IBM Cúram Social Program Management

Cúram Cache

Version 6.0.4

Note

Before using this information and the product it supports, read the information in Notices at the back of this guide.

This edition applies to version 6.0.4 of IBM Cúram Social Program Management and all subsequent releases and modifications unless otherwise indicated in new editions.

Table of Contents

Chapter 1

Introduction

1.1 Purpose

The aim of this document is to introduce Cúram Cache, a generic caching service designed to satisfy the requirement for short and long lived caches in the application.

1.2 Audience

This guide is intended for architects and developers interested in using Cúram Cache to satisfy their caching requirements.

Chapter 2

Cúram Cache

2.1 What is Cúram Cache

Cúram Cache is a generic caching service that is designed to satisfy the requirement for short and long lived caches in the application. The service is available in both the client and server containers of an application server environment (online application) as well as in a standard standalone Java™ process (batch programs).

Cúram Cache allows the creation of three types of caches:

- Global - these are global (at JVM level) caches.
- Thread local - these are caches that live as long as the thread that owns them.
- Transaction local - these are caches that exist for the duration of the current transaction.

The last two type of caches are collectively referred to in this document as multi-instance caches because at any given moment there could be more than one instance of a cache with a given name (one for each active transaction or thread).

2.2 Configuration

The configuration of all types of caches in Cúram Cache is entirely declarative and it is based on the configuration mechanism provided by the application. Cache configuration parameters must be added to the APP_CACHE property section.

In the current implementation, global caches support both size and time based eviction policies while the multi-instance caches have support only for time based eviction policy.

The following cache configuration parameters can be adjusted:

- Size – the maximum number of elements in memory. The default value is 200. The type is `INT32` .
- Eviction policy – the policy used for evicting items from memory when the maximum number of elements in memory is reached. The default value is `LRU` . The type is `STRING` . Valid values are:
  - `LRU` – least recently used
  - `LFU` – least frequently used
  - `FIFO` – first in first out
- Time to live - the number of seconds an item is allowed to live in a cache. When this is set to a non-zero positive value, any items that have been in the cache for longer than the value of this parameter, in seconds, are discarded.
- Time to idle - the number of seconds an item in the cache is allowed to be unused before being discarded. When this is set to a non-zero positive value, any items that have been unused for more than the value of this parameter, in seconds, are discarded.

All cache configuration properties must conform to this notation:

`curam.cache.<cache_group_name>.<cache_name>.<parameter>`

where:

- `<cache_group_name>` - is the name of the cache group the cache belongs to.
- `<cache_name>` - is the name of the cache. This could also be "*" and then, the configuration parameter is applied to all caches in this cache group.
- `<parameter>` - can be `size` , `evictionPolicy, timeToIdle or timeToLive` .

In the example below, the global cache `curam.myproject.mycache` in the default global cache group `curam-group` is configured with a size of 1000 items and an eviction policy of Least Recently Used.

```
curam.cache.curam-group.curam.myproject.mycache.size=1000
curam.cache.curam-group.curam.myproject.mycache.evictionPolicy=LRU
```

Example 2.1 Configuring a cache

In this second example, the transaction local cache `curam.myproject.mycache` in the transaction local cache group `transaction-group` is configured with a time to idle of 10 seconds while all other transaction local caches are configured with a value of 5

seconds.

```
curam.cache.transaction-group.curam.myproject.mycache
                                              .timeToIdle=10
curam.cache.transaction-group.*.timeToIdle=5
```

Example 2.2 Configuring all caches in a group

Cache configuration data stored in the application configuration repository can be overridden by passing the relevant values as JVM system properties. This might be of interest for batch processes where the application profile might be different than the online application.

The example below shows how to disable the global cache `curam.myproject.mycache` in the default global cache group for a batch process.

```
ant -f app_batchlauncher.xml
     -Dcuram.cache.curam-group.curam.myproject.mycache.size=0
     -Dbatch.userna…
```

Example 2.3 Disabling a cache for a batch process

2.3 Statistics

All caches in Cúram Cache are instrumented for statistics and these are integrated with the Cúram JMX infrastructure. The following minimum set of statistics are exposed by each type of cache via the `CuramCacheStats` MBean:

- Cache group - the name of the cache group
- Cache – the name of the cache
- Layer – the name of the cache layer (memory, disk,...)
- Size – the number of items in the cache
- Hits - the number of requests to the cache that returned an item already loaded in the cache
- Misses - the number of requests to the cache that returned an item which had to be loaded in the cache
- Evictions - the number of times items that have been evicted from the cache
- Average get time(ns) - the average elapsed time, in nanoseconds, that takes for an item to be read from the cache. Note that some cache providers might only support millisecond resolution.

Multi-instance caches offer snapshot and aggregated statistics. Snapshot statistics are for all instances alive at the moment of the query and aggregated statistics are calculated from all instances that have been created.

2.4 Shutting Down Cúram Cache

Cúram Cache requires orderly shutdown on JVM exit . Cúram Cache installs automatically a JVM shut down hook to clear the cache as the last resort solution but it is recommended, where possible, the use of the explicit shutdown by invoking `CacheManager.shutdown()` when the application is shutdown.

Chapter 3

Global Caches

3.1 Introduction

Global caches are caches that exist in the scope of the JVM process or beyond. In the current version of Cúram Cache, global caches exist only in the scope of the JVM process. An entry stored in a global cache lives across transaction boundaries until it is removed explicitly, by the developer, or implicitly, as a result of the eviction policy associated with the cache.

It is important to note that because global caches are long lived, their data is prone to short periods of inconsistency when cached objects are updated. When an update is made in the application that affects a cached object, the associated cache entry is invalidated asynchronously. The caching infrastructure guarantees that the cache entry is, eventually, invalidated but it cannot guarantee a certain maximum time frame. Understanding this behavior is very important when deciding if certain application data can be cached in a global cache.

3.2 Global Cache Provider

Cúram Cache implements large parts of the global caching infrastructure using third party caching solutions which are referred to in this document as caching providers. The default provider is Ehcache, an open source, high performance, distributed caching infrastructure.

3.3 Default Global Cache Group

Global caches are grouped together based on common configuration requirements such as replication and disk storage. All caches in the application should be created in the default cache group. The name of the default cache group is `curam-group` .

In the current implementation, the default cache group is not self-replicating and does not support disk overflow and disk persistence. Because self-replication is disabled, the cache operations are only visible to the JVM where the global cache is located. However to keep all caches in the default cache group consistent throughout the application server cluster, an explicit cache invalidation mechanism is provided. The cache invalidation can only be triggered from the server code but it invalidates caches in both the server and the client containers across all JVMs in the application server cluster.

3.4 Global Caches

Global caches are created with a call to the get() method of the cache group. If a cache does not exist already, a cache is created and configuration data, if this exists, is applied to it before being returned. Global caches are usually populated using cache loaders registered by cache clients. This approach isolates the cache client from the management of concurrent access to the cache while the cache is loading.

Cúram Cache does not enforce the use of serializable objects in its API, however certain features offered by the caching infrastructure are only available if the key or the cached object are serializable. For this reason, it is recommended that, whenever possible, serializable keys and values should be used in Cúram Cache.

Usage of non-serializable keys

Cache entries that have non-serializable keys are only invalidated on the local JVM and not throughout the application server cluster.

3.4.1 Usage Recommendations

The following is a list of recommendations on how a global cache should be used:

- Only cache immutable objects.
- Use serializable keys and values whenever possible. At the very least the keys should be serializable.
- Use a cache loader to populate the cache. This allows the cache to take advantage of the fine grain concurrency optimizations built into the cache provider and it does not require the user to be concerned with managing concurrent access to the cache.
- The loading of a cache without a loader (using `get()` and `put()` calls) must be avoided for two main reasons:
  - Concurrency management - In this case the user is responsible for managing concurrent access to the cache while the cache is loading. The user has two choices:

- Control concurrent access to the `get()` and `put()` block of code – this approach is not recommended in a performance sensitive part of the application but it offers the guarantee that on object is only loaded once.
- Allow concurrent access to the `get()` and `put()` block of code – this approach supports higher concurrency but an object might be loaded more than once by different threads.

- Efficient data management - without a cache loader, the cache must be pre-populated with all data. With a cache loader, only required data is pulled into the cache.

- Use cache names that are prefixed with a package name unique to your project. For instance `curam.cpm.myCache` would be a suitable name for a cache in the Curam Provider Management™ project.

3.4.2 Configuration

All global caches inherit the following default values for the configuration parameters.

- `size` - 200
- `evictionPolicy` - LRU
- `timeToLive` - 0 (not active)
- `timeToIdle` - 0 (not active)

These values can be overridden for any global cache. In the examples below `curam-group` is the name of the default cache group and `curam.myproject.mycache` is the name of the cache.

```
curam.cache.curam-group.curam.myproject.mycache.size=1000
curam.cache.curam-group.curam.myproject.mycache.evictionPolicy=LRU
curam.cache.curam-group.curam.myproject.mycache.timeToLive=3600
curam.cache.curam-group.curam.myproject.mycache.timeToIdle=300
```

Example 3.1 Configuring a global cache

3.4.3 Using Global Caches in a Transactional Context

When a global cache is used in a transactional context, care must be taken to ensure that the cache maintains its consistency in case the current transaction is rolled back.

Cache invalidation in a transactional context

When modifying data that affects the content of a cache do not remove or update the cached element directly; instead invoke the `Ca-`

cheManagerEjb.postInvalidationMessage() method to post an invalidation message that will trigger the cache invalidation.

3.4.4 Code Samples

This section contains code samples that show how to write a cache loader, how to use the cache and how to invalidate a cache entry.

Cache Loader

In the example below, the CacheLoaderAdapter class is used to help in the implementation of MyCacheLoader.

```
...
public class MyCacheLoader extends
    CacheLoaderAdapter<Integer, ReadWorkQueueDetails> {
  /* (non-Javadoc)
   * @see curam.util.cache.CacheLoader#load(java.lang.Object)
   */
  public ReadWorkQueueDetails load(Integer workQueueID)
      throws AppException, InformationalException {
    WorkAllocation wa = (WorkAllocation)WorkAllocationFactory
                                    .newInstance();
    ReadWorkQueueKey key = new ReadWorkQueueKey();
    key.key = new ReadWorkQueueKey();
    key.key.key = new WorkQueueKey();
    key.key.key.workQueueID = workQueueID;
    ReadWorkQueueDetails item = wa.readWorkQueue(key);
    if(item != null) {
      return item.dtls;
    }
    return null;
  }
}
...
```

Example 3.2 Using CacheLoaderAdapter to implement a cache loader

Cache Client

The example below shows the usual way of registering a cache loader and using the cache.

```
...
public class MyCacheClient {
  // keep a static reference to mycache
  private static Cache<Integer,
            ReadWorkQueueDetails> myCache;

  static {
    // retrieve a reference to mycache and register
    // the cache loader
    myCache = CacheManager.getDefaultCacheGroup()
                                .getCache("mycache");
    myCache.registerCacheLoader(new MyCacheLoader());
  }

  public WorkAllocation() {
...
```

```
  }
...
  // use the cache
  ReadWorkQueueDetails wq = myCache.get(1);
...
```

Example 3.3 Registering a cache loader and using the cache

Cache Invalidation

The example below shows how to invalidate a cache entry in code running in a transactional context (server code). As explained in Default Global Cache Group , cache invalidation for global caches in the default cache group can only be triggered by server code.

```
...
CacheManagerEjb.postInvalidationMessage(
          new CacheInvalidationMessage<Key>("mycache", 1));
...
```

Example 3.4 Invalidating a cache entry in server code

Chapter 4

Thread Local Caches

4.1 Overview

These are caches that are closely tied to the thread used to create them. No other thread can access data in these caches and caches are only destroyed when the thread that created them is terminated. Thread local caches are very specialized. They must only be used for small caches where the overhead of multi-threaded access control that exists for global cache cannot be tolerated.

4.2 Configuration

Thread local caches support only a time based eviction policy. The only two configuration parameters that can be used and their default values are:

- `timeToLive` - 0 (not active)
- `timeToIdle` -0 (not active)

The name of the group for thread local caches is `thread-group` . This name must be used to configure thread local cache as shown in the example below.

```
curam.cache.thread-group.curam.myproject.mycache.timeToLive=60
curam.cache.thread-group.curam.myproject.mycache.timeToIdle=10
```

Example 4.1 Configuring a thread local cache with name **`curam.myproject.mycache`**

4.3 Code Samples

Thread local caches should only by accessed where the correct context

(thread) exists. For instance, it is not recommended to set up a thread local cache in static block of code as that thread might not be the same as the thread using the cache later.

```
public void myMethod() {
  ...
  Cache<String, String> threadCache = CacheManager.
           getThreadLocalCacheGroup().getCache("mycache");
  String value = threadCache.get("key");
  if(value == null) {
    // perform expensive operation to calculate value - this
    // processing only happens once for each thread
    ...
    // and store the result
    threadCache.put("key", "value");
  }
  ...
}
```

Example 4.2 Setting up and using a thread local cache

Chapter 5

Transaction Local Caches

5.1 Overview

A transaction local cache is a cache that lives only for the duration of the current transaction. This type of cache is only available in the server application.

5.2 Configuration

Transaction local caches support only a time based eviction policy. The only two configuration parameters that can be used and their default values are:

- `timeToLive` - 0 (not active)
- `timeToIdle` -5

The name of the group for transaction local caches is `transaction-group` . This name must be used to configure transaction local caches as shown in the example below.

```
curam.cache.transaction-group.curam.myproject.mycache
                                                .timeToLive=60
curam.cache.transaction-group.curam.myproject.mycache
                                                .timeToIdle=10
```

Example 5.1 Configuring a transaction local cache with name **curam.myproject.mycache**

5.3 Code Samples

Like thread local caches, transaction local caches should only by accessed where the correct context (transaction) exists.

```
public void myMethod() {
  ...
  Cache<String, String> txnCache = CacheManagerEjb.
           getTransactionLocalCacheGroup().getCache("mycache");
  String value = txnCache.get("key");
  if(value == null) {
    // perform expensive operation to calculate value - this
    // processing only happens once per transaction
    ...
    // and store the result
    txnCache.put("key", "value");
  }
  ...
}
```

Example 5.2 Setting up and using a transaction local cache

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing

Legal and Intellectual Property Law.

IBM Japan Ltd.

1623-14, Shimotsuruma, Yamato-shi

Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typograph-

ical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation

Dept F6, Bldg 1

294 Route 100

Somers NY 10589-3216

U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources.

IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products

should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This publication documents intended programming interfaces that allow the customer to write programs to obtain the services of IBM Cúram Social Pogram Management.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/us/en/copytrade.shtml .

Java and all Java-based trademarks and logos are registered trademarks of Oracle and/or its affiliates.

Other names may be trademarks of their respective owners. Other company, product, and service names may be trademarks or service marks of others.