



# BEA WebLogic Portal<sup>®</sup>

## Cache Reference

Version 10.00  
Revised: March 2007

# Copyright

Copyright © 1995-2007 BEA Systems, Inc. All Rights Reserved.

## Restricted Rights Legend

This software is protected by copyright, and may be protected by patent laws. No copying or other use of this software is permitted unless you have entered into a license agreement with BEA authorizing such use. This document is protected by copyright and may not be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form, in whole or in part, without prior consent, in writing, from BEA Systems, Inc.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE DOCUMENTATION IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA SYSTEMS DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE DOCUMENT IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

## Trademarks and Service Marks

Copyright © 1995-2007 BEA Systems, Inc. All Rights Reserved. BEA, BEA JRockit, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop, BEA Workshop for WebLogic Platform, BEA WebLogic RFID Mobile SDK, Built on BEA, Jolt, JoltBeans, SteelThread, Top End, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA AquaLogic, BEA AquaLogic Data Services Platform, BEA AquaLogic Enterprise Security, BEA AquaLogic Interaction, BEA AquaLogic Interaction Analytics, BEA AquaLogic Interaction Collaboration, BEA AquaLogic Interaction Integration Services, BEA AquaLogic Interaction Process, BEA AquaLogic Interaction Publisher, BEA AquaLogic Interaction Studio, BEA AquaLogic Service Bus, BEA AquaLogic Service Registry, BEA AquaLogic BPM Designer, BEA AquaLogic BPM Studio, BEA AquaLogic BPM Enterprise Server – Standalone, BEA AquaLogic BPM Enterprise Server – BEA WebLogic, BEA AquaLogic BPM Enterprise Server – IBM WebSphere, BEA AquaLogic BPM Enterprise Server – JBoss, BEA AquaLogic BPM Process Analyzer, BEA AquaLogic Interaction Development Kit, BEA AquaLogic Interaction JSR-168 Consumer, BEA AquaLogic Interaction Identity Service – Active Directory, BEA AquaLogic Interaction Identity Service – LDAP, BEA AquaLogic Interaction Content Service – Microsoft Exchange, BEA AquaLogic Interaction Content Service – Lotus Notes, BEA AquaLogic Interaction Logging Utilities, BEA AquaLogic Interaction WSRP Consumer, BEA AquaLogic Interaction Portlet Framework – Microsoft Excel, BEA AquaLogic Interaction .NET Application Accelerator, AquaLogic Interaction Content Service – Documentum, BEA AquaLogic Interaction Content Service – Windows Files, BEA AquaLogic Interaction Portlet Suite – IMAP, BEA AquaLogic Interaction Portlet Suite – Lotus Notes, BEA AquaLogic Interaction Portlet Suite – Exchange, BEA AquaLogic Interaction Portlet Suite – Documentum, BEA AquaLogic Interaction IDK Extension, BEA AquaLogic HiPer Workspace for BPM, BEA AquaLogic HiPer Workspace for Retail, BEA AquaLogic Sharepoint Console, BEA AquaLogic Commerce Services, BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Kodo, BEA Liquid Data for WebLogic, BEA Manager, BEA MessageQ, BEA SALT, BEA Service Architecture Leveraging Tuxedo, BEA WebLogic Commerce Server, BEA WebLogic Communications Platform, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Enterprise Security, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Java Adapter for Mainframe, BEA WebLogic JDriver, BEA WebLogic Log Central, BEA WebLogic Mobility Server, BEA WebLogic Network Gatekeeper, BEA WebLogic Personalization Server, BEA WebLogic Personal Messaging API, BEA WebLogic Platform, BEA WebLogic Portlets for Groupware Integration, BEA WebLogic Real Time, BEA WebLogic RFID Compliance Express, BEA WebLogic RFID Edge Server, BEA WebLogic RFID Enterprise Server, BEA WebLogic Server Process Edition, WebLogic Server Virtual Edition, WebLogic Liquid Operations Control, BEA LiquidVM, BEA WebLogic SIP Server, BEA WebLogic WorkGroup Edition, BEA Workshop for WebLogic Platform, BEA Workshop for JSF, BEA Workshop for JSP, BEA Workshop for Struts, BEA Workshop Studio, Dev2Dev, Liquid Computing, and Think Liquid are trademarks of BEA Systems, Inc. Accelerated Knowledge Transfer, AKT, BEA Mission Critical Support, BEA Mission Critical Support Continuum, CollabraSuite – BEA Edition, BEA Guardian and BEA SOA Self Assessment are service marks of BEA Systems, Inc.

All other names and marks are property of their respective owners.

# Contents

## WebLogic Portal Cache Reference

- Adding a Cache ..... 1
- Portal Framework Caches ..... 2
- WSRP Caches ..... 5
- Content and Ad Caches ..... 7
- User Management Caches ..... 8
- Campaign and Discount Caches ..... 11
- Commerce Caches ..... 12



# WebLogic Portal Cache Reference

This reference guide lists the available caches for WebLogic Portal that can be managed within the Portal Administration Console.

Caches referenced in this guide can be modified within the Administration Console. Although some caches are not pre-configured within the Administration Console. You can add these caches to the Administration Console.

This book includes the following sections:

- [Adding a Cache](#)
- [Portal Framework Caches](#)
- [WSRP Caches](#)
- [Content and Ad Caches](#)
- [User Management Caches](#)
- [Campaign and Discount Caches](#)
- [Commerce Caches](#)

## Adding a Cache

If you want to use a cache that is not in the list of configured caches, you must add the cache to the Portal Administration Console.

To add a cache:

1. Choose **Configuration Settings > Service Administration**.
2. Select the Cache Manager node in the tree.
3. In the Browse tab, click **Add Cache**.
4. Enter the name of the cache.
5. Optionally, enter or modify the default cache configuration settings.
6. Click **Update**.

The cache you have added appears in the list of caches.

## Portal Framework Caches

**Table 1** portalContentUriCache

<b>Cache</b>	portalContentUriCache
<b>Use</b>	Used to store portal content URIs for a combination of webapp, portal, locale and optional user name.
<b>Key</b>	Key is equal to portal path + name of web application.
<b>Value</b>	Portal content URI
<b>Notes</b>	Set this cache according the number of portals that have associated content URIs. The default values are recommended. Default values: MaxEntries=500; TimeToLive=-1

**Table 2** portalLocalizationLocaleCache

<b>Cache</b>	portalLocalizationLocaleCache
<b>Use</b>	Used to store collection of LocalizationLocale objects. Localization locale specifies language, character encoding, country and variant.
<b>Key</b>	The key is private static final String called portalLocalizationLocaleCachekey.

**Table 2 portalLocalizationLocaleCache (Continued)**

<b>Value</b>	A set of LocalizationLocale objects.
<b>Notes</b>	Default TTL value should be okay. Max Entries could be set to a number based on the number of rows in the L10N_LOCALE table, i.e. number of supported locales. Default values: MaxEntries=500; TimeToLive=-1

**Table 3 portletControlTreeCache**

<b>Cache</b>	portletControlTreeCache
<b>Use</b>	Used to store portlet control trees for floating portlets.
<b>Key</b>	The combination portletInstanceId and locale.
<b>Value</b>	A portlet control tree.
<b>Notes</b>	Default TTL value should be okay, MaxEntries could be set to a number based on number of floatable portlet instances in a portal (including user customized portlets) and number of supported locales.  It is recommended that the TTL be left at -1 because the cached default desktop needs to be kept in the cache indefinitely and the cached item for a logged in user is removed when they log out so there is no need to expire a user's cached items. To avoid having the LRU mechanism kick the cached default desktop out of the cache, the MaxEntries should be set to at least (max # of concurrent logged in users + 1) X (# of locales supported). If the cache is too small then LRU will kick out the cached default desktop and the memory saving advantage of this approach will be lost.  Default values: MaxEntries=500; TimeToLive=-1

**Table 4 portletPreferencesCache**

<b>Cache</b>	portletPreferencesCache
<b>Use</b>	Used to store portlet preferences.

**Table 4 portletPreferencesCache (Continued)**

<b>Key</b>	An instance of PortletPreferenceId.
<b>Value</b>	A map of preferences.
<b>Notes</b>	Default TTL and Max Entries values could be set to a value depending on amount of available memory and total number of preferences (at the application level). Defaults: MaxEntries = 500, TimeToLive=60000 (one minute)

**Table 5 portalLocalizationResourceCache**

<b>Cache</b>	portalLocalizationResourceCache
<b>Use</b>	Used to store localization resources.
<b>Key</b>	The localizationIntersection.
<b>Value</b>	A LocalizationResource.
<b>Notes</b>	Default TTL and MaxEntries values could be set to a value based on total number of localization resources in the system, which is a combination of non-customized and customized localization resources, and the amount of available memory. Default values: MaxEntries=500; TimeToLive=-1

**Table 6 portalControlTreeCache**

<b>Cache</b>	portalControlTreeCache
<b>Use</b>	Used to store portal control trees. Only used for streaming portals.
<b>Key</b>	The combination of webapp, portal, desktop, locale and optional user name.

**Table 6 portalControlTreeCache (Continued)**

<b>Value</b>	A portal control tree.
<b>Notes</b>	Default TTL value should be okay. This cache will contain one entry for the default portal, plus one entry for each user who has customized his or her portal. Max Entries could be set to a number based on number of users and available memory. If there are any changes to portal this cache will be flushed.  Default values: MaxEntries=500; TimeToLive=-1

**Table 7 portalMarkupdefinitionCache**

<b>Cache</b>	portalMarkupdefinitionCache
<b>Use</b>	Used to store MarkupDefinition objects.
<b>Key</b>	A MarkupDefintionID.
<b>Value</b>	A MarkupDefinition.
<b>Notes</b>	Set this according to the number of rows in the PF_MARKUP_Definition.  Markup is the blueprint for a portal library resource (desktop, book, page, portlet, placeholder, menu, Look And Feel, layout, shell or theme).  Default values: MaxEntries=500; TimeToLive=60000 (one minute).

## WSRP Caches

**Table 8 remoteProducerInfoCache**

<b>Cache</b>	remoteProducerInfoCache
<b>Use</b>	Caches the metadata for producers added to a consumer application.
<b>Key</b>	Name of the consumer web application.

**Table 8 remoteProducerInfoCache (Continued)**

<b>Value</b>	A java.util.HashMap containing producer metadata. This map is keyed with the producerHandle of each producer.
<b>Notes</b>	This cache is used to look for producer metadata when a user or administrator is trying to interact with a remote portlet or a producer. Default values: MaxEntries=500; TimeToLive=-1

**Note:** The remoteProducerInfoCache is not included in the Administration Console. If you want to manage this cache, you need to add it manually, see [“Adding a Cache.”](#)

**Table 9 registrationHandleCache**

<b>Cache</b>	registrationHandleCache
<b>Use</b>	Used to store registration handles of all registered consumers, for all producers.
<b>Key</b>	The registrationHandle of the consumer.
<b>Value</b>	A java.lang.boolean object with a value of true/false.
<b>Notes</b>	This cache is used to cache whether or not a particular registrationHandle is valid. Default values: MaxEntries=500;TimeToLive=-1.

**Note:** The registrationHandleCache is not included in the Administration Console. If you want to manage this cache, you need to add it manually, see [“Adding a Cache.”](#)

**Table 10 proxyPortletCache**

<b>Cache</b>	proxyPortletCache
<b>Use</b>	This caches the ProxyPortlets by proxyportletId.
<b>Key</b>	String representing the portlet instance id.

**Table 10** `proxyPortletCache`

<b>Value</b>	Information from the consumer registry and about the proxy portlet instance ( <code>com.bea.wsrp.services.persistence.internal.ProxyPortletInfoInternal.ProxyPortletInfoInternalObject</code> ).
<b>Notes</b>	Default values: MaxEntries: 100; TimeToLive = -1

## Content and Ad Caches

**Table 11** `binaryCache.repository_name`

<b>Cache</b>	<code>binaryCache.repository_name</code>
<b>Use</b>	Used to store binary property values for a repository node.
<b>Key</b>	String (node ID + Property ID)
<b>Value</b>	A byte array associated with the binary property.
<b>Notes</b>	Set this according to the number and size of binary property values. Default values: MaxEntries: 10; TimeToLive:60000 (one minute)

**Table 12** `adServiceCache`

<b>Cache</b>	<code>adServiceCache</code>
<b>Use</b>	Used to store the results of searches for content rendered in a placeholder (ads). Used by the AdHelper to increase the speed of ad queries.
<b>Key</b>	The ad query ( <code>java.lang.String</code> )
<b>Value</b>	A <code>Content []</code>
<b>Notes</b>	Set this according to the number of ad queries and the amount of content expected to be retrieved. Consider basing the maximum size on the total number of ad queries. If the ads returned from a particular query do not change, consider increasing the TTL. Default values: MaxEntries=32; TimeToLive=300000 (five minutes)

**Table 13** *nodePathCache.repository\_name*

<b>Cache</b>	<i>nodePathCache.repository_name</i>
<b>Use</b>	Used to store a list of nodes for a repository based on a path.
<b>Key</b>	A String (NodeID).
<b>Value</b>	A Node.
<b>Notes</b>	Set according to the number of nodes in a repository. Default values: MaxEntries=50; TimeToLive=60000 (one minute)

**Table 14** *searchCache*

<b>Cache</b>	<i>searchCache</i>
<b>Use</b>	Used to store an array of IDs for nodes that satisfy a content search.
<b>Key</b>	A Search, which contain parameters for a query.
<b>Value</b>	An ID array of nodes that satisfy a query.
<b>Notes</b>	There is only one search cache used for all repositories. Default values: MaxEntries=20; TimeToLive=60000 (one minute) Set the MaxEntries according to the amount of content expected to be retrieved. Set Time To Live according to how fresh the content should be.

## User Management Caches

**Table 15** *entityIdCache*

<b>Cache</b>	<i>entityIdCache</i>
<b>Use</b>	Caches the ID for an entity (user or group ID)

**Table 15 entityIdCache (Continued)**

<b>Key</b>	A com.bea.p13n.property.PropertyLocator. PropertyLocator is based on a user or group name (ENTITY.ENTITY_NAME) and entity type (ENTITY.ENTITY_TYPE).
<b>Value</b>	The entity ID (java.lang.Long).
<b>Notes</b>	Use the ENTITY table as a guide for the maximum size. The object being stored is a Long, which is fairly small. Therefore, it might be possible to set this cache's maximum size to the number of entries in the ENTITY table.  Consider how often the ENTITY table might change when setting the TTL.  Default values: MaxEntries=500;TimeToLive=600000

**Table 16 jndiNameCache**

<b>Cache</b>	jndiNameCache
<b>Use</b>	Stores the JNDI names of entity property managers and UUP managers.
<b>Key</b>	An entity ID.
<b>Value</b>	The home name, which is a string value.
<b>Notes</b>	Set this according the combination of the number of entity property managers and the number of UUP managers.  Default values: MaxEntries=500;TimeToLive=600000

**Table 17 entityPropertyCache**

<b>Cache</b>	entityPropertyCache
<b>Use</b>	Caches property values for users and groups.
<b>Key</b>	A com.bea.p13n.property.PropertyLocator. PropertyLocator is based on the user or group name (ENTITY.ENTITY_NAME), entity type (ENTITY.ENTITY_TYPE, user or group) and property set type (PROPERTY_KEY.PROPERTY_SET_TYPE, usually USER).

**Table 17 entityPropertyCache (Continued)**

<b>Value</b>	A <code>com.bea.p13n.property.EntityPropertyCache</code> object. This object contains a Map that stores property values keyed off the property set name and property name.
<b>Notes</b>	<p>The larger you can afford to make this cache, the better.</p> <p>Use the ENTITY table as a guide for maximum size. The number of entries in this table should be the maximum number of cache entries that would ever be created. In most cases, there will be more entries here than you would want for a maximum cache size. So consider the average number of users you expect to be using your application at the same time.</p> <p>Consider a TTL based on how often new properties will be added to the property sets. If they are not being modified often, then a higher TTL might be appropriate.</p> <p>Default values: MaxEntries=500;TimeToLive=600000</p>

**Table 18 profileTypeCache**

<b>Cache</b>	profileTypeCache
<b>Use</b>	Caches user profile types that are used to look up the appropriate user manager profile manager when retrieving a user profile.
<b>Key</b>	A String (the user name).
<b>Value</b>	A String (the profile type).
<b>Notes</b>	<p>This should be set based on the number of concurrent users. Set the TimeToLive never to expire</p> <p>Default values: MaxEntries=100;TimeToLive=3600000</p>

**Table 19 propertyKeyIdCache**

<b>Cache</b>	propertyKeyIdCache
<b>Use</b>	Caches the unique ID associated with a property set type, property set and property name combination (primary key in the PROPERTY_KEY database table).

**Table 19** `propertyKeyIdCache` (Continued)

<b>Key</b>	Based on a property set type, property set, and property name combination (inner class called <code>PropertyKeyLocator</code> ).
<b>Value</b>	The ID ( <code>java.lang.Long</code> )
<b>Notes</b>	<p>Maximum size should be set with an eye towards the maximum number of properties in the application (use the <code>PROPERTY_KEY</code> table as an indicator).</p> <p>Consider a TTL based on how often these unique ID combinations are likely to change.</p> <p>Default value: <code>MaxEntries=500;TimeToLive=600000</code></p>

## Campaign and Discount Caches

**Table 20** `globalDiscountCache`

<b>Cache</b>	<code>globalDiscountCache</code>
<b>Use</b>	Stores computed global discount definitions. This is the set of global discounts that is applicable to all users.
<b>Key</b>	The <code>globalDiscountSet</code> name ( <code>java.lang.String</code> )
<b>Value</b>	The <code>java.util.Set</code> of <code>qualificationDiscountDef</code> objects.
<b>Notes</b>	<p>Set this to the number of global discounts in your application.</p> <p>The frequency of changes to the global discounts should determine TTL.</p> <p>Default values: <code>MaxEntries=10; TimeToLive=300000</code> (five minutes)</p>

**Table 21** `discountCache`

<b>Cache</b>	<code>discountCache</code>
<b>Use</b>	Used to store computed discount definitions (applicable to individual customers or to customer segments).

**Table 21 discountCache (Continued)**

<b>Key</b>	A QualificationDiscountId. This is essentially a wrapping around a java.lang.Integer that represents the ID of a discount.
<b>Value</b>	The java.util.Set of qualificationDiscountDef objects
<b>Notes</b>	Set this to the number of discounts in your application. Frequency of changes to the global discounts should determine TTL. Default values: MaxEntries=100; TimeToLive=300000 (five minutes)

## Commerce Caches

**Table 22 globalDiscountAssocCache**

<b>Cache</b>	globalDiscountAssocCache
<b>Use</b>	Stores computed global discount associations. This is the set of discount associations that is applicable to all users.
<b>Key</b>	CustomerPk, which is a unique identifier for a customer (java.lang.String).

**Table 22 globalDiscountAssocCache (Continued)**

<b>Value</b>	A <code>DiscountAssociation</code> object. A discount association is the mapping of a Customer to a Discount. It is used to track and limit how many times the discount is used by a particular customer.
<b>Notes</b>	<p>Default values: <code>MaxEntries=100</code>; <code>TimeToLive=3600000</code> (one hour).</p> <p>Set <code>MaxEntries</code> to the number of global discount associations in your application. The frequency of changes to the global discount associations should determine TTL.</p> <p>To use this cache, you must start the WebLogic server using the command line option: <code>-Denable.discount.assoc.caches=true</code></p> <p>This enables caching of discount associations and global discount associations. The default is <code>false</code>, which means a separate read is performed for every pricing calculation for every line item in every shopping cart. Enabling the cache reduces database load by storing associations in a cache after the first read.</p> <p>You can use the Service Administration tools within the Portal Administration Console to manage this cache. First you must add the cache so that it is visible in the Portal Administration Console, as described in the online help for the Portal Administration Console.</p> <p>You can also manage the cache using the p13n cache API.</p>

**Table 23 discountAssocCache**

<b>Cache</b>	<code>discountAssocCache</code>
<b>Use</b>	Stores computed discount associations (applicable to individual customers or to customer segments).
<b>Key</b>	<code>CustomerPk</code> , which is a unique identifier for a customer ( <code>java.lang.String</code> ).

**Table 23 discountAssocCache (Continued)**

<b>Value</b>	A DiscountAssociation object. A discount association is the mapping of a Customer to a Discount. It is used to track and limit how many times the discount is used by a particular customer.
<b>Notes</b>	<p>Default values: MaxEntries=100; TimeToLive=3600000 (one hour).</p> <p>Set <code>MaxEntries</code> to the number of discount associations in your application.</p> <p>The frequency of changes to the discount associations should determine TTL.</p> <p>To use this cache, you must start the Weblogic server using the command line option:  <code>-Denable.discount.assoc.caches=true</code></p> <p>This enables caching of discount associations and global discount associations.</p> <p>The default is <code>false</code>, which means a separate read is performed for every pricing calculation for every line item in every shopping cart. Enabling the cache reduces database load by storing associations in a cache after the first read.</p> <p>You can use the Service Administration tools within the Portal Administration Console to manage this cache. First you must add the cache so that it is visible in the Portal Administration Console, as described in the online help for the Portal Administration Console.</p> <p>You can also manage the cache using the p13n cache API.</p>

**Table 24 CategoryCache**

<b>Cache</b>	<b>categoryCache</b>
<b>Use</b>	<p>Stores the root <code>com.beasys.commerce.ebusiness.catalog.Category</code>, the total number of categories in the product catalog (<code>java.lang.Integer</code>) and the <code>CategoryInfo</code> for each category.</p> <p><code>CategoryManagerImpl</code> gets the cache name from the <code>ejb-jar.xml</code> in <code>commerce.jar</code></p>
<b>Key</b>	<p>The key for the root <code>Category</code> is a static final <code>String</code> variable in the <code>CategoryManagerImpl</code> class. The key for the total number of categories is also a static final <code>String</code> variable in the <code>CategoryManagerImpl</code> class. The key for a given <code>CategoryInfo</code> object is a <code>com.beasys.commerce.ebusiness.catalog.CategoryKey</code>.</p>

**Table 24 CategoryCache (Continued)**

<b>Value</b>	The value for the root <code>Category</code> is <code>com.beasys.commerce.ebusiness.catalog.Category</code> . The value for the total number of categories is a <code>java.lang.Integer</code> . The value for the category info objects is a <code>com.beasys.commerce.ebusiness.catalog.service.category.CategoryInfo</code> .
<b>Notes</b>	The root <code>Category</code> and the total number of categories occupy two slots in the cache and the remaining slots are occupied by the <code>CategoryInfo</code> objects, so consider the total number of categories in the product catalog plus 2 when setting the maximum cache size.  Consider how often these categories will change when setting TTL.  Default values: <code>MaxEntries:1000;TimeToLive: 8640000</code>

**Table 25 ProductItemCache**

<b>Cache</b>	<code>ProductItemCache</code> ( <code>ProductItemManagerImpl</code> gets the cache name from the <code>ejb-jar.xml</code> in <code>commerce.jar</code> .)
<b>Use</b>	Stores the total number of product items in the catalog as well as the product items
<b>Key</b>	The key for the total number of product items is a static final <code>String</code> variable in <code>ProductItemManagerImpl</code> . The key for the product items is a <code>com.beasys.commerce.ebusiness.catalog.ProductItemKey</code> .
<b>Value</b>	The value for the total number of product items is a <code>java.lang.Integer</code> . The value for the product item is a <code>com.beasys.commerce.ebusiness.catalog.ProductItem</code> .
<b>Notes</b>	Consider the total number of product items when setting the maximum cache size.  Consider how often these product items will change when setting the TTL.  Default values: <code>MaxEntries=1000;TimeToLive=21600000</code>

## WebLogic Portal Cache Reference