    }

    public void setGender(String param) {
        this.gender = param;
    }

    public String getGender() {
        return gender;
    }


    @Override
    public boolean equals(Object object) {
        if (this == object) {
            return true;
        }
        if (!(object instanceof Person)) {
            return false;
        }
        final Person other = (Person)object;
        if (id != other.id) {
            return false;
        }
        if (!(surname == null ? other.surname == null :
            surname.equals(other.surname))) {
            return false;
        }
        if (!(firstname == null ? other.firstname == null :
            firstname.equals(other.firstname))) {
            return false;
        }
        if (!(address == null ? other.address == null :
            address.equals(other.address))) {
            return false;
        }
        if (age != other.age) {
            return false;
        }
        if (!(gender == null ? other.gender == null :
            gender.equals(other.gender))) {
            return false;
        }
        return true;
    }

    @Override
    public int hashCode() {
        final int PRIME = 37;
        int result = 1;
        result = PRIME * result + ((surname == null) ? 0 :
                 surname.hashCode());
        result = PRIME * result + ((firstname == null) ? 0 :
                 firstname.hashCode());
        result = PRIME * result + ((address == null) ? 0 :
                 address.hashCode());
        result = PRIME * result + ((gender == null) ? 0 :
                 gender.hashCode());
        return result;
    }

    public void readExternal(DataInput dataInput) throws IOException {
        this.id = ExternalizableHelper.readInt(dataInput);
        this.surname = ExternalizableHelper.readSafeUTF(dataInput);
        this.firstname = ExternalizableHelper.readSafeUTF(dataInput);
        this.address = ExternalizableHelper.readSafeUTF(dataInput);
        this.gender = ExternalizableHelper.readSafeUTF(dataInput);
        this.age = ExternalizableHelper.readInt(dataInput);
    }

    public void writeExternal(DataOutput dataOutput) throws IOException {
        ExternalizableHelper.writeInt(dataOutput, this.id);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.surname);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.firstname);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.address);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.gender);
        ExternalizableHelper.writeInt(dataOutput, this.age);
    }
}
```

Lab 5 Solutions - Querying & Aggregating Data in a cache

1. Example 1 – PopulatePeople

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

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

import java.io.IOException;

import java.util.Random;

public class PopulatePeople {
    public PopulatePeople() {
    }

    public static void main(String[] args) throws IOException {

        NamedCache person = CacheFactory.getCache("person");
        Random generator = new Random();

        for (int i = 1; i <= 10000; i++) {
            Person p = new Person(i, "Surname" + i, "Firstname" + i,
                         Address" + i, generator.nextInt(100) + 1,
             (generator.nextInt(2) == 1 ? Person.FEMALE : Person.MALE) ) ;
            person.put(p.getId(),p);

        }

        System.out.println(person.entrySet().size());

    }
}
```

Example 2 – QueryExample

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

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

import com.tangosol.util.filter.AndFilter;
import com.tangosol.util.filter.EqualsFilter;

import com.tangosol.util.filter.GreaterEqualsFilter;

import java.util.Set;

public class QueryExample {
    public QueryExample() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");

        // get a set of all males
        Set males = person.entrySet(
                        new EqualsFilter("getGender",Person.MALE ));

        Set malesOver35 = person.entrySet(
                        new AndFilter(
                         new EqualsFilter("getGender",Person.MALE ),
                         new GreaterEqualsFilter("getAge",35)) );

        System.out.println("Total number of males is " + males.size());
        System.out.println("Total number of males > 35 " +
                           malesOver35.size());
    }
}
```

Example 3 – AggregationExample (no timings)

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

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.util.Filter;
import com.tangosol.util.aggregator.DoubleAverage;
import com.tangosol.util.aggregator.DoubleMax;
import com.tangosol.util.filter.AlwaysFilter;
import com.tangosol.util.filter.EqualsFilter;

import java.math.BigDecimal;

public class AggregationExample {
    public AggregationExample() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");

        // create a new query

        Double averageAgeMales = (Double)person.aggregate(
                          new EqualsFilter("getGender",Person.MALE ),
                          new DoubleAverage("getAge")
                          );

        Double averageAgeFemales = (Double)person.aggregate(
                          new EqualsFilter("getGender",Person.FEMALE ),
                          new DoubleAverage("getAge")
                          );

        Double maxAge = (Double)person.aggregate(
                          (Filter)null,   // new AlwaysFilter()
                          new DoubleMax("getAge")
                          );

        Double averageAge = (Double)person.aggregate(
                          new AlwaysFilter(),
                          new DoubleAverage("getAge")
                          );

        System.out.println("Average age of males is " + averageAgeMales);
        System.out.println("Average age of femals is " +
                            averageAgeFemales);
        System.out.println("Max age is " + maxAge);
        System.out.println("Average age  " + averageAge);
    }
}
```

Example 4 – AggregationExample – timings in loop

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

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.util.Filter;
import com.tangosol.util.aggregator.DoubleAverage;
import com.tangosol.util.aggregator.DoubleMax;
import com.tangosol.util.filter.AlwaysFilter;
import com.tangosol.util.filter.EqualsFilter;

import java.math.BigDecimal;

public class AggregationExample {
    public AggregationExample() {
    }

    public static void main(String[] args) {
        Double averageAgeMales = null;
        Double averageAgeFemales = null;
        Double maxAge = null;
        Double averageAge = null;
        int max = 100;

        NamedCache person = CacheFactory.getCache("person");
        long totalTime = 0;

        // create a new query

        for (int i = 0; i< max; i++) {
          long startTime = System.currentTimeMillis();

          averageAgeMales = (Double)person.aggregate(
                          new EqualsFilter("getGender",Person.MALE ),
                          new DoubleAverage("getAge")
                          );

          averageAgeFemales = (Double)person.aggregate(
                          new EqualsFilter("getGender",Person.FEMALE ),
                          new DoubleAverage("getAge")
                          );

          maxAge = (Double)person.aggregate(
                          (Filter)null,   // new AlwaysFilter()
                          new DoubleMax("getAge")
                          );

          averageAge = (Double)person.aggregate(
                          new AlwaysFilter(),
                          new DoubleAverage("getAge")
                          );


          long endTime = System.currentTimeMillis();

          System.out.println("Total time taken is " + (endTime -
                             startTime) / 1000F + " seconds");
          totalTime += (endTime - startTime);
        }

        System.out.println("Average time is " +
                             (totalTime / max) / 1000F);
        //System.out.println("Average age of males is " +
                              averageAgeMales);
        //System.out.println("Average age of femals is " +
                              averageAgeFemales);
        //System.out.println("Max age is " + maxAge);
        //System.out.println("Average age  " + averageAge);
    }
}
```

Example 4 – Bonus – Add indexes for speed

```
   ...

   NamedCache person = CacheFactory.getCache("person");

   // add indexes
   person.addIndex(new ReflectionExtractor("getGender"), true, null);
   person.addIndex(new ReflectionExtractor("getAge"), false, null);

   ...
```

Lab 6 – Solutions – Observing Data Changes

1. Example 1 – ListenForNewPerson

```
package com.oracle.coherence.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.filter.EqualsFilter;
import com.tangosol.util.filter.MapEventFilter;

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

public class ListenForNewPerson {
    public ListenForNewPerson() {
    }

    public static void main(String[] args) throws IOException {
        // connect to named cache
        NamedCache person = CacheFactory.getCache("person");

        // listen for insert events on Person
        // This can be done in an easier way by using a new
        //  AbstractMapListener()
        // and then overriding only the method you want to
        //
        person.addMapListener(new MapListener()
        {
          public void entryDeleted(MapEvent mapEvent) {
            // ignore
          }
          public void entryInserted(MapEvent mapEvent) {
             Person p = (Person)mapEvent.getNewValue();
             System.out.println("New person added: " + p.getFirstname() +
                              " " + p.getSurname());
          }
          public void entryUpdated(MapEvent mapEvent) {
             // ignore
          }
        }
        );


        System.out.println("waiting for events");
        BufferedReader console = new BufferedReader(
                               new InputStreamReader(System.in));
        String text = console.readLine();

    }
}
```

Example 2 – ListenForUpdatedPerson

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

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.util.AbstractMapListener;
import com.tangosol.util.MapEvent;
import com.tangosol.util.MapListener;
import com.tangosol.util.filter.EqualsFilter;
import com.tangosol.util.filter.MapEventFilter;

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

public class ListenForUpdatedPerson {
    public ListenForUpdatedPerson() {
    }

    public static void main(String[] args) throws IOException {
        // connect to named cache
        NamedCache person = CacheFactory.getCache("person");

        // listen for insert events on Person

        person.addMapListener(new AbstractMapListener()
        {
              public void entryUpdated(MapEvent mapEvent) {
                Person oldPerson = (Person)mapEvent.getOldValue();
                 Person newPerson = (Person)mapEvent.getNewValue();

                 // better to implement toString() on the person object to
                 // display it.. :)
                 System.out.println("Old person is " +
                                    oldPerson.getFirstname() + " " +
                                    oldPerson.getSurname());
                 System.out.println("New person is " +
                                    newPerson.getFirstname() + " " +
                             newPerson.getSurname());
               }
           },
          new MapEventFilter(MapEventFilter.E_UPDATED, new
                             EqualsFilter("getGender", Person.MALE)),
                             false   // not a lite event

        );


        System.out.println("waiting for events");
        BufferedReader console =
               new BufferedReader(new InputStreamReader(System.in));
        String text = console.readLine();


    }
}
```

Example 3 – ListenForUpdatedPerson

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

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

public class PersonEventTester {
    public PersonEventTester() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");

        Person p1 = new Person(1,"Middleton","Tim",
                  "Level 2, 66 Kings Park Road, West Perth",
                  39,Person.MALE);

       System.out.println("put person");
       person.put(p1.getId(),p1);

       Person p2 = (Person)person.get(p1.getId());
       p2.setFirstname("Timothy");

        System.out.println("Update person");
       person.put(p2.getId(),p2);
    }
}
```

Exmaple 4 – Bonus ChatMessage

```
package com.oracle.coherence.handson.chat;

import java.io.Serializable;

public class ChatMessage implements Serializable {

    private String from;
    private long entryTime;
    private String message;

    public ChatMessage() {
    }

    public ChatMessage(String from, String message) {
        this.from = from;
        this.message = message;
        this.entryTime = System.currentTimeMillis();
    }


    public void setFrom(String from) {
        this.from = from;
    }

    public String getFrom() {
        return from;
    }

    public void setEntryTime(long entryTime) {
        this.entryTime = entryTime;
    }

    public long getEntryTime() {
        return entryTime;
    }

    public void setMessage(String message) {
        this.message = message;
    }

    public String getMessage() {
        return message;
    }
}
```

Example 5 – ChatClient

Note: There are a number of ways of doing this. Here is just an example.

```
package com.oracle.coherence.handson.chat;

import com.tangosol.net.*;

import java.io.*;
import com.tangosol.util.UUID;
import com.tangosol.util.filter.*;
import com.tangosol.util.MapEvent;
import com.tangosol.util.AbstractMapListener;

import java.util.Date;
import java.util.Iterator;
import java.util.Map;
import java.util.Set;

public class ChatClient {

    public static void main(String[] args) {
      String userName = null;
      String message;
      NamedCache chatmembers = null;

       try {

           System.out.println("Welcome to the Coherence Chat Client");
           System.out.println("------------------------------------");

           BufferedReader console = new BufferedReader(new
                                      InputStreamReader(System.in));

           System.out.print("User Name:");
           userName = console.readLine();

           // join the chatroom named cache as storage enabled = true
           NamedCache cache = CacheFactory.getCache("chatroom");
           chatmembers = CacheFactory.getCache("chatmembers");

           chatmembers.put(userName,userName);

           // register a listener to display the messages
           cache.addMapListener(new AbstractMapListener() {
                   public void entryInserted(MapEvent event) {
                      ChatMessage msg = (ChatMessage)event.getNewValue();

                      System.out.println("From: " + msg.getFrom());
                      System.out.println("Time: " + new
                                         Date(msg.getEntryTime()));
                      System.out.println("Mesg: " + msg.getMessage() );

                      System.out.println();
                    } }
                    ,
                    new MapEventFilter(MapEvent.ENTRY_INSERTED, new
                            NotEqualsFilter("getFrom", userName)),
                            false);
           chatmembers.addMapListener( new AbstractMapListener() {
                 public void entryDeleted(MapEvent event) {
                     String who = (String)event.getOldValue();
                     System.out.println(who + " has left the chat");
                 }

                 public void entryInserted(MapEvent event) {
                     String who = (String)event.getNewValue();
                     System.out.println(who + " has entered the chat");
                 }
              }
           );

           do {
               System.out.print("\nEnter message or bye to quit: ");
               message = console.readLine();

               if ("bye".equals(message))
                  break;
               // else add this to the chat
               else if ("help".equals(message)) {
                   System.out.println("HELP:");
                   System.out.println("bye - quit");
                   System.out.println("who - list of users \n");
               }
               else if ("who".equals(message)) {
                   System.out.println("Current chat memebers");
                   System.out.println("====================");
                   Set s = chatmembers.entrySet();
                   for (Iterator<Map.Entry> entries =
                chatmembers.entrySet().iterator() ; entries.hasNext();) {
                       Map.Entry entry = entries.next();
                       String member = (String)entry.getValue();

                       System.out.println(member);
                   }
               }
               else {

                cache.put(new UUID(),new ChatMessage(userName, message));

               }

           } while (true);

           System.out.println("Bye");
       }
       catch (Exception e) {
           e.printStackTrace();
       }
       finally {
           chatmembers.remove(userName);
       }

    }
}
```

Lab 7 Solutions - In-Place processing of data

1. Example 1 – Employee Class

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

import com.tangosol.io.ExternalizableLite;
import com.tangosol.util.ExternalizableHelper;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;

import java.math.BigDecimal;


public class Employee implements  ExternalizableLite {
    private int empId;
    private String surname;
    private String firstname;
    private double salary;


    public Employee() {
    }

    public Employee(int empId1, String surname1, String firstname1,
                    double salary1) {
        super();
        this.empId = empId1;
        this.surname = surname1;
        this.firstname = firstname1;
        this.salary = salary1;
    }

    public void setEmpId(int param) {
        this.empId = param;
    }

    public int getEmpId() {
        return empId;
    }

    public void setSurname(String param) {
        this.surname = param;
    }

    public String getSurname() {
        return surname;
    }

    public void setFirstname(String param) {
        this.firstname = param;
    }
    public String getFirstname() {
        return firstname;
    }

    public void setSalary(double param) {
        this.salary = param;
    }

    public double getSalary() {
        return salary;
    }

    @Override
    public boolean equals(Object object) {
        if (this == object) {
            return true;
        }
        if (!(object instanceof Employee)) {
            return false;
        }
        final Employee other = (Employee)object;
        if (empId != other.empId) {
            return false;
        }
        if (!(surname == null ? other.surname == null :
            surname.equals(other.surname))) {
            return false;
        }
        if (!(firstname == null ? other.firstname == null :
            firstname.equals(other.firstname))) {
            return false;
        }
        if (Double.compare(salary, other.salary) != 0) {
            return false;
        }
        return true;
    }

    @Override
    public int hashCode() {
        final int PRIME = 37;
        int result = 1;
        result = PRIME * result + ((surname == null) ? 0 :
                 surname.hashCode());
        result = PRIME * result + ((firstname == null) ? 0 :
                 firstname.hashCode());
        long temp = Double.doubleToLongBits(salary);
        result = PRIME * result + (int) (temp ^ (temp >>> 32));
        return result;
    }

    public void readExternal(DataInput dataInput) throws IOException {
        this.empId = ExternalizableHelper.readInt(dataInput);
        this.surname = ExternalizableHelper.readSafeUTF(dataInput);
        this.firstname = ExternalizableHelper.readSafeUTF(dataInput);
        this.salary =
           ExternalizableHelper.readBigDecimal(dataInput).doubleValue();
    }

    public void writeExternal(DataOutput dataOutput) throws IOException {
        ExternalizableHelper.writeInt(dataOutput, this.empId);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.surname);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.firstname);
        ExternalizableHelper.writeBigDecimal(dataOutput,
                        new BigDecimal(this.salary));

    }
}
```

Example 2 – RaiseSalary

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

import com.tangosol.util.processor.AbstractProcessor;
import com.tangosol.util.InvocableMap.Entry;

import java.util.Map;

public class RaiseSalary extends AbstractProcessor {
    public RaiseSalary() {
    }

    public Object process(Entry entry ) {
        Employee emp = (Employee)entry.getValue();
        emp.setSalary(emp.getSalary() * 1.10);
        entry.setValue(emp);
        return null;
    }
}
```

Example 3 – InvokeTest

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

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.util.filter.AlwaysFilter;

public class InvokeTest {
    public InvokeTest() {
    }

    public static void main(String[] args) {

        NamedCache empCache = CacheFactory.getCache("employees");

        Employee e1 = new Employee(1,"Middleton","Tim",5000);
        empCache.put(e1.getEmpId(), e1);

        Employee e2 = new Employee(2,"Jones","Chris",10000);
        empCache.put(e2.getEmpId(), e2);

        empCache.invokeAll(AlwaysFilter.INSTANCE, new RaiseSalary());

        e1 = (Employee)empCache.get(e1.getEmpId());
        e2 = (Employee)empCache.get(e2.getEmpId());

        System.out.println("Salary for emp 1 is now: " + e1.getSalary());
        System.out.println("Salary for emp 2 is now: " + e2.getSalary());

    }
}
```

Example 4 – SayHelloProcessor

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

import com.tangosol.util.processor.AbstractProcessor;
import com.tangosol.util.InvocableMap.Entry;

public class SayHelloProcessor extends AbstractProcessor {
    public SayHelloProcessor() {

    }

    public Object process(Entry entry ) {
        Employee emp = (Employee)entry.getValue();
        System.out.println("\nHello from " + emp.getFirstname() + " " +
                        emp.getSurname() + "\n");
        return null;
    }
}
```

Example 5 – WhereAreMyEmployees

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

import com.tangosol.net.CacheFactory;
import com.tangosol.net.NamedCache;
import com.tangosol.util.filter.AlwaysFilter;

public class WhereAreMyEmployees {
    public WhereAreMyEmployees() {
    }

    public static void main(String[] args) {

        NamedCache empCache = CacheFactory.getCache("employees");

        empCache.invokeAll(AlwaysFilter.INSTANCE,
                           new SayHelloProcessor());
    }
}
```

Lab 8 Solutions - Extending your objects with Cache Keys & Data Affinity

1. Example 1 – Person class with Key and KeyAssociation

```
package com.oracle.coherence.handson;
import com.tangosol.io.ExternalizableLite;
import com.tangosol.net.cache.KeyAssociation;
import com.tangosol.util.ExternalizableHelper;
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;

public class Person implements ExternalizableLite {

    private int id;
    private String surname;
    private String firstname;
    private String address;
    private int age;
    private String gender;

    public static String MALE = "M";
    public static String FEMALE = "F";

    public Person() {
    }

    public Person(int id1, String surname1, String firstname1, String address1,
                  int age1, String gender1) {
        super();
        this.id = id1;
        this.surname = surname1;
        this.firstname = firstname1;
        this.address = address1;
        this.age = age1;
        this.gender = gender1;
    }

    public Key getKey() {
          return new Key(this);
    }


    public void setId(int param) {
        this.id = param;
    }

    public int getId() {
        return id;
    }

    public void setSurname(String param) {
        this.surname = param;
    }

    public String getSurname() {
        return surname;
    }

    public void setFirstname(String param) {
        this.firstname = param;
    }

    public String getFirstname() {
        return firstname;
    }
```

```
public void setAddress(String param) {
    this.address = param;
}

public String getAddress() {
    return address;
}

public void setAge(int param) {
    this.age = param;
}

public int getAge() {
    return age;
}

public void setGender(String param) {
    this.gender = param;
}

public String getGender() {
    return gender;
}

public double getAgeDouble() {
    return (double)this.age;
}

@Override
public boolean equals(Object object) {
    if (this == object) {
        return true;
    }
    if (!(object instanceof Person)) {
        return false;
    }
    final Person other = (Person)object;
    if (id != other.id) {
        return false;
    }
    if (!(surname == null ? other.surname == null :
         surname.equals(other.surname))) {
        return false;
    }
    if (!(firstname == null ? other.firstname == null :
         firstname.equals(other.firstname))) {
        return false;
    }
    if (!(address == null ? other.address == null : address.equals(other.address))) {
        return false;
    }
    if (age != other.age) {
        return false;
    }
    if (!(gender == null ? other.gender == null : gender.equals(other.gender))) {
        return false;
    }
    return true;
}

@Override
public int hashCode() {
    final int PRIME = 37;
    int result = 1;
    result = PRIME * result + ((surname == null) ? 0 : surname.hashCode());
    result = PRIME * result + ((firstname == null) ? 0 : firstname.hashCode());
    result = PRIME * result + ((address == null) ? 0 : address.hashCode());
    result = PRIME * result + ((gender == null) ? 0 : gender.hashCode());
    return result;
}
```

```
    public void readExternal(DataInput dataInput) throws IOException {
        this.id = ExternalizableHelper.readInt(dataInput);
        this.surname = ExternalizableHelper.readSafeUTF(dataInput);
        this.firstname = ExternalizableHelper.readSafeUTF(dataInput);
        this.address = ExternalizableHelper.readSafeUTF(dataInput);
        this.gender = ExternalizableHelper.readSafeUTF(dataInput);
        this.age = ExternalizableHelper.readInt(dataInput);
    }

    public void writeExternal(DataOutput dataOutput) throws IOException {
        ExternalizableHelper.writeInt(dataOutput, this.id);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.surname);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.firstname);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.address);
        ExternalizableHelper.writeSafeUTF(dataOutput, this.gender);
        ExternalizableHelper.writeInt(dataOutput, this.age);
    }

    public static class Key implements ExternalizableLite, KeyAssociation {

        // lets define a key of id and version
        private int id;
        private int version;
        private String surname;

        public Key() {
                //for serializble
        }

        public Object getAssociatedKey() {
           return surname;
        }

        public Key(int id, int version) {
           this.id = id;
           this.version = version;
        }

        public Key(Person p) {
            this.id = p.getId();
            this.version = 1;  // default
            this.surname = p.getSurname();
        }

        public void writeExternal(DataOutput dataOutput) throws IOException {
            ExternalizableHelper.writeInt(dataOutput, this.id);
            ExternalizableHelper.writeInt(dataOutput, this.version);
            ExternalizableHelper.writeSafeUTF(dataOutput, this.surname);
        }

        public void readExternal(DataInput dataInput) throws IOException {
            this.id = ExternalizableHelper.readInt(dataInput);
            this.version = ExternalizableHelper.readInt(dataInput);
            this.surname = ExternalizableHelper.readSafeUTF(dataInput);
        }

        @Override
        public boolean equals(Object object) {
            if (this == object) {
                return true;
            }
            if (!(object instanceof Person.Key)) {
                return false;
            }
            final Person.Key other = (Person.Key)object;
            if (id != other.id) {
                return false;
            }
            if (version != other.version) {
                return false;
            }
            if (surname != other.surname) {
                return false;
            }
            return true;
        }

        @Override
        public int hashCode() {
            final int PRIME = 37;
            int result = 1;
            return result;
        }
    }
}
```

Example 2 – KeyTester

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

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

public class KeyTester {
    public KeyTester() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");

        Person p = new Person(1,"Middleton","Tim","Address",29,Person.MALE);

        person.put(p.getKey(), p);

        Person p2 = (Person)person.get(new Person.Key(1,1));

        System.out.println("Person is " + p2.getSurname() + " " +  p2.getSurname());


    }
}
```

Example 3 – KeyTester2

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

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

import java.util.LinkedList;

public class KeyTester2 {
    public KeyTester2() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");
        LinkedList peopleList = new LinkedList();

        // build the list
        peopleList.add( new Person(1,"Flinstone","Fred","Address",29,Person.MALE));
        peopleList.add( new Person(2,"Flinstone","Wilma","Address",29,Person.FEMALE));
        peopleList.add( new Person(3,"Rubble","Barney","Address",44,Person.MALE));
        peopleList.add( new Person(4,"Rubble","Betty","Address",44,Person.FEMALE));
        peopleList.add( new Person(5,"Rubble","Dino","Address",44,Person.FEMALE));

        for (java.util.Iterator iterator = peopleList.iterator(); iterator.hasNext();) {

                Person p = (Person)iterator.next();
                person.put(p.getKey(), p);

                Member m =
((DistributedCacheService)person.getCacheService()).getKeyOwner(p.getKey());
                System.out.println("Person " + p.getFirstname() + " " + p.getSurname() +
                                   " is owned by member " + m.getId());
        }


    }
}
```

Bonus Example 4 – WhereAreYouProcessor

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

import com.tangosol.util.processor.AbstractProcessor;
import com.tangosol.util.InvocableMap.Entry;

public class WhereAreYouProcessor extends AbstractProcessor {
    public WhereAreYouProcessor() {
    }

    public Object process(Entry entry ) {
        Person p = (Person)entry.getValue();
        System.out.println("\nHello from " + p.getFirstname() + " " +
                           p.getSurname() + "\n");
        return null;
    }
}
```

Bonus Example 5 – EntryProcessorExample

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

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

import com.tangosol.util.filter.AlwaysFilter;

import java.util.LinkedList;

public class EntryProcessorExample {
    public EntryProcessorExample() {
    }

    public static void main(String[] args) {

        NamedCache person = CacheFactory.getCache("person");
        LinkedList peopleList = new LinkedList();

        // build the list
        peopleList.add(
            new Person(1,"Flinstone","Fred","Address",29,Person.MALE));
        peopleList.add(
            new Person(2,"Flinstone","Wilma","Address",29,Person.FEMALE));
        peopleList.add(
            new Person(3,"Rubble","Barney","Address",44,Person.MALE));
        peopleList.add(
            new Person(4,"Rubble","Betty","Address",44,Person.FEMALE));
        peopleList.add(
            new Person(5,"Rubble","Dino","Address",44,Person.FEMALE));

        for (java.util.Iterator iterator = peopleList.iterator();
             iterator.hasNext();) {
             Person p = (Person)iterator.next();
             person.put(p.getKey(), p);
        }

        // now run the entry processor to display where they are!
        person.invokeAll(AlwaysFilter.INSTANCE,
                         new WhereAreYouProcessor());
    }
}
```

Lab 9 Solutions – Using JPA with Coherence

1. Example 1 – annotated Employees

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

import java.io.Serializable;

import java.sql.Timestamp;

import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.NamedQuery;
import javax.persistence.OneToMany;

@Entity
@NamedQuery(name = "Employees.findAll", query = "select o from Employees o")
public class Employees implements Serializable {
    @Column(name="COMMISSION_PCT")
    private Double commissionPct;
    @Column(name="DEPARTMENT_ID")
    private Long departmentId;
    @Column(nullable = false)
    private String email;
    @Id
    @Column(name="EMPLOYEE_ID", nullable = false)
    private Long employeeId;
    @Column(name="FIRST_NAME")
    private String firstName;
    @Column(name="HIRE_DATE", nullable = false)
    private Timestamp hireDate;
    @Column(name="JOB_ID", nullable = false)
    private String jobId;
    @Column(name="LAST_NAME", nullable = false)
    private String lastName;
    @Column(name="PHONE_NUMBER")
    private String phoneNumber;
    private Double salary;
    @ManyToOne
    @JoinColumn(name = "MANAGER_ID", referencedColumnName = "EMPLOYEE_ID")
    private Employees employees;
    @OneToMany(mappedBy = "employees")
    private List<Employees> employeesList;

    public Employees() {
    }

    public Double getCommissionPct() {
        return commissionPct;
    }

    public void setCommissionPct(Double commissionPct) {
        this.commissionPct = commissionPct;
    }

    public Long getDepartmentId() {
        return departmentId;
    }

    public void setDepartmentId(Long departmentId) {
        this.departmentId = departmentId;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public Long getEmployeeId() {
        return employeeId;
    }

    public void setEmployeeId(Long employeeId) {
        this.employeeId = employeeId;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public Timestamp getHireDate() {
        return hireDate;
    }

    public void setHireDate(Timestamp hireDate) {
        this.hireDate = hireDate;
    }

    public String getJobId() {
        return jobId;
    }

    public void setJobId(String jobId) {
        this.jobId = jobId;
    }

    public String getLastName() {
        return lastName;
    }

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


    public String getPhoneNumber() {
        return phoneNumber;
    }

    public void setPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }
    public Double getSalary() {
        return salary;
    }

    public void setSalary(Double salary) {
        this.salary = salary;
    }

    public Employees getEmployees() {
        return employees;
    }

    public void setEmployees(Employees employees) {
        this.employees = employees;
    }

    public List<Employees> getEmployeesList() {
        return employeesList;
    }

    public void setEmployeesList(List<Employees> employeesList) {
        this.employeesList = employeesList;
    }

    public Employees addEmployees(Employees employees) {
        getEmployeesList().add(employees);
        employees.setEmployees(this);
        return employees;
    }

    public Employees removeEmployees(Employees employees) {
        getEmployeesList().remove(employees);
        employees.setEmployees(null);
        return employees;
    }
}
```

Example 2 – RunEmployeeExample

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

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

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() * 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() );
    }
}
```