



BEA WebLogic Portal[®]

Service Administration Guide

Version 8.1
Revised: December 2004

Copyright

Copyright © 20004 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software and documentation is subject to and made available only pursuant to the terms of the BEA Systems License Agreement and may be used or copied only in accordance with the terms of that agreement. It is against the law to copy the software except as specifically allowed in the agreement. This document may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from BEA Systems, Inc.

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the BEA Systems License Agreement and in subparagraph (c)(1) of the Commercial Computer Software-Restricted Rights Clause at FAR 52.227-19; subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013, subparagraph (d) of the Commercial Computer Software--Licensing clause at NASA FAR supplement 16-52.227-86; or their equivalent.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE SOFTWARE AND DOCUMENTATION ARE 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 SOFTWARE OR WRITTEN MATERIAL IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks or Service Marks

BEA, Jolt, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Liquid Data for WebLogic, BEA Manager, BEA WebLogic Commerce Server, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Personalization Server, BEA WebLogic Platform, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop and How Business Becomes E-Business are trademarks of BEA Systems, Inc.

All other trademarks are the property of their respective companies.

Contents

Service Administration Guide

Service Administration Overview	1-1
Portal Framework Caches	A-1
WSRP Caches	A-5
Content and Ad Caches	A-6
User Management Caches	A-8
Campaign and Discount Caches	A-10
Commerce Caches	A-11

Service Administration Guide

Service Administration Overview

WebLogic Portal provides a single framework for configuring, accessing, monitoring, and maintaining caches. If configured properly, the caches can reduce the time needed to retrieve frequently used data.

Many WebLogic Portal services use preconfigured caches that you can tune to meet your performance needs. Some services use internally configured caches that you cannot configure or access. If you extend or create additional services, developers can use the cache framework to define and use your own set of caches, and make them available to you in the WebLogic Administration Portal.

The administration portal allows you to configure the following default services:

- AdContentProvider
- Ad Service
- Behavior Tracking Service
- Caches
- Campaign Service
- Document Connection Pool
- Document Manager
- Event Service

- Mail Service
- Payment Service Client
- Scenario Service
- Tax Service Client

WebLogic Portal Cache Settings

This appendix lists the caches used to tune performance in WebLogic Portal 8.1. Find a cache you need to adjust, then use the WebLogic Administration Portal to adjust the cache settings.

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

Portal Framework Caches

Table 0-1 portalContentUriCache

Cache	portalContentUriCache
Use	This caches portal content URIs for a combination of webapp, portal, locale and optional user name.
Key	Key is equal to portal path + name of web application.

Table 0-1 portalContentUriCache

Value	Portal content URI
Notes	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 0-2 portalLocalizationLocaleCache

Cache	portalLocalizationLocaleCache
Use	Used to store collection of LocalizationLocale objects. Localization locale specifies language, character encoding, country and variant.
Key	The key is private static final String called "portalLocalizationLocaleCachekey"
Value	A set of LocalizationLocale objects.
Notes	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 0-3 portletControlTreeCache

Cache	portletControlTreeCache
Use	Used to store portlet control trees for floating portlets.
Key	The combination portletInstanceId and locale.

Table 0-3 portletControlTreeCache

Value	A portlet control tree.
Notes	<p>Default TTL value should be okay, Max Entries 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.</p> <p>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.</p> <p>Default values: MaxEntries=500; TimeToLive=-1</p>

Table 0-4 portletPreferencesCache

Cache	portletPreferencesCache
Use	Used to store portlet preferences.
Key	An instance of PortletPreferenceId.
Value	A map of preferences.
Notes	<p>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).</p> <p>Defaults: MaxEntries = 500, TimeToLive=60000 (one minute)</p>

Table 0-5 portalLocalizationResourceCache

Cache	portalLocalizationResourceCache
Use	Used to store localization resources.
Key	The localizationIntersection.

Table 0-5 portalLocalizationResourceCache

Value	A LocalizationResource.
Notes	Default TTL and Max Entries 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 0-6 portalControlTreeCache

Cache	portalControlTreeCache
Use	Used to store portal control trees. Only used for streaming portals.
Key	The combination of webapp, portal, desktop, locale and optional user name.
Value	A portal control tree.
Notes	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 0-7 portalMarkupDefinitionCache

Cache	portalMarkupDefinitionCache
Use	Used to store MarkupDefinition objects.
Key	A MarkupDefintionID.

Table 0-7 portalMarkupDefinitionCache

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

WSRP Caches

Table 0-8 remoteProducerInfoCache

Cache	remoteProducerInfoCache
Use	Caches the metadata for producers added to a consumer application.
Key	Name of the consumer web application.
Value	A java.util.HashMap containing producer metadata. This map is keyed with the producerHandle of each producer.
Notes	<p>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.</p> <p>Default values: MaxEntries=500; TimeToLive=-1</p>

Table 0-9 registrationHandleCache

Cache	registrationHandleCache
Use	Used to store registrationHandles of all registered consumers, for all producers.
Key	The registrationHandle of the consumer.

Table 0-9 registrationHandleCache

Value	A java.lang.boolean object with a value of true/false.
Notes	This cache is used to cache whether or not a particular registrationHandle is valid. Default values: MaxEntries=500;TimeToLive=-1.

Content and Ad Caches

Table 0-10 binaryCache.<repository_name>

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

Table 0-11 adServiceCache

Cache	adServiceCache
Use	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.
Key	The ad query (java.lang.String)
Value	A Content []
Notes	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 0-12 nodePathCache.<repository_name>

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

Table 0-13 searchCache

Cache	searchCache
Use	Used to store an array of IDs for nodes that satisfy a content search.
Key	A Search, which contain parameters for a query.
Value	An ID array of nodes that satisfy a query.
Notes	There is only one search cache used for all repositories. Default values: MaxEntries=20; TimeToLive=60000 (one minute)

Table 0-14 nodeCache.<repository_name>

Cache	nodeCache.<repository_name>
Use	Used to store repository nodes. Each repository has its own cache setting.
Key	A String representing the node ID.
Value	A node.
Notes	Set this according to the number of nodes in a repository. Default values: MaxEntries=50; TimeToLive=6000 (one minute)

User Management Caches

Table 0-15 entityIdCache

Cache	entityIdCache
Use	Caches the id for an entity (user or group id)
Key	A com.bea.p13n.property.PropertyLocator. PropertyLocator is based on a user or group name (ENTITY.ENTITY_NAME) and entity type (ENTITY.ENTITY_TYPE).
Value	The entity id (java.lang.Long).
Notes	<p>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.</p> <p>Consider how often the ENTITY table might change when setting the TTL.</p> <p>Default values: MaxEntries=500;TimeToLive=600000</p>

Table 0-16 jndiNameCache

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

Table 0-17 entityPropertyCache

Cache	entityPropertyCache
Use	Caches property values for users and groups.

Table 0-17 entityPropertyCache

Key	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).
Value	A com.bea.p13n.property.EntityPropertyCache object. This object contains a Map that stores property values keyed off the property set name and property name.
Notes	<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 0-18 profileTypeCache

Cache	profileTypeCache
Use	Caches user profile types that are used to look up the appropriate user manager profile manager when retrieving a user profile.
Key	A String (the user name).
Value	A String (the profile type).
Notes	<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 0-19 propertyKeyIdCache

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

Campaign and Discount Caches

Table 0-20 globalDiscountCache

Cache	globalDiscountCache
Use	Stores computed global discount definitions. This is the set of global discounts that is applicable to all users.
Key	The globalDiscountSet name (java.lang.String)
Value	The java.util.Set of qualificationDiscountDef objects.
Notes	<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: MaxEntries=10; TimeToLive=300000 (five minutes)</p>

Table 0-21 discountCache

Cache	discountCache
Use	Used to store computed discount definitions (applicable to individual customers or to customer segments).
Key	A QualificationDiscountId. This is essentially a wrapping around a java.lang.Integer that represents the ID of a discount.
Value	The java.util.Set of qualificationDiscountDef objects
Notes	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 0-22 categoryCache

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

Table 0-22 categoryCache

Value	The value for the root Category 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> .
Notes	<p>The root Category and the total number of categories occupy two slots in the cache and the remaining slots are occupied by the CategoryInfo objects, so consider the total number of categories in the product catalog plus 2 when setting the maximum cache size.</p> <p>Consider how often these categories will change when setting TTL.</p> <p>Default values: MaxEntries:1000;TimeToLive: 8640000</p>

Table 0-23 productItemCache

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