Glossary

About Terracotta Documentation

This documentation is about Terracotta DSO, an advanced distributed-computing technology aimed at meeting special clustering requirements.

Terracotta products without the overhead and complexity of DSO meet the needs of almost all use cases and clustering requirements. To learn how to migrate from Terracotta DSO to standard Terracotta products, see Migrating From Terracotta DSO. To find documentation on non-DSO (standard) Terracotta products, see Terracotta Documentation. Terracotta release information, such as release notes and platform compatibility, is found in Product Information.

Release: 3.6
Publish Date: November, 2011 **Documentation Archive »**

Glossary of Terms

A

Autolock

A cluster-wide lock on a clustered object wherever there is Java synchronization on that object. **>> Read more**

B

BeanShell

A scripting language similar to ordinary Java code which does not require type declarations and which can be executed at runtime. BeanShell has been standardized in JSR-274. More information on BeanShell is at http://www.beanshell.org/.

C

Class Expression

An element in the Terracotta configuration file allowing class sharing. For example, the following code makes the class `Counter` shareable:

```
<application>
  <dso>
     ...
      <instrumented-classes>
        <include>
          <class-expression>Counter</class-expression>
        </include>
      </instrumented-classes>
     ...
  </dso>
</application>
```

Client Disconnect

An action taken by a Terracotta server to drop a Terracotta client from a cluster when the server has detected a severed connection to that client.

Cluster

See Terracotta Cluster.

Clustered Lock

A lock on a clustered object. A clustered lock allows access to data in a shared object.

Clustered Object

An object referenced by a Terracotta root directly or by another object referenced by the root. Clustered objects are shared across the Terracotta cluster and can be safely accessed and modified.

Concurrent Lock

A non-exclusive lock whose use is not communicated to the Terracotta server. A concurrent lock allows a Terracotta transaction to proceed, but there is no guarantee of correctness because two or more threads could write to an object concurrently.

D

DSO

Distributed Shared Objects (DSO) is the Terracotta component allowing an application to operate in a multi-JVM environment.

G

Greedy Lock

A lock issued to one Terracotta client based on that client's heavy usage of that lock. The Terracotta server prevents other clients from acquiring the lock. The Terracotta server can reacquire the lock if the owning client dies, or if other clients experience a negative impact from the inability to acquire the lock.

L

L1

An informal name for a **Terracotta client**. The terminology is borrowed from Level 1/Level 2 cache terminology in computer hardware.

L2

An informal name for a **Terracotta server**. The terminology is borrowed from Level 1/Level 2 cache terminology in computer hardware.

Literal

A field whose value corresponds to an actual value, as opposed to a fields whose value is a reference. Terracotta considers all non-reference fields to be literals.

Lock

Locks in Terracotta perform two duties: to coordinate access to critical sections of code between threads and to serve as boundaries for Terracotta transactions. Terracotta locks are analogous to synchronization in Java. **>> Read more**

Lock Upgrade

An attempt by a thread to exchange its read lock for a write lock.

Logically Managed Class

A class in which field changes are propagated by replaying method calls on the other members of the Terracotta cluster. These classes of objects are described as "logically managed" because Terracotta records and distributes the logical operations that were performed on them rather than changes to their internal structure. **>> Read more**

M

Method Expression

An element in the Terracotta configuration file allowing method clustering through locks. For example, the following code makes the class `count` lockable:

```
<application>
  <dso>
     ...
      <locks>
       <autolock>
          <method-expression>void Counter.count()</method-expression>
          <lock-level>write</lock-level>
       </autolock>
      </locks>
     ...
  </dso>
</application>
```

N

Named Lock

A lock specified in a "named lock" stanza in the Terracotta configuration A thread that attempts to execute a method that matches a named lock stanza must first acquire the lock of that name from the Terracotta server. Named locks are very coarse-grained and should only be used when autolocks are not possible. **>> Read more**

Non-Persistent Mode

A setting that forces Terracotta servers to use disk space only for swapping data. Found in the Terracotta configuration file under servers/server/dso /persistence. A Terracotta server in non-persistent mode writes data to disk on a temporary basis, and if restarted will **not** be restored to its pre-shutdown state.

O

On-load

A set of specific startup values and behaviors, specified in the Terracotta configuration file. For example, the on-load section can specify a method or code for a particular class included for instrumentation:

```
<include>
...
   <on-load>
    <execute><![CDATA[...]]></execute>
    <method>myMethod</method>
   </on-load>
</include>
```

P

Persistent Mode

A data-preservation setting for Terracotta servers, found in the Terracotta configuration file under servers/server/dso/persistence. A Terracotta server in persistent mode immediately and permanently writes data to disk, and if restarted will reload that data to restore its pre-shutdown state0. If more than one Terracotta server is used in a cluster, persistent mode must be used.

Physically Managed Class

Physically Managed Classes are those in which Terracotta records and distributes changes to the physical structure of the object. When a field of a physically managed object changes, the new value of the field is sent to the Terracotta server and to other members of the Terracotta cluster that currently have the changed object in memory. **>> Read more**

Primitive

A primitive data type such as an int or boolean, and including primitive wrappers like Long and Integer. Terracotta roots defined as a primitive can be freely initialized and mutated. Compare to a root of reference type.

R

Read Lock

A lock set to *read* level, allowing access to data for read operations. A read lock is non-exclusive, so that multiple reads may occur on the same data. A read lock cannot grant read access to data which is under a write lock until the write lock is lifted.

Reconnect Window

A configurable interval of time during which Terracotta clients are allowed to reconnect to the Terracotta server after the server has restarted.

Root

A "root" is the top of a clustered object graph as declared in the Terracotta configuration. Object sharing in Terracotta starts with these declared roots. **>> Read more**

S

Shared Lock

A lock on a shared object, allowing safe access to that object.

Shared Object

An object shared across a Terracotta cluster.

T

Terracotta Client

A Terracotta client is a JVM that participates in a Terracotta cluster and that your application runs in. **>> Read more**

Terracotta Cluster

A Terracotta cluster consists of one or more Terracotta servers along with one or more Terracotta clients working together as if in a single JVM. The servers ensure failure recovery, maintain data coherence, and provide other services to clients.

Terracotta Server

The Terracotta server is the heart of a Terracotta cluster. It performs two basic functions:

1. Cluster-wide thread coordination and lock management
2. Clustered object data management and storage

>> Read more

Transactions

Terracotta transactions are sets of clustered object changes that must be applied atomically. Transactions are bounded by lock acquisition and release. **>> Read more**

Transaction Boundary

The point at which a lock acquisition or release occurs. A Terracotta transaction begins at lock acquisition and ends at lock release.

Transient (Terracotta Transience)

Terracotta Transience is a mechanism to allow certain fields to be skipped during sharing, so that a few non-sharable objects do not prevent you from sharing other portions of an object graph. Terracotta also allows you to automatically run various methods or snippets of code when an object is loaded so that you can assign appropriate values to transient fields. **>> Read more**

W

Write Lock

A lock set to *write* level, allowing access to data for read/write operations. A write lock is exclusive, so that only one write operation can occur on the same data while the write lock is in effect. In the Terracotta configuration file, locks defined without a level are by **default** write locks. A write lock cannot grant write access to data which is under a read lock until the read lock is lifted.