Pentaho ETL integration

Pentaho ETL to Ehcache / Terracotta integration

The following document outlines all steps required to integrate Pentaho ETL (Kettle) with Terracotta and/or Ehcache.

The plugin was developed as an open source plugin for Kettle and can be found online: https://github.com/sgrotz/PentahoEhCachePlugin/

Table of contents:

Additional details:

This plugin can be used within Pentaho Data Integration aka. Kettle (PDI) to access an Ehcache / Terracotta instance.

Please note, that this plugin is still in ALPHA stage - it has been tested to work well for simple key/value pairs. It is not yet considered stable, but "usable" :) ... Please use with caution.

Installation:

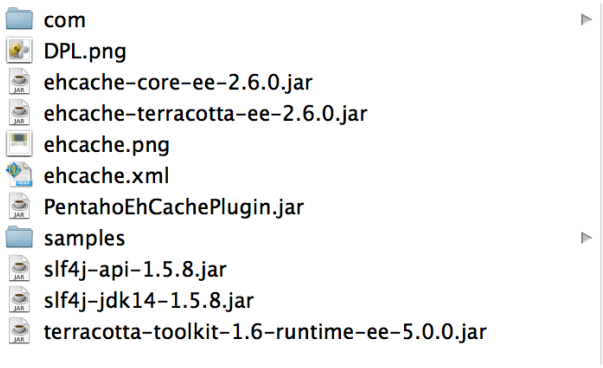
To install the plugin, please make sure that you:

- Install and start your Terracotta Server (only applies if you want to distribute your cache through Terracotta!)
- Download & Install Kettle/PDI from here: http://www.pentaho.de/download/
- Download & Unzip the ehCachePlugin_vXX.zip from https://github.com/sgrotz/PentahoEhCachePlugin/blob/master/ehCachePlugin_latest.zip
- Place the unpacked content into the <PDI directory>/plugins/steps/ehCachePlugin folder
- Copy the terracotta-license.key into the classpath (PDI root directory works fine)

Configuration:

The plugin ships automatically with the Terracotta 3.7.0 libraries. This may be outdated at the point, when you start using the plugin. Please make sure that you copy the latest server jar files into the <PDI directory>/plugins/steps/ehCachePlugin folder. You can find the latest libraries in your Terracotta Server root/common as well as the Terracotta Server Root/ehcache/lib folder.

Your local installation <PDI directory>/plugins/steps/ehCachePlugin, should look similar to this:



Within the ehcache.xml, you can add your caches, which you would like to use:

```
<ehcache
    name="TestCache"
    updateCheck="false">
    <diskStore
        path="./cacheStore/testCache">
    </diskStore>

    <defaultCache
        eternal="false"
        maxElementsInMemory="5000"
        overflowToDisk="false">
    </defaultCache>

    <cache
        name="TestCache"
        eternal="true"
        maxBytesLocalHeap="100M"
        overflowToDisk="false"
        maxElementsOnDisk="2000"
        statistics="true">
                <!--  -->
        <terracotta/>
    </cache>


    <terracottaConfig
        url="localhost:9510"
        >
    </terracottaConfig>

</ehcache>
```

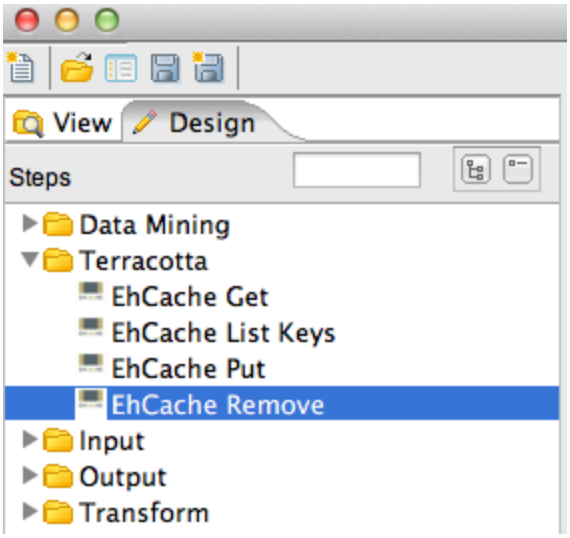
Please make sure that you amend the cache settings as to per your requirements. In my sample, the Terracotta Server is installed locally - you may have to amend this to point to your shared TSA.

Start Kettle / PDI:

- Start Kettle through the spoon.sh and check the startup script for error log entries

> **Hint**
>
> It has been mentioned, that when starting PDI through the executable directly (especially on MacOS), the plugin may not load and execute correctly. If you experience classNotFound errors and/or MethodNotFound errors, then try starting it through the spoon.sh instead - this may fix the problem.

Once you have configured the caches in your ehcache.xml - you are able to start your local kettle/PDI designer. Once started, you can create a new transformation and within the design view, you should see the new plugins:



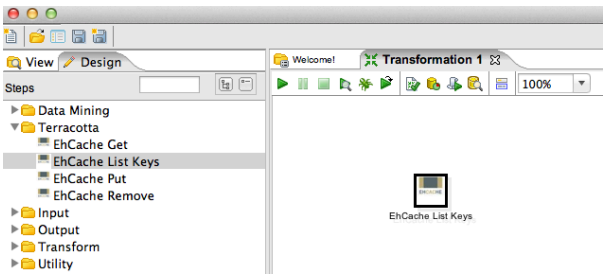
Using the Ehcache Plugins - Overview:

Once you have started PDI and are able to see the plugins, you are able to embed them into your own transformations.

The following services/plugins are available for you:

- **EhCache Get** - Get a key/value from an ehcache instance.
- **EhCache List Keys** - List all keys in a particular ehcache
- **EhCache Put** - Write a new key/value to an ehcache instance.
- **EhCache Remove** - Remove a key/value pair from an ehcache instance.

Simply drag & drop the plugin into your transformation and double click the plugin for additional inputs.



Using the EhCache Put plugin

First of all, we will use the EhCache Put plugin, which will allow us to load data into the cache. Please note that the current version only supports native key/values - and no POJO's. If you want to embed POJO's, please take a look at the advanced section.

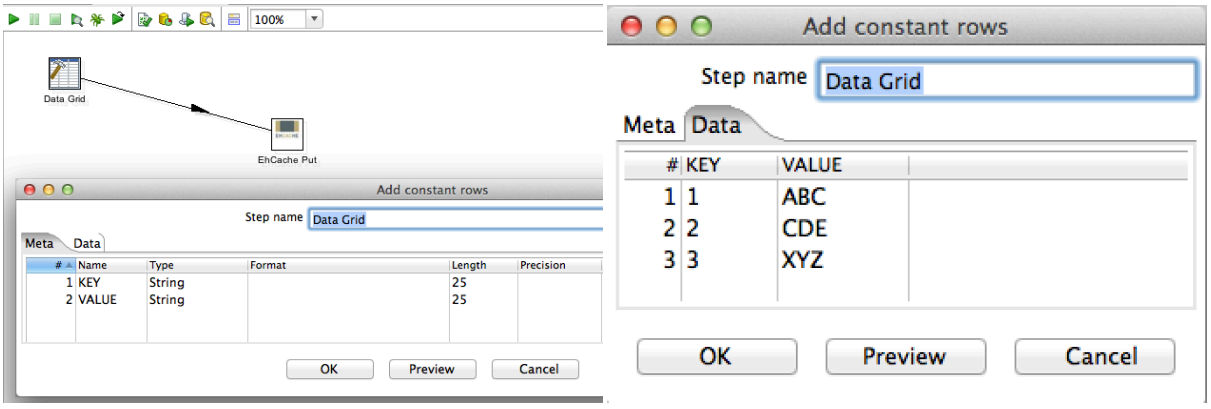
In order to load data to the Ehcache, first drag and drop your datasource into the transformation. For this example we use a simple datagrid step. (It doesnt matter which source you use, as long as you convert the output to a key/value pair)

Input:

Field Name	Mandatory	Null	Format	Comment
KEY	true	false	String	The key for your object.
VALUE	true	true	String	The value (content) of your object.

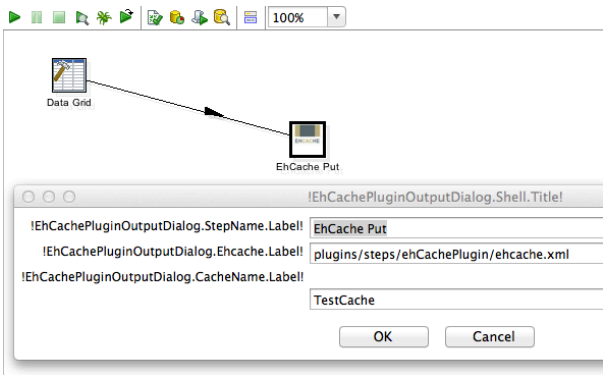
Output:

Field Name	Mandatory	Null	Format	Comment
No output				



Important: *The EhCache Put step only accepts a KEY and a VALUE column. Please make sure that you name it accordingly as an output from your data-source.*

Now double-click on the EhCache Put step, to enter your cache details.



The step name can be chosen freely. It will be displayed in your transformation accordingly. Make sure you point to the correct ehcache.xml, in which the configuration can be found. Last but not least - make sure to specify the correct cachename, to which to load the data. The cachename has to be identical with the value specified in your ehcache.xml file.

Now you can run the transformation - make sure to set the logging to debug level, so that you can see potential error messages for the first run.

Run report - Logging

```
2013/02/26 16:53:58 - Data Grid.0 - Running on slave server #0/1.
2013/02/26 16:53:58 - EhCache Put.0 - *** Using Cache: TestCache from configuration
file: plugins/steps/ehCachePlugin/ehcache.xml
2013/02/26 16:54:53 - EhCache Put.0 - Running on slave server #0/1.
2013/02/26 16:54:53 - EhcachePut - Step [Data Grid.0] initialized flawlessly.
2013/02/26 16:54:53 - EhcachePut - Step [EhCache Put.0] initialized flawlessly.
2013/02/26 16:54:53 - EhcachePut - Transformation has allocated 2 threads and 1
rowsets.
2013/02/26 16:54:53 - Data Grid.0 - Starting to run...
2013/02/26 16:54:53 - EhCache Put.0 - Starting to run...
2013/02/26 16:54:53 - Data Grid.0 - Signaling 'output done' to 1 output rowsets.
2013/02/26 16:54:53 - Data Grid.0 - Finished processing (I=0, O=0, R=0, W=3, U=0,
E=0)
2013/02/26 16:54:53 - EhCache Put.0 - *** This is the first record ...
2013/02/26 16:54:53 - EhCache Put.0 - *** Writing new Element: 1 with value ABC
2013/02/26 16:54:53 - EhCache Put.0 - *** Writing new Element: 2 with value CDE
2013/02/26 16:54:53 - EhCache Put.0 - *** Writing new Element: 3 with value XYZ
2013/02/26 16:54:53 - EhCache Put.0 - *** There is no more input ... Nothing to add
to the cache! :(
2013/02/26 16:54:53 - EhCache Put.0 - Signaling 'output done' to 0 output rowsets.
2013/02/26 16:54:53 - EhCache Put.0 - Finished processing (I=0, O=0, R=3, W=3, U=0,
E=0)
2013/02/26 16:54:53 - Spoon - The transformation has finished!!
```

Success! The 3 records were successfully loaded into the testcache.

Using the EhCache List Keys plugin

Once the data is in the cache, lets make sure that the data can be found. Please note that the current version only supports native key/values - and no POJO's. If you want to embed POJO's, please take a look at the advanced section.

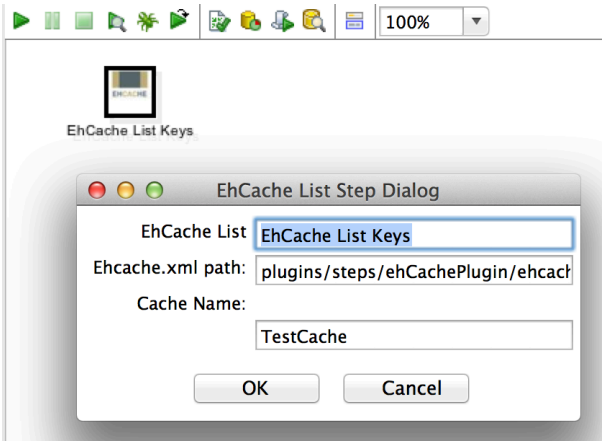
Input:

Field Name	Mandatory	Null	Format	Comment
No input accepted				

Output:

Field Name	Mandatory	Null	Format	Comment
KEYS	true	false	String	Returns a list of keys, which are stored in the cache

Drag & Drop the list-keys plugin into your transformation and double click it to see the advanced settings.



The step name can be chosen freely. It will be displayed in your transformation accordingly. Make sure you point to the correct ehcache.xml, in which the configuration can be found. Last but not least - make sure to specify the correct cachename. The cachename has to be identical with the value specified in your ehcache.xml file.

Now you can run the transformation - make sure to set the logging to debug level, so that you can see potential error messages for the first run.

Run report - Logging

```
2013/02/26 17:06:52 - EhCache List Keys.0 - *** Using Cache: TestCache from
configuration file: plugins/steps/ehCachePlugin/ehcache.xml
2013/02/26 17:06:52 - EhCache List Keys.0 - Running on slave server #0/1.
2013/02/26 17:06:52 - EhCacheList - Step [EhCache List Keys.0] initialized
flawlessly.
2013/02/26 17:06:52 - EhCacheList - Transformation has allocated 1 threads and 0
rowsets.
2013/02/26 17:06:52 - EhCache List Keys.0 - Starting to run...
2013/02/26 17:06:52 - EhCache List Keys.0 - *** This is the first record ...
2013/02/26 17:06:53 - EhCache List Keys.0 - *** Found 3 Elements: [2, 3, 1]
2013/02/26 17:06:53 - EhCache List Keys.0 - *** Adding Key 2 to the output list ...
2013/02/26 17:06:53 - EhCache List Keys.0 - *** Adding Key 3 to the output list ...
2013/02/26 17:06:53 - EhCache List Keys.0 - *** Adding Key 1 to the output list ...
2013/02/26 17:06:53 - EhCache List Keys.0 - Signaling 'output done' to 0 output
rowsets.
2013/02/26 17:06:53 - EhCache List Keys.0 - Finished processing (I=0, O=0, R=0, W=6,
U=0, E=0)
2013/02/26 17:06:53 - Spoon - The transformation has finished!!
```

Success! The 3 records were successfully listed. You can parse the output of the step to additional further steps.

Using the EhCache Get plugin

Once the data is in the cache, lets make sure that the data can be found. Please note that the current version only supports native key/values - and no POJO's. If you want to embed POJO's, please take a look at the advanced section.

Input:

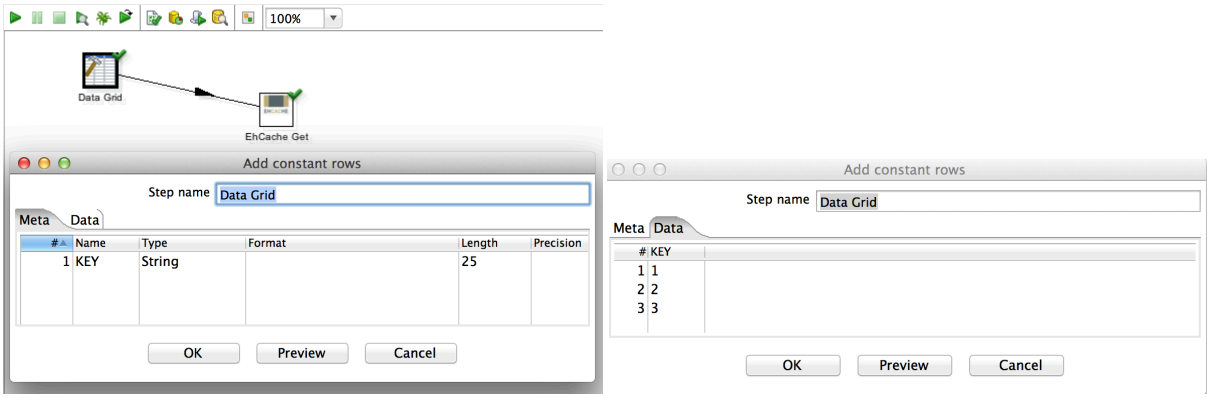
Field Name	Mandatory	Null	Format	Comment
KEY	true	false	String	Supply the key value to get from the cache...

Output:

Field Name	Mandatory	Null	Format	Comment
KEY	true	false	String	The key for your object.

VALUE	true	true	String	The value (content) of your object.

Drag & Drop the data grid (copy it from the previous step) as well as the EhCache Get plugin into your transformation and double click it to see the advanced settings.



The step name can be chosen freely. It will be displayed in your transformation accordingly. Make sure you point to the correct ehcache.xml, in which the configuration can be found. Last but not least - make sure to specify the correct cachename. The cachename has to be identical with the value specified in your ehcache.xml file.

Now you can run the transformation - make sure to set the logging to debug level, so that you can see potential error messages for the first run.

Run report - Logging

```
2013/02/26 17:18:23 - Data Grid.0 - Running on slave server #0/1.
2013/02/26 17:18:23 - EhCache Get.0 - *** Using Cache: TestCache from configuration
file: plugins/steps/ehCachePlugin/ehcache.xml
2013/02/26 17:18:23 - EhCache Get.0 - Running on slave server #0/1.
2013/02/26 17:18:23 - EhCacheGet - Step [Data Grid.0] initialized flawlessly.
2013/02/26 17:18:23 - EhCacheGet - Step [EhCache Get.0] initialized flawlessly.
2013/02/26 17:18:23 - EhCacheGet - Transformation has allocated 2 threads and 1
rowsets.
2013/02/26 17:18:23 - Data Grid.0 - Starting to run...
2013/02/26 17:18:23 - EhCache Get.0 - Starting to run...
2013/02/26 17:18:23 - Data Grid.0 - Signaling 'output done' to 1 output rowsets.
2013/02/26 17:18:23 - Data Grid.0 - Finished processing (I=0, O=0, R=0, W=3, U=0,
E=0)
2013/02/26 17:18:23 - EhCache Get.0 - *** This is the first record ...
2013/02/26 17:18:23 - EhCache Get.0 - *** Getting Element 1 from the cache ...
2013/02/26 17:18:23 - EhCache Get.0 - *** Getting Element 2 from the cache ...
2013/02/26 17:18:23 - EhCache Get.0 - *** Getting Element 3 from the cache ...
2013/02/26 17:18:23 - EhCache Get.0 - *** There is no more input ... Nothing to get
from the cache! :(
2013/02/26 17:18:23 - EhCache Get.0 - Signaling 'output done' to 0 output rowsets.
2013/02/26 17:18:23 - EhCache Get.0 - Finished processing (I=0, O=0, R=3, W=6, U=0,
E=0)
2013/02/26 17:18:23 - Spoon - The transformation has finished!!
```

Success! The 3 records were successfully loaded. You can parse the elements now to additional further steps.

Using the EhCache Remove plugin

Once the data is in the cache, lets make sure that the data can be found. Please note that the current version only supports native key/values - and no POJO's. If you want to embed POJO's, please take a look at the advanced section.

Input:

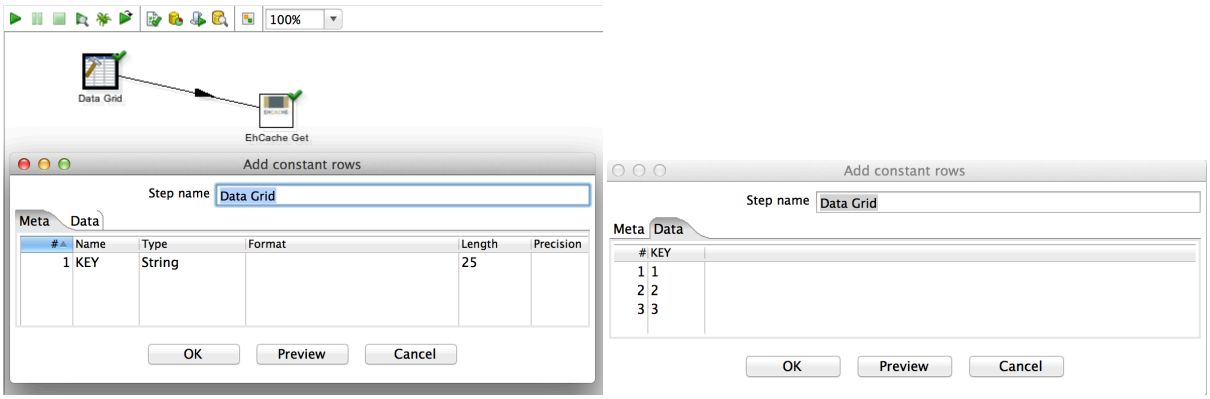
Field Name	Mandatory	Null	Format	Comment
KEY	true	false	String	Supply the key value to remove from the cache...

Output:

Field Name	Mandatory	Null	Format	Comment

No output				

Drag & Drop the data grid (copy it from the previous step) as well as the EhCache Remove plugin into your transformation and double click it to see the advanced settings.



The step name can be chosen freely. It will be displayed in your transformation accordingly. Make sure you point to the correct ehcache.xml, in which the configuration can be found. Last but not least - make sure to specify the correct cachename. The cachename has to be identical with the value specified in your ehcache.xml file.

Now you can run the transformation - make sure to set the logging to debug level, so that you can see potential error messages for the first run.

Run report - Logging

```
2013/02/26 17:24:12 - Data Grid.0 - Starting to run...
2013/02/26 17:24:12 - EhCache Remove.0 - Starting to run...
2013/02/26 17:24:12 - Data Grid.0 - Signaling 'output done' to 1 output rowsets.
2013/02/26 17:24:12 - Data Grid.0 - Finished processing (I=0, O=0, R=0, W=3, U=0,
E=0)
2013/02/26 17:24:12 - EhCache Remove.0 - *** This is the first record ...
2013/02/26 17:24:12 - EhCache Remove.0 - *** Removing Element 1 from the cache ...
2013/02/26 17:24:12 - EhCache Remove.0 - *** Removing Element 2 from the cache ...
2013/02/26 17:24:12 - EhCache Remove.0 - *** Removing Element 3 from the cache ...
2013/02/26 17:24:12 - EhCache Remove.0 - *** There is no more input ... Nothing more
to delete from the cache! :(
2013/02/26 17:24:12 - EhCache Remove.0 - Signaling 'output done' to 0 output
rowsets.
2013/02/26 17:24:12 - EhCache Remove.0 - Finished processing (I=0, O=0, R=3, W=3,
U=0, E=0)
2013/02/26 17:24:12 - Spoon - The transformation has finished!!
```

Success! The 3 records were successfully removed from the cache.

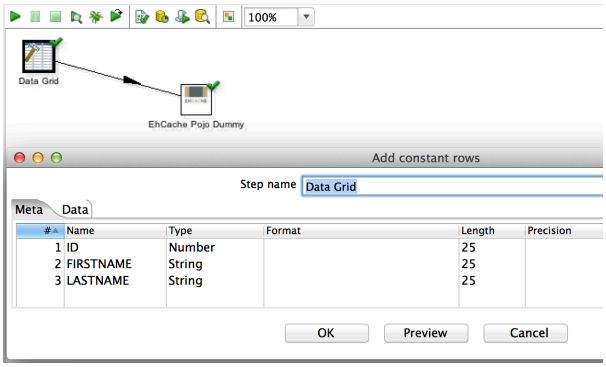
Using the EhCache plugin to load / retrieve Java Objects

The first section of this document focused mainly on the integration from PDI to Ehcache using simplel key/value maps. In this section we will take a closer look at how we can also embed pojo's into Terracotta. This is particularly useful to make use of all of Terracotta's features, such as indexing, search etc.

Saying this however, it is important to understand that working with java objects is not as straight forward as using the key/value plugins. They require custom development and may be more advanced - but once created, they remain rather static and can be used for a sustained period of time.

Write sample objects to the cache

Take a look at the sample plugin service called EhCache POJO Dummy - this is a sample service, which is loading a java pojo into the testcache. It is a very simplistic example, but I am sure you get the drift



We will load the pojo with the required variables, ID + FIRSTNAME + LASTNAME.

Kettle will map the values into a pojo and put it into the cache. The pojo object looks similar to this:

Object definition

```
package com.ehcache.pentaho.dummyPojo;

import java.io.Serializable;


public class SampleObject implements Serializable {

 private static final long serialVersionUID = 277567342978497425L;
 public Double ID;
 public String firstName;
 public String lastName;



 // Default constructor
 public SampleObject() {
 }


 public Double getID() {
  return ID;
 }


 public void setID(Double iD) {
  this.ID = iD;
 }


 public String getFirstName() {
  return firstName;
 }


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


 public String getLastName() {
  return lastName;
 }


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


}
```

Within the sample plugin - take a look at the EhCachePluginDummyPojo.java, which contains the following mapping:

Object definition

```
...
  // ###
  // ### STARTING FROM HERE, ADD YOUR OWN POJO IMPLEMENTATION
  // ### Make sure to place your POJO definition in the classpath, reference your
object and map it into
  // ### the ELEMENT. With this, you can also define searchable attributes.
  // ### The following sample uses the SampleObject class.
  // ###



  /*
   * The following code is just a sample - how an object can be written into the
cache
   *
   */

  SampleObject myObject = new SampleObject();

  // Map the ID, firstname and lastname from the input row into your object ...
  myObject.setID(data.outputRowMeta.getNumber(obj,
data.outputRowMeta.indexOfValue("ID")));
  myObject.setFirstName(data.outputRowMeta.getString(obj,
data.outputRowMeta.indexOfValue("FIRSTNAME")));
  myObject.setLastName(data.outputRowMeta.getString(obj,
data.outputRowMeta.indexOfValue("LASTNAME")));


  if ((myObject.getID() != null)) {
   cache = manager.getCache(meta.getCacheName());


   // Add the elements to the cache...
   logDebug("*** Writing new SampleObject " + myObject.getID().toString() + " ...");

   cache.put(new Element(myObject.getID(), myObject));

   if (checkFeedback(linesRead)) logBasic("Linenr "+linesRead);

   incrementLinesWritten();
   return true;

  } else {
   // Throw an error when the ID or Value was null
   logError("*** KEY or VALUE field was null ...");
   return false;
  }


  // ###
  // ### POJO IMPLEMENTATION END
  // ###
```

If you execute the transformation - you should see a similar result to this:

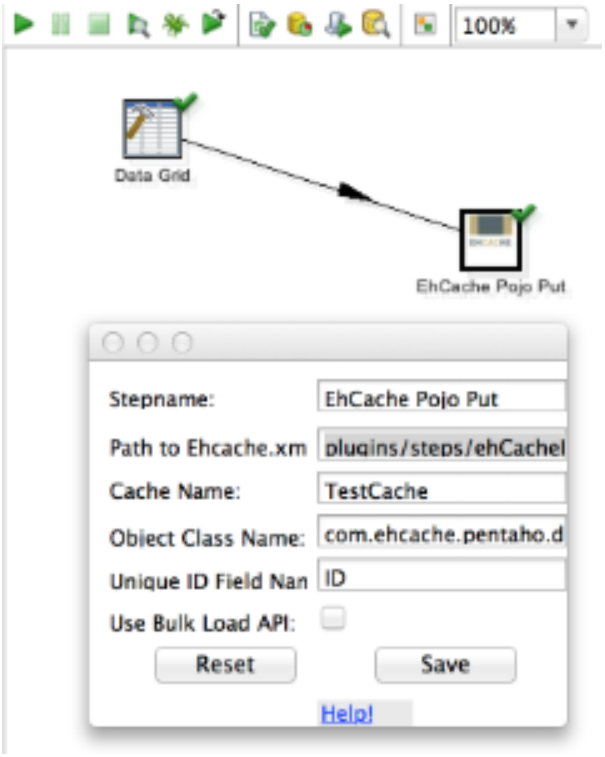
Run report - Logging

```
2013/02/26 21:27:21 - Data Grid.0 - Running on slave server #0/1.
2013/02/26 21:27:21 - EhCache Pojo Dummy.0 - *** Using Cache: TestCache from
configuration file: plugins/steps/ehCachePlugin/ehcache.xml
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - Running on slave server #0/1.
2013/02/26 21:27:25 - EhCachePutSampleObject - Step [Data Grid.0] initialized
flawlessly.
2013/02/26 21:27:25 - EhCachePutSampleObject - Step [EhCache Pojo Dummy.0]
initialized flawlessly.
2013/02/26 21:27:25 - Data Grid.0 - Starting to run...
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - Starting to run...
2013/02/26 21:27:25 - EhCachePutSampleObject - Transformation has allocated 2
threads and 1 rowsets.
2013/02/26 21:27:25 - Data Grid.0 - Signaling 'output done' to 1 output rowsets.
2013/02/26 21:27:25 - Data Grid.0 - Finished processing (I=0, O=0, R=0, W=3, U=0,
E=0)
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - *** This is the first record ...
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - *** Writing new SampleObject 1.0 ...
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - *** Writing new SampleObject 2.0 ...
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - *** Writing new SampleObject 3.0 ...
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - *** There is no more input ... Nothing
to add to the cache! :(
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - Signaling 'output done' to 0 output
rowsets.
2013/02/26 21:27:25 - EhCache Pojo Dummy.0 - Finished processing (I=0, O=0, R=3,
W=3, U=0, E=0)
2013/02/26 21:27:25 - Spoon - The transformation has finished!!
```

The service has finished successfully - 3 sample objects were loaded into the cache. You can now enable searchable fields within your ehcache.xml

Reflective Java Object Put Service

Starting with release 0.4 of the ETL Plugin, a new service for reflective pojo putting is available. You can use the service very similarly as the dummy pojo load service, without having to write any line of code.



To do so, drag and drop the Ehcache Pojo Put Service into your transformation and double click the step to enter the details. In addition to the normal fields (path, cache name), you will now also find the object class name, as well as the Unique ID Field. Make sure to set them properly.

Please remember

In order to use the Pojo Putter Service, you need to ensure that:

- The object class name is valid - and the class/jar file is available in your classpath. Make sure to copy it into the <Pentaho>/libext folder.
- All incoming fields from the previous service are named as the setterMethod in the pojo. For example - the incoming field should be "ID" so that the service can call the "setID" method.
- The Unique ID Field Name has to be an **INT/LONG** value - it defines the field in the input stream, which will be used to identify the element in the cache (the key!).
- The service removes underscores "_" from the incoming field names. Make sure that there are no other special characters in there.
- The service ignores null values - if the incoming field is empty or null, it will not call the getter method.

Lets take a look at the sample transformation Ehcache Pojo Put.tkr. If you run the transformation, it will try to load the cache with the sample class (see above for details) with the data from the datagrid service.

Run report - Logging

```
2013/03/28 09:10:56 - Data Grid.0 - Starting to run...
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Starting to run...
2013/03/28 09:10:56 - Data Grid.0 - Signaling 'output done' to 1 output rowsets.
```

```
2013/03/28 09:10:56 - Data Grid.0 - Finished processing (I=0, O=0, R=0, W=3, U=0,
E=0)
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** This is the first record ...
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList: ID
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList:
FIRSTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList:
LASTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Object
com.ehcache.pentaho.dummyPojo.SampleObject has 3 set methods...
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setID will be mapped with data
from Field ID
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting ID value as Number: 1.0
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setFirstName will be mapped
with data from Field FIRSTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting FIRSTNAME value as String:
ABC
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setLastName will be mapped
with data from Field LASTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting LASTNAME value as String: ZZZ
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Finished mapping object
com.ehcache.pentaho.dummyPojo.SampleObject
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Writing new Java Object as element: 1
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList: ID
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList:
FIRSTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList:
LASTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Object
com.ehcache.pentaho.dummyPojo.SampleObject has 3 set methods...
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setID will be mapped with data
from Field ID
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting ID value as Number: 2.0
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setFirstName will be mapped
with data from Field FIRSTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting FIRSTNAME value as String:
CDE
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setLastName will be mapped
with data from Field LASTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting LASTNAME value as String: UUU
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Finished mapping object
com.ehcache.pentaho.dummyPojo.SampleObject
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Writing new Java Object as element: 2
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList: ID
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList:
FIRSTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Added new fieldname to the fieldList:
LASTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Object
com.ehcache.pentaho.dummyPojo.SampleObject has 3 set methods...
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setID will be mapped with data
from Field ID
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting ID value as Number: 3.0
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setFirstName will be mapped
with data from Field FIRSTNAME
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting FIRSTNAME value as String:
XYZ
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Method setLastName will be mapped
with data from Field LASTNAME
```

```
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Setting LASTNAME value as String: III
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Finished mapping object
com.ehcache.pentaho.dummyPojo.SampleObject
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** Writing new Java Object as element: 3
2013/03/28 09:10:56 - EhCache Pojo Put.0 - *** There is no more input ... Nothing to
add to the cache! :(
2013/03/28 09:10:56 - EhCache Pojo Put.0 - Signaling 'output done' to 0 output
rowsets.
2013/03/28 09:11:01 - EhCache Pojo Put.0 - Finished processing (I=0, O=0, R=3, W=3,
U=0, E=0)
2013/03/28 09:11:01 - Spoon - The transformation has finished!!
```

```
2013/03/28 09:14:51 -
com.ehcache.pentaho.pojo.output.EhCachePluginPojoOutputMeta@7115e8a - *** CacheName
is: TestCache xml URL is: plugins/steps/ehCachePlugin/ehcache.xml
```

As you can see in the output, the service will first query all available setter methods in the java class. In the next step, it will try to map all incoming data to the available setter methods. Thats it! Your transformation ran successfully and now all your objects are in the cache.

Summary:

Use the dummy sample code above as your baseline if the default pojo putter does not work. You can clone and adjust the code to reflect your designated pojo implementation. Simply copy the plugin, adjust the objects and make sure the meta-information is updated. Redeploy the new plugin into the plugins/steps folder and test it. Alternatively try using the pojo putter service, which simplifies the integration severely.

As an additional sidenote - make sure to take a look at hibernate tools. With that tool you can easily create java classes out of your existing database schema.

It may sound easier than it actually is - but hey: May the force be with you ... If you have any additional questions, please drop me an email on stephan@terracottatech.com