Oracle® Fusion Middleware

Release Notes for Oracle Coherence

12*c* (12.2.1)

E55690-05

April 2016

ORACLE®

Oracle Fusion Middleware Release Notes for Oracle Coherence , 12c (12.2.1)

E55690-05

Contents

3   Deprecated and Desupported Features

Preface

This document describes changes and enhancements that have been made to Oracle Coherence.

- Audience

- Documentation Accessibility

- Related Documents

- Conventions

Audience

This document is intended for users of Oracle Coherence.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Coherence documentation set:

- *Administering Oracle CoherenceAdministering Oracle Coherence*

- *Administering HTTP Session Management with Oracle Coherence*Web*

- *Developing Applications with Oracle Coherence*

- *Developing Remote Clients for Oracle Coherence*

- *Installing Oracle Coherence*

- *Integrating Oracle Coherence*

- *Managing Oracle Coherence*

- *Securing Oracle Coherence*

- *Java API Reference for Oracle Coherence*

- *.NET API Reference for Oracle Coherence*

- *C++ API Reference for Oracle Coherence*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Technical Changes and Enhancements

Technical changes and enhancements made to Oracle Coherence.
Oracle updates the release notes periodically after the software
release. This document is accurate at the time of publication.

This chapter includes the following sections:

Download and Install the Latest Software Patch

New and Improved Coherence Data Grid Functionality

Oracle Coherence for Java 12c (12.2.1)

Oracle Coherence for .NET 12c (12.2.1)

Oracle Coherence for C++ 12c (12.2.1)

Patch Notes for 12c (12.2.1.0.1)

1.1 Download and Install the Latest Software Patch

Go to My Oracle Support to download the latest software patches.

https://support.oracle.com

See the README file in the patch distribution for up-to-date information on the
software fixes provided by the patch.

To download and install the latest software patch:

1. Login to My Oracle Support.

2. Click the **Patches & Updates** tab.

3. Under the Patch Search tab, select **Product or Family (Advanced Search)**, and
    select the **Include all patches in a product family** check box.

4. Enter **Oracle Coherence** as the product, select the platform and release, and click
    Search.

The list of currently available patches for Oracle Coherence is returned.

1.2 New and Improved Coherence Data Grid Functionality

Oracle Coherence is the industry's leading distributed in-memory data grid product.
This release contains many new features that help reduce complexity, ease
configuration, and accelerate time to market of scalable solutions.

- **Persistence** – Coherence persistence is a set of tools and technologies that manage
  the persistence and recovery of Coherence distributed caches. Cached data is
  persisted so that it can be quickly recovered after a catastrophic failure or after a

cluster restart due to planned maintenance. For more information, see "Persisting Caches" in *Administering Oracle Coherence*.

- **Federated Caching** – Federated caching replicates and synchronizes cache data across multiple geographically dispersed clusters. Cached data is replicated across clusters to provide redundancy, off-site backup, and multiple points of access for application users in different geographical locations. For more information, see "Replicating Caches Across Clusters" in *Administering Oracle Coherence*.

- **Security Enhancements** – Coherence includes new security functionality. The functionality includes:

  - Audit logs – Audit logs are used to record user access to cluster operations. For more information, see "Enable Security Audit Logs" in *Securing Oracle Coherence*.

  - Access control authorization – Access control authorization allows applications to define their own authorization logic to limit access to cluster operations. For more information, see "Authorizing Access to Server-Side Operations" in *Securing Oracle Coherence*.

  - Additional JAAS permissions – JAAS permissions protect various parts of the Coherence API using the Java Security Manager. For more information, see "Programmatically Specifying Local Permissions" in *Securing Oracle Coherence*.

  - SSL Protocols and Cipher Suites – An SSL socket provider can be configured to control the use of potentially weak ciphers or specific protocol versions. For more information, see "Controlling Cipher Suites and Protocol Version Usage" in *Securing Oracle Coherence*.

- **Support for Java 8 Features** – The Coherence API has been updated so that you can use programming features that were introduced in the Java 8 release. The features include lambda, streams, and default methods. These features provide ease of use and flexibility when performing data grid operations. For more information, see "Support for Java 8 Features" in *Developing Applications with Oracle Coherence*.

- **Support for Generics** – Java Generics provide compile and runtime type checking together with compile type-inference. The Coherence API has been refactored to support Java Generics. In addition, types can be explicitly configured as part of the `NamedCache` API. For more information, see "Support for Generics" in *Developing Applications with Oracle Coherence*.

- **Cluster Port and Address Management** – Port and address selection has been changed to simplify cluster management and to allow the sharing of cluster ports and addresses among multiple clusters. For most use cases, ports and addresses do not need to be explicitly set.

  - Coherence port selection has been simplified to facilitate port management. Coherence now includes a common cluster port that is used for multicast communication, well known addresses, and extend proxies that are using the name service. In addition, unicast ports are automatically assigned. For more information, see "Setting Up a Cluster" in *Developing Applications with Oracle Coherence* and "Configuring Extend Proxies" in *Developing Remote Clients for Oracle Coherence*.

  - Coherence address selection has been simplified to facilitate address management. For unicast communication, Coherence automatically selects a routable IP with the highest MTU for computers that have multiple IPs or NICs. For well known addresses, Coherence selects the IP which is routable to the IPs

on the WKA list. In addition, Coherence can now automatically resolve IP addresses (at runtime) that are associated with a DNS name. Well known addresses and proxy addresses can be stored in a DNS server and centrally managed and updated in real time. For more information, see "Setting Up a Cluster" in *Developing Applications with Oracle Coherence*.

- **Multitenancy** – Coherence applications that are deployed using managed Coherence servers can take full advantage of the density and operational efficiencies that are provided by Weblogic Server Multitenant. Coherence features include: isolating caches across domain partitions, sharing caches across domain partitions, and tooling support exposed through WLST and Fusion Middleware Control. For more information, see "Configuring Coherence" in *Oracle Fusion Middleware Using WebLogic Server Multitenant*.

- **Asynchronous NamedCache API** – The `AsyncNameCache` interface allows cache operations to be completed in parallel and can improve throughput and result in more responsive user interfaces. For more information, see "Performing NameCache Operations Asynchronously" in *Developing Applications with Oracle Coherence*.

- **Truncate Operation** – The `truncate` method on the `NamedCache` interface clears a cache but does not raise any entry-level cache events. This new API significantly reduces the memory pressure on the server side and dramatically reduces the network consumption for caches with listeners and is especially beneficial for near caching.

- **Sliding Expiry** – Sliding expiry extends the expiry of cache entries that are being accessed. Sliding expiry is enabled by default for cache entries that are updated, but can also be enabled based on read operations and operations that are non-mutative. For more information, see "Capacity Planning" in *Developing Applications with Oracle Coherence*.

- **Dynamic Management Mode** – Dynamic management elects one of the nodes in the cluster to be the management node. This is the default management mode for managed Coherence servers. Cluster members no longer need to be explicitly configured for JMX management. For more information, see "Using Dynamic Management Mode" in *Managing Oracle Coherence*.

- **Dynamic Thread Pools** – All Coherence services use dynamically sized daemon thread pools. Dynamic thread pools are enabled by default and are configured for a service using the `<thread-count-max>` and `<thread-count-min>` elements when defining a cache scheme.

- **Coherence-JVisualVM Plug-In** – A new version of the Coherence-JVisualVM plug-in is available and includes new functionality. The functionality includes: reporting node state, displaying near cache statistics, displaying partition statistics, managing cache persistence, and managing federated caching. For more information, see "Using the Coherence-JVisualVM Plug-In" in *Managing Oracle Coherence*.

- **Coherence CohQL** – New functionality is included in Coherence CohQL. The functionality includes: support for loading GAR modules, support for WLS Multitenant, and support for cache persistence. For more information, see, "Using Coherence Query Language" in *Developing Applications with Oracle Coherence*.

- **Server-Sent Events** – Server-sent events allow Coherence REST applications to automatically receive cache events from the Coherence cluster. For more

information, see "Using Server-Sent Events" in *Developing Remote Clients for Oracle Coherence*.

1.3 Oracle Coherence for Java 12*c* (12.2.1)

New features, improvements, and bug fixes have been added to these Oracle Coherence for Java components:

Partitioned Cache Enhancements and Fixes

Management and Monitoring Enhancements and Fixes

TCMP/Cluster Enhancements and Fixes

Security Enhancements and Fixes

Coherence*Extend Framework Enhancements and Fixes

Coherence*Web Enhancements and Fixes

Serialization Framework Enhancements and Fixes

Configuration Enhancements and Fixes

Managed Coherence Servers Enhancements and Fixes

Other Enhancements and Fixes

1.3.1 Partitioned Cache Enhancements and Fixes

- The read-write backing map now accelerates store and write operations for entries that are being evicted but that have not yet been written to the cache store.

- Added `SamplingEventTransformer` class that implements server-side event throttling.

- Added asynchronous `NamedCache` API support (`AsyncNameCache` interface).

- Added support for non-blocking invocable. The `NonBlockingInvocable` interface is designed to allow the invocation service thread to execute the corresponding task and get an invocation result without blocking.

- Added support for callbacks to asynchronous entry processors.

- Added support for streaming aggregation to reduce memory overhead and improve performance.

- Added support for the use of Java Generics.

- Added the `<key-type>` and `<value-type>` elements within the `<cache-mapping>` element to declare the types that are supported by a `NamedCache` cache.

- Added support for lambdas to the `NamedCache` API

- Added support for dynamic lambda execution.

- Added support for default `Map` methods introduced in Java 8 to the `InvocableMap` interface.

- Added the ability to rebuild index data structures asynchronously after a partition transfer or failover to significantly reduce the time to recovery.

- Added support for executing asynchronous aggregators.

- Added support in Elastic Data to use pure RAM Journal without overflowing to Flash.

- Improved near cache performance when using the `present` invalidation strategy.

- Added the ability to enforce filter evaluation order for array based filters (for example, `AndFilter`, `AnyFilter`).

- Enhanced CohQL so that it is possible to provide a custom value extractor (for example `POFExtractors`) to be used in CohQL queries.

- Added the `truncate` method to the `NamedCache` interface that is functionally equivalent to `clear` but doesn't raise any entry-level cache events. This new API significantly reduces the memory pressure on the server side and dramatically reduces the network consumption for caches with listeners and is especially beneficial for near caching.

- Added support for sliding expiry. The expiry of cache entries are extended upon read access.

- Cache entries are proactively evicted from a backing map after the cache expiry time is reached and no longer require a cache operation to initiate the eviction.

- Fixed inconsistent duplicate event interceptor registration behavior when using name and identifier.

- Fixed a problem causing a slow memory leak for map listeners that receive local-only (in-process) events.

- Fixed an issue where repeated calls to `NamedCache` operations (`create`, `write` and `destroy`) cause `Storage` instance leak.

- Fixed the inability to determine the number of tasks waiting to be executed (`ServiceMBean.EventBacklog`) on the `EventDispatcher` thread, or the total number of tasks executed (`ServiceMBean.EventCount`).

- Fixed an issue where the `NamedCache.lock(key, -1)` call does not wait until the lock is available.

- Fixed an issue where the `addIndex` method that is run in a non-synchronized thread gave the wrong results for a replicated cache.

- Fixed an issue when using the `MultiExtractor` class as an index that may result in inaccurate indexed data.

- Fixed an issue with invoking the `GuardSupport.heartbeat` method during execution of post commit live events.

- Fixed an issue with the use of expiry or eviction together with high load causes an `AssertionException` exception or deadlock.

- Fixed an issue where near cache event interceptors that are defined in a cache configuration file are not registered when using the `ExtensibleConfigurableCacheFactory` class.

- Fixed an issue that caused asynchronous entry processors to be deferred after a service is suspended and resumed.

- Fixed an issue with `ConfigurableCacheMap` methods returning incorrect values when using an `ObservableSplittingBackingCache` instance.

- Fixed an issue with the `UpdaterProcessor` class that could cause `CacheStore`, `BinaryEntryStore`, and `Listener` instances to miss entry updates.

- Fixed an issue where a backing map listener does not process events immediately and can result in a stale backup value to be returned.

- Fixed an issue where precommit transaction events (`COMMITTING`) show entries that have not changed.

- Fixed an issue where the front map of a near cache does not get cleared when the back cache is destroyed.

- Fixed an issue where the front cache of a near cache is not updated after all storage nodes are restarted.

1.3.2 Management and Monitoring Enhancements and Fixes

- Added an ability to call `MBeanServer` remotely.

- Added a new dynamic management mode that elects one of the nodes in the cluster to be the management node.

- Added an attribute called `IndexTotalUnits` in the `StorageManagerMBean`.

- Improved index size accuracy.

- Add the ability to pass report file and report group XML as a string when using `ReporterMBean` operations.

- Fixed an issue where the `ClusterMBean.logClusterState` method call is not generating thread dumps if the `extendedmbeanname` system property is set to `true`.

- Fixed a bug where a value set for the `<default-domain-name>` element in an operational override configuration file was being ignored.

1.3.3 TCMP/Cluster Enhancements and Fixes

- The default unicast listener address is derived from the Well Known Address (WKA) list when available, selecting a local IP on the same subnet as the WKA addresses.

- The WKA lists can be represented using a single hostname where DNS has been configured to return a list of IPs.

- Coherence uses port 7574 as the default cluster port for multicast communication as well as for the Name Service. Unicast ports are automatically selected.

- Multiple clusters can now share a cluster port and Multicast or WKA address. For most use cases, there is no reason to change the cluster port, or multicast address.

1.3.4 Security Enhancements and Fixes

- Added an ability to authorize access to cache data

- Added the ability to capture audit logs to record user access to clustered data.

- Added additional JAAS permissions to protect various parts of the Coherence API using the Java Security Manager.

- Added the ability to configure an SSL socket provider to control the use of potentially weak ciphers or specific protocol versions.

1.3.5 Coherence*Extend Framework Enhancements and Fixes

- The Proxy service now binds to the same address as the Name service.

- Coherence*Extend clients can be redirected by a proxy server that is at its connection limit.

- Added Guardian support to the `NameService` interface to auto-restart the acceptor.

- Added support for server-sent events to Coherence REST.

- The `<key-class>` and `<value-class>` elements can either be defined within the `<resource>` element in the REST configuration file or within the `<cache-mapping>` element in the cache configuration file.

- Added support for Coherence*Extend clients to use JCache.

- Added support for Jetty HTTP server to Coherence REST.

- Oracle Traffic Director (OTD) can be used to load balance Coherence*Extend connections.

- Added `SimplePrincipal` which derives from `GenericPrincipal`. `SimplePrincipal` uses the `Identity.Name` property to determine object equality.

- Fixed an issue which could cause the proxy server `TcpProcessor` thread to get stuck in a CPU busy loop.

- Removed the `ConnectionManager` MBean from the name service.

- Fixed a concurrent access issue with UUIDs.

- The `<connect-timeout>` element has been removed from the `<tcp-initiator>` element. It is defined in the `<initiator-config>` element instead.

- Fixed an issue where an extend client doesn't receive delete events from a replicated cache.

- Reduced contention on the proxy service for key-based requests that target the same keys.

1.3.6 Coherence*Web Enhancements and Fixes

- Added a new HTTP session reaping mechanism that uses entry processors to increase performance when deleting sessions.

- Added the ability to have HTTP session caches replicated across federated clusters.

- Fixed an issue where HTTP session attributes are mandated to implement `java.io.Serializable`.

1.3.7 Serialization Framework Enhancements and Fixes

- Added support for Java 8 `Date` and `Time` API to POF.

- Added support for optional types to the `PofReader` and `PofWriter` interfaces.

- Fixed an issue with the POF configuration generator not allowing directories with spaces as a root directory of class files.

- Fixed a POF serialization issue for negative dates with nonzero subseconds.

- Coherence JCache POF ids were moved from 700-799 in previous releases to 610-699 in order to avoid collisions with transaction POF ids.

1.3.8 Configuration Enhancements and Fixes

- Coherence system property names no longer require the `tangosol` prefix.

- The `<port-auto-adjust>` element has been changed to support either a boolean value or an upper bound on the port range.

- Added support for enabling or disabling specified SSL protocols.

- Added the ability to use system properties for element values in configuration files. The syntax is:

  ${*system.property default_value*}

- Fixed an issue with the `partitioned` setting of the backing map. When explicitly set to `false`, it was also setting the in-memory backup partitioned setting to `false`.

- Fixed an issue where resources containing # character in filename or path failed to load.

1.3.9 Managed Coherence Servers Enhancements and Fixes

- Added a new dynamic management mode that elects one of the nodes in the cluster to be the management node. This is the default management mode for managed Coherence servers.

- A GAR file can reference shared libraries.

- A GAR file can be deployed across multiple domain partitions and used by all tenants. Isolation is provided at the domain partition level and is transparent to the application.

- Added the ability override cache properties for each domain partition.

- Added the ability to allow all domain partitions in a cluster to share caches.

- Added the ability to use JCache with managed Coherence servers.

- The coherence REST library (`coherence-rest.jar`) is located on the WebLogic Server classpath by default and does not need to be packaged and deployed as part of an application.

- Fixed an issue where internal session attribute names are returned without stipping the `InternalWLSAttribute` prefix.

- The `MBeanServerFinder` interface has a new method to return the `JMXServiceURL` value for the MBean server.

- Fixed an issue where daemon threads were not using the specified service classloader.

1.3.10 Other Enhancements and Fixes

- Added support for log4j 2.

- Fixed an issue with Windows command scripts that fail to handle `JAVA_HOME` path containing spaces and parenthesis.

1.4 Oracle Coherence for .NET 12*c* (12.2.1)

New features, improvements, and bug fixes have been added to these Oracle Coherence for .NET components:

Coherence*Extend Framework Enhancements and Fixes

Serialization Framework Enhancements and Fixes

Other Enhancements and Fixes

1.4.1 Coherence*Extend Framework Enhancements and Fixes

- .NET clients can now configure POF configuration files in the cache configuration file rather than the application configuration file. Note that a custom serializer must now implement a constructor which initializes the `<init-params>` of the `<serializer>` element in the cache configuration file.

- The default connect time out value is consistent across platforms.

- Added a `SimplePrincipal` which derives from `GenericPrincipal`. `SimplePrincipal` uses the `Identity.Name` property to determine object equality.

- The `<connect-timeout>` element has been removed from the `<tcp-initiator>` element. It is defined in the `<initiator-config>` element instead.

- Fixed an issue with client read and write lock acquisition.

- Fixed an issue where the `SynchronizedDictionary.AcquireWriteLock` method may hide thread interrupts.

- Fixed a concurrent access issue with UUIDs.

1.4.2 Serialization Framework Enhancements and Fixes

- Added support for optional types to the `PofReader` and `PofWriter` interfaces.

- Fixed an issue in the `ValueChangeEventFilter` class to correctly extract the value used by the filter.

1.4.3 Other Enhancements and Fixes

- Fixed an issue where `SimpleMapIndex` instances that use key-based extractors are very slow.

- Fixed an issue where the front cache of a near cache is not updated after all storage nodes are restarted.

- Cache entries are proactively evicted from a backing map after the cache expiry time is reached and no longer require a cache operation to initiate the eviction.

- Coherence system property names no longer require the `tangosol` prefix.

- Added the ability to use system properties for element values in configuration files. The syntax is:

  ```
  ${system.property default_value}
  ```

1.5 Oracle Coherence for C++ 12*c* (12.2.1)

New features, improvements, and bug fixes have been added to these Oracle Coherence for C++ components:

Coherence*Extend Framework Enhancements and Fixes

Serialization Framework and Enhancements Fixes

Other Enhancements and Fixes

1.5.1 Coherence*Extend Framework Enhancements and Fixes

- Added support for STLport on all Solaris distributions.

- Added support for C++11.

- Added support for Windows 64 bit `BackTrace`.

- Added a `Thread::isAlive` method.

- Added new methods to the `coherence::util::Map` interface, as per the Java 1.8 JDK additions. These methods implement the defaults but can be overridden. See the API documentation for `coherence::util::Map` for more information.

- Added property `tangosol.coherence.heap.logging` to enable logging details about the thread local memory allocator hit and miss rate.

- Changed `FinalView InKeySetFilter.f_vSetKeys` to `MemberView m_vSetKeys` as `InKeySetFilter::ensureConverted` requires that `m_vSetKeys` be modifiable.

- The `<connect-timeout>` element has been removed from the `<tcp-initiator>` element. It is defined in the `<initiator-config>` element instead.

- The `Object::toStream` method has been deprecated in favor of `Object::toString`.

- The default value for connect timeout is now consistent across platforms.

- Fixed an issue with getting debug stack traces.

- Fixed an issue where `TreeMap::put` returns the new value instead of the old value.

- Fixed an issue with strict-aliasing compilation warnings that occurred from the use of the -O3 optimization option.

- Fixed an issue with the memory allocator always running with heap padding diagnostics enabled.

- Fixed an issue where dependent threads of a Coherence service may fail to stop during service shutdown.

- Fixed a serialization buffer allocation performance issue.

- Fixed a concurrent access issue with UUIDs.

- Fixed an issue with `String::substring` prematurely ending a substring search.

1.5.2 Serialization Framework and Enhancements Fixes

- Fixed an issue in the `ValueChangeEventFilter` class to correctly extract the value used by the filter.

1.5.3 Other Enhancements and Fixes

- Added support for outputting managed objects to `std::wostream`.

- Fixed an issue where `SimpleMapIndex` instances that use key-based extractors are very slow.

- Fixed an issue where the front cache of a near cache is not updated after all storage nodes are restarted.

- Cache entries are proactively evicted from a backing map after the cache expiry time is reached and no longer require a cache operation to initiate the eviction.

- Coherence system property names no longer require the `tangosol` prefix.

- Added the ability to use system properties for element values in configuration files. The syntax is:

  `${`*system.property default_value*`}`

1.6 Patch Notes for 12*c* (12.2.1.0.1)

New features, changes, and enhancements for the Coherence 12.2.1.0.1 release.

New Features Introduced in this Patch

- No new features have been introduced in this patch.

Changes and Enhancements for Oracle Coherence for Java

- Added federated cache replication support for entries that are loaded by a read-through cache when the backing map is backed by a cache store. For details, see Using Federated Caching with Cache Stores in *Developing Applications with Oracle Coherence*.

- Fixed an issue where the `ReplicateAll` function may fail with a null pointer exception when all of the participants are running on a single machine.

- Fixed an issue where only the last service to start is included in the default service report.

- Fixed a memory leak in the `SafeNamedCache` class in a Coherence*Extend client and proxy with repeated create and destroy operations on a uniquely named cache.

- Fixed an issue where a cluster with an inline `<unicast-listener>` SSL configuration fails to start.

- Fixed an issue where asynchronous API exceptions are not being propagated to a client.

- Fixed an issue with the `com.tangosol.coherence.servlet.AbstractHttpSession.toString()` method.

- Fixed an issue where federation journal records are discarded on active recovery.

- Changed `AsyncNamedCache` operations to return an empty collection instead of `null` when there are no entries.

- Fixed an issue where filter-related calls on a partitioned cache can yield incorrect results.

- Fixed an issue where `FederatedCache` services' log messages are not written when the Coherence logging level is increased at runtime using JMX.

- Fixed an issue causing truncate to not remove persisted data.

- Fixed an issue where persistence and key listeners are not being persisted after a partition is recovered.

- Fixed an issue where an `UnsupportedOperationException` exception can be thrown when calling the `entry.setValue` method from an entry processor

- Fixed a memory leak in the `ContinuousQueryCache` class.

- Added `FederatedPartitionEvent` events (`SYNCING` and `SYNCED`), which can be used to track federated cache `ReplicateAll` operations.

- Improved quorum policy handling to allow for more timely response to timeout based death detection.

- Fixed an issue with the JVisualVM Plugin where the Machine tab and Load Average graph are blank when connected to a cluster using the IBM JDK on AIX.

- Fixed an issue where a Coherence REST application can experience a `com.tangosol.net.RequestPolicyException: No storage-enabled nodes exist for service DistributedCache` exception when undeploying and then re-deploying the application.

- Improved the Java examples so that setting *COHERENCE_HOME* is no longer required.

- Fixed an issue where the `AbstractAggregator.init` method is not called and the `GroupAggregator` class' nested aggregator returns `null`.

- Added instructions in the REST example for Windows to install maven artifacts into the repository.

- Added the ability to sort data tables by any column by clicking on the column header in the JVisualVM Plugin.

- Fixed an issue where severe clock skew can result in cluster join failures.

- Fixed an issue where running CohQL or QueryPlus in silent mode still displays the header for the query results making the output inconsistent with previous versions of Coherence.

- Fixed an issue where Coherence can hang during configuration processing due to a recursive macro value in system properties (that is, `-Dtangosol.pof.enabled= ${tangosol.pof.enabled}`).

- Updated the REST example to use the preferred approach of using the `EventInterceptor<LifecycleEvent>` interface to bootstrap application data.

- Fixed an issue where unnecessary `unregister MBean` messages are sent to a managed node after a cache is destroyed

- Fixed an issue where classes for user types are not found at runtime when a Coherence REST application is deployed to WebLogic Sever. Coherence REST with POF serialization can safely be used in WebLogic Server without any errors.

- Added a `cache-mapping/federated` element to indicate that the matching caches should not be replicated to remote clusters by `FederatedCache` services.

- Updated all `PofWriter.writeCollection` methods to have consistent signatures.

- Fixed an issue where the `ensureCache` method performs access control checks in the absence of a security manager.

- Fixed an issue where an `IndexOutOfBoundsException` exception is thrown on `ChainedRequest` modification.

- Fixed an issue where standard copyright headers are not applied to the Java, C++ and .NET source code examples.

- Updated the Coherence JVisualVM Plug-in user interface to more closely align with other JVisualVM Plug-ins.

- Fixed an issue where Coherence security examples fail to run on Windows.

- Fixed an issue where a deadlock can occur during dynamic pool downsizing.

- Removed resolving Coherence configuration element names against system properties; that is, the `autostart` system property does not override the Coherence configuration element `autostart`.

- Fixed an issue where the `ReportTime` column is missing in the `report-cache-size.xml` file.

Changes and Enhancements for Oracle Coherence for .NET

- Fixed a memory leak in the `ContinuousQueryCache` class.

- Fixed a concurrency issue with `CacheFactory` methods.

- Added the ability to override XML values in Coherence configuration files using the `system-property` attribute (similar to Coherence for Java). These properties can either be specified as environment variables or with `property` elements in the `<coherence>` section of the application configuration. For example:

```
<property name="coherence.address" value="myserver"/>
```

The default cache configuration file now includes the following properties: `coherence.profile`, `coherence.ns.address`, and `coherence.ns.port`.

Changes and Enhancements for Oracle Coherence for C++

- Fixed an issue where a `LimitFilter` implementation that uses a POF extractor can throw an `UnsupportedOperationException` exception.

- Fixed a memory leak in the `ContinuousQueryCache` class.

- Fixed an issue with UTF-8 string validation.

2

Known Problems and Workarounds

known problems at the time of release.

This chapter includes the following sections:

2.1 Changing the Partition Count When Using Active Persistence

The partition count cannot be changed when using active persistence. If you change a services partition count, then on restart of the services all active data is moved to the persistence trash and must be recovered after the original partition count is restored. Data that is persisted can only be recovered to services running with the same partition count.

Ensure that the partition count is not modified if active persistence is being used. If the partition count is changed, then a message similar to the following is displayed when the services are started:

```
<Warning> (thread=DistributedCache:DistributedCachePersistence, member=1):
Failed to recover partition 0 from SafeBerkeleyDBStore(...); partition-count
mismatch 501(persisted) != 277(service); reinstate persistent store from
trash once validation errors have been resolved
```

The message indicates that the change in the partition-count is not supported and the current active data has been copied to the trash directory. To recover the data:

1. Shutdown the entire cluster.

2. Remove the current active directory contents for the cluster and service affected on each cluster member.

3. Copy (recursively) the contents of the trash directory for each service to the active directory.

4. Restore the partition count to the original value.

5. Restart the cluster.

2.2 Inconsistent Data in Federated Caches During Snapshot Recovery

There is a known issue that can result in inconsistent data when using federated caching together with cache persistence. Currently, a recovered snapshot of a service is not propagated to federated clusters. The cached data on the originating cluster is recovered, but the cache data on the destination cluster may still contain the cached data that was present prior to the recovery. Support for this use case is scheduled for the next patch release.

2.3 ClassNotFoundException with Coherence REST on WebLogic Server

There is a known issue where classes for user types are not found at runtime when a Coherence REST application is deployed to WebLogic Sever and `coherence-rest.jar` is on the server classpath.

To work around this issue:

1. Rename the *COHERENCE_HOME*`/lib/coherence-rest.jar` library to `coherence-ext-rest.jar`.

2. Include the *COHERENCE_HOME*`/lib/coherence-ext-rest.jar` library in the application classpath so the application compiles.

3. Package the `coherence-ext-rest.jar` library in the Coherence REST application WAR module.

4. Deploy the Coherence REST application.

2.4 Impact of Generics

- Care should be taken to ensure that federated caches across a federation are configured to use the same types as no runtime-type-checking is performed between clusters in a federation.

- Care should be taken to ensure that recoverable caches are consistently configured to use the same types during restarts as no runtime-type-checking is performed.

2.5 Support for JVisualVM Plugin

The Coherence JvisualVM Plugin is supported for Coherence 3.7.1.X and above. Using the plugin to connect to older clusters is not supported.

3

Deprecated and Desupported Features

A listing of features that have been deprecated and desupported.

This chapter includes the following sections:

Deprecated Features

Desupported Features

3.1 Deprecated Features

Coherence features that are deprecated.

BACKUP CACHE and RESTORE CACHE

Managed Coherence Server MBean Attributes

Coherence*Web Container Support

ActiveCache (active-cache.jar)

Thread Count

Specifying Ports in the WKA List

Specifying tangosol in System Properties

TopLink Implementations

Object::toStream Deprecated

ParallelAwareAggregator

3.1.1 BACKUP CACHE and RESTORE CACHE

The `BACKUP CACHE` and `RESTORE CACHE` commands in CohQL are deprecated. A new set of snapshot commands has been provided for use with the new the persistence feature. For details, see "Persisting Cache Data to Disk" in *Developing Applications with Oracle Coherence*.

3.1.2 Managed Coherence Server MBean Attributes

The following MBean attributes are deprecated in Managed Coherence Server:

- `CoherenceClusterParamsBean.UnicastListenPort`

- `CoherenceClusterParamsBean.UnicastPortAutoAdjust`

- `CoherenceClusterParamsBean.MulticastListenPort`

- `CoherenceClusterWellKnownAddressBean.ListenPort`

The following deprecated MBean attribute is removed:

- `CoherenceClusterParamsBean.UnicastListenAddress`

3.1.3 Coherence*Web Container Support

Coherence*Web no longer supports the following web containers: Apache Tomcat 5.5.*n*, Apache Tomcat 6.0.*n*, Caucho Resin 3.1.*n*, IBM WebSphere 5.*n*, IBM WebSphere 6.*n*, IBM WebSphere 7.*n*, Sun GlassFish 2.*n*, Sun Application Server 8.*n*, Oracle OC4J 10.1.3.*n*, Oracle OC4J 10.1.2.*n*, Oracle GlassFish 3.*n*, Oracle GlassFish 4.*n*, Jetty 6.1.*n*, Jetty 5.1.*n*, JBoss Application Server. Applications that require Coherence HTTP session management must be migrated to use a supported web container version. For details, See *Administering HTTP Session Management with Oracle Coherence*Web*.

3.1.4 ActiveCache (active-cache.jar)

ActiveCache (`active-cache.jar`) has been deprecated. ActiveCache can still be used with applications that have been developed to run on older versions of WebLogic Server.

ActiveCache functionality has been replaced by Managed Coherence Servers. For more information on Managed Coherence Servers, see *Oracle Fusion Middleware Developing Oracle Coherence Applications for Oracle WebLogic Server*.

3.1.5 Thread Count

The `<thread-count>` element is deprecated. Use the `<thread-count-min>` and `<thread-count-max>` elements instead.

3.1.6 Specifying Ports in the WKA List

The functionality to specify ports in the WKA list is deprecated as of the of the 12.2.1 release. Support for this feature will be removed in a future release.

3.1.7 Specifying `tangosol` in System Properties

Coherence system property names no longer require the `tangosol` prefix. For example, the system property `tangosol.coherence.distributed.localstorage` can now be written as `coherence.distributed.localstorage`. System properties that only contained the `tangosol` prefix now use the `coherence` prefix. For example, the system property `tangosol.pof.enabled` can now be written as `coherence.pof.enabled`. The changes are also applicable to Unix-based environments. For example `TangosolCoherenceCacheConfig` can be written as `CoherenceCacheConfig`.

System properties that include the `tangosol` prefix are still supported; however, support may be removed in a future release.

3.1.8 TopLink Implementations

The `TopLinkCacheLoader` and `TopLinkCacheStore` implementations are deprecated as of the of the 12.2.1 release.

3.1.9 Object::toStream Deprecated

The `Object::toStream` method has been deprecated. Applications should use the `Object::toString` method instead.

3.1.10 ParallelAwareAggregator

The `ParallelAwareAggregator` interface has been deprecated and should no longer be used. Applications should use the `StreamingAggregator` interface to implement custom aggregators. For details, see "Performing Data Grid Aggregation Using Streams" in *Developing Applications with Oracle Coherence*.

3.2 Desupported Features

This section describes features that have been removed in this release.

Encryption Network Filters

3.2.1 Encryption Network Filters

Encryption network filters, which were deprecated in Coherence 3.7, have been removed and are no longer supported as of the 12*c* (12.2.1) release. TCMP inter-cluster and *Extend TCP connections should be secured using SSL instead. The following APIs have been removed:

- `com.tangosol.net.security.AbstractEncryptionFilter.java`

- `com.tangosol.net.security.AsymmetricEncryptionFilter.java`

- `com.tangosol.net.security.BlockCipherInputStream.java`

- `com.tangosol.net.security.BlockCipherOutputStream.java`

- `com.tangosol.net.security.ClusterEncryptionFilter.java`

- `com.tangosol.net.security.PasswordBasedEncryptionFilter.java`

- `com.tangosol.net.security.SymmetricEncryptionFilter.java`

Index

M

My Oracle Support, *1-1*

S

software patches, *1-1*