

Win an iPad 2

**Visit the Terracotta booth to sign up
for a chance to win!**



High-Performance Scalability with Enterprise Ehcache



Presented by: Eric Mizell, Terracotta
emizell@terracotta.org

Agenda

- Standard Caching Theory & Practice
- Ehcache API & Configuration
- Scale Up With BigMemory
- Advanced Scaling Techniques
- Search API
- Developer's Console
- Q&A



Standard Caching Theory & Practice

Why Cache?

- Caching is the most widely used strategy for improving application performance
- A cache stores data so that future requests for that data can be served fast
- Caches speed data access by eliminating network and database operations



Benefits of Caching

- **Speed:** Increase application performance to better meet business and customer expectations
- **Costs:** Reduce hardware, database, operating and support costs
- **Scalability:** Improve ability to respond to growing data and processing needs



Caching Strategies

- Know your data
 - Access patterns
 - User experience
 - Read/Write ratios
- Best data to cache
 - Static data
 - Data that is reused regularly
 - Data that is expensive to retrieve
- Eviction Policies
 - Eternal is best solution when data is changed in process
 - TTL & TTI
 - LRU, LFU, FIFO



Ehcache API & Configuration

About Ehcache

- Founded in 2003
- Apache 2.0 License
- Integrated by lots of projects, products
- Hibernate Provider implemented 2003
- Web Caching 2004
- Distributed Caching 2006
- JCACHE (JSR107) implementation 2007
- REST and SOAP APIs 2008
- SourceForge Project of the Month March 2009
- Acquired by Terracotta 2009; Integrated with Terracotta Server
- Ehcache 2.0 and Enterprise Ehcache March 2010
- Forrester Wave “Leader” May 2010
- BigMemory Sept 2010



Snap Into Existing Applications

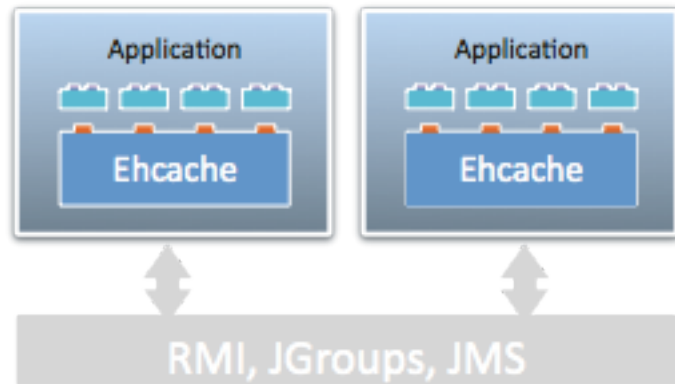
- Simple API honed by 100,000's of production deployments

```
Cache cache = manager.getCache("sampleCache1");  
Element element = new Element("key1", "value1");  
cache.put(element);
```

- Default cache for popular frameworks
 - Hibernate, iBatis, Open JPA
 - Spring (Annotations)
 - Grails
 - JRuby
 - Liferay
 - Cold Fusion



Ehcache Standalone & Replication



- Up to 8GB in process
- Up to 20GB on disk
- Replicated distribution up to 20 nodes
- Not coherent
- Not transactional
- No HA

Ehcache.xml for standalone deployment

```
<ehcache>
  <defaultCache
    maxElementsInMemory="10000"
    eternal="false"
    timeToLiveSeconds="120"
    memoryStoreEvictionPolicy="LFU"/>

  <cache name="WheelsCache"
    maxElementsInMemory="500"
    maxElementsOnDisk="10000"
    timeToLiveSeconds="3600"
    memoryStoreEvictionPolicy="LFU"/>

  <cache name="CarCache"
    maxElementsInMemory="10000"
    timeToldleSeconds="300"
    memoryStoreEvictionPolicy="LFU"/>
</ehcache>
```



Common Ehcache Uses

- **Hibernate / JPA, and DAO / JDBC**
 - Using Hibernate H2LC provider
 - Spring Annotations
 - Using General API
- **Web Caching**
 - Ehcache-web module provides page and fragment and filters
 - Cache anything that you can generate in a web container: HTML, XML, JSon, images, binary files ...
 - Scalable due to BlockingCache
 - Use or Subclass SimpleCachingFilter or SimplePageFragmentCachingFilter
 - Configure a filter and an URL mapping in web.xml and a cache entry matching the filter name



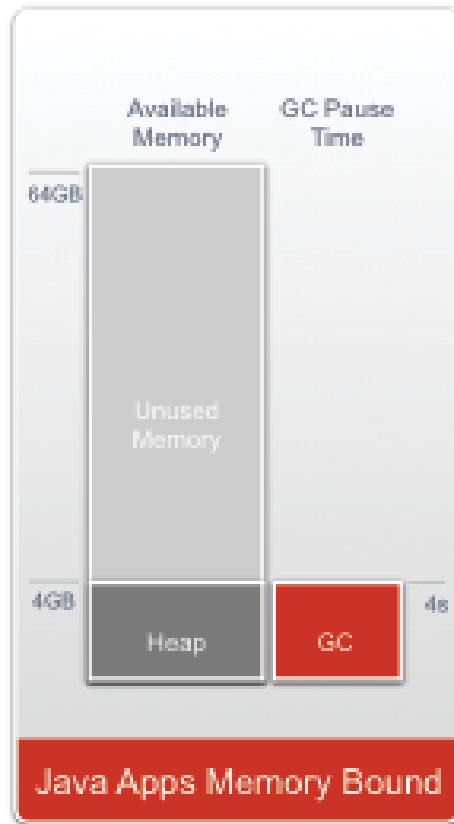
Ehcache API's

- **NonStopCache** – allows your application to keep running in the event of a disruption in your caching service
- **UnlockedReadsView** – unlocked reads of data when SLA's allow eventual consistent data
- **Explicit Locking** – when your application needs fine grain locking on data
- **Write Behind** – persist your data asynchronously
- **Event Listeners** – register callback methods that will be executed when a cache event occurs

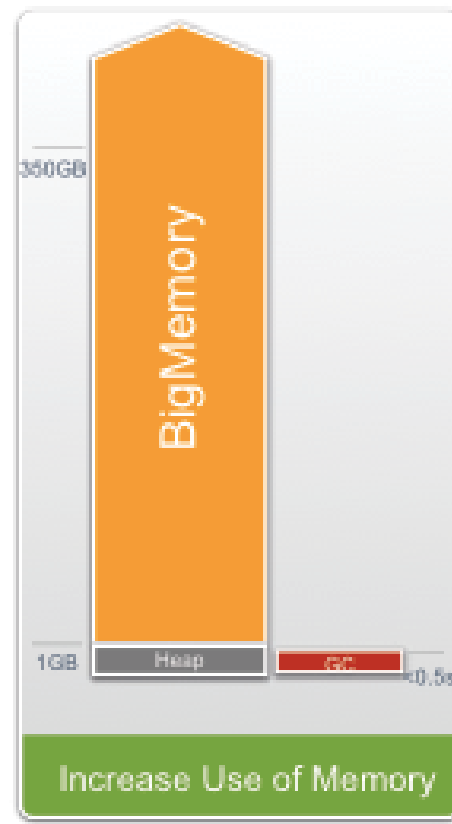


Scale Up With BigMemory

BigMemory for Enterprise Ehcache



Without BigMemory, Java apps are memory bound.



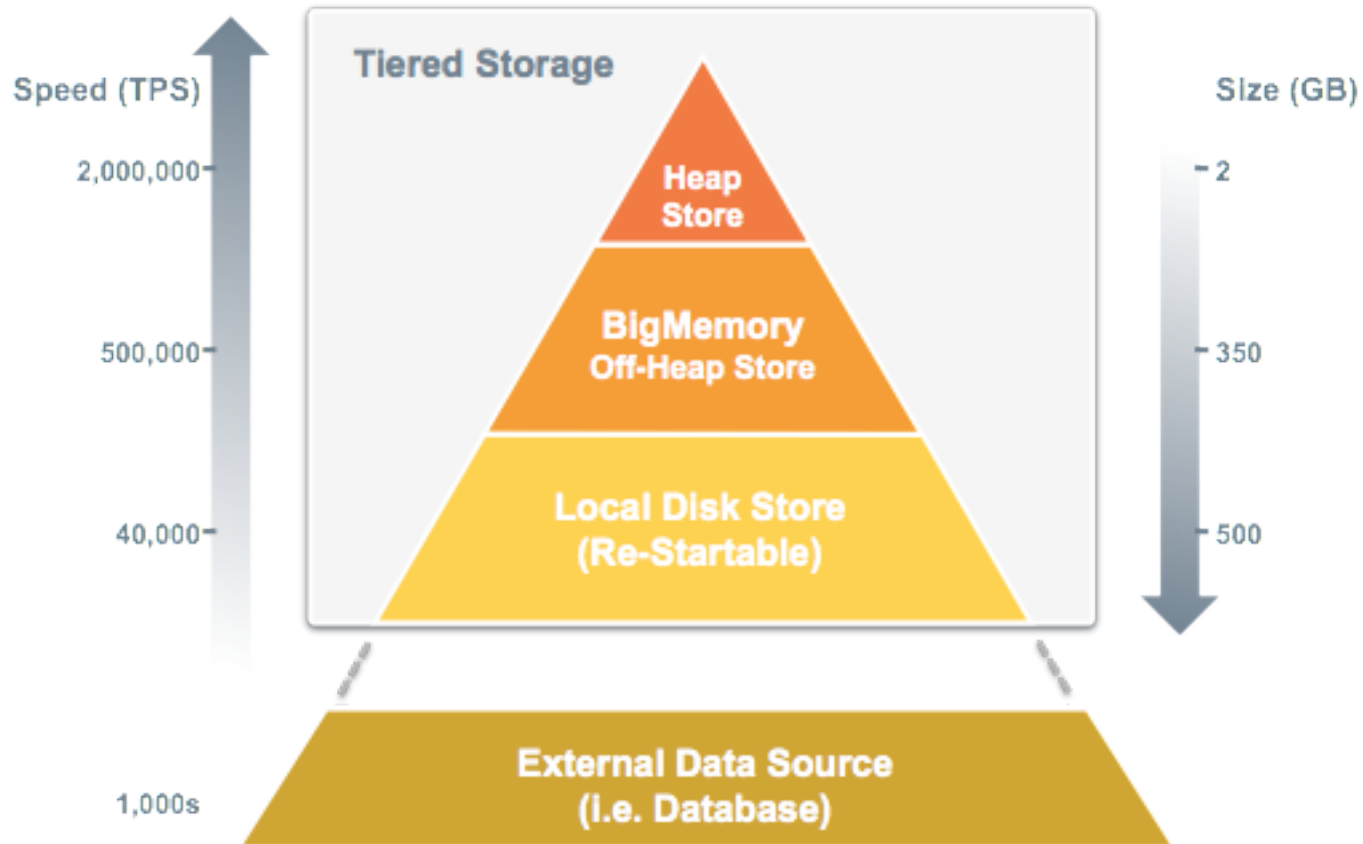
With BigMemory, a single JVM uses all available memory.

Sample ehcache.xml for standalone BigMemory

```
<ehcache>
  <defaultCache maxElementsInMemory="10000"
    eternal="true"
    memoryStoreEvictionPolicy="LRU"
    statistics="false" />
  <cache name="BigCache"
    maxElementsInMemory="10000"
    eternal="true"
    memoryStoreEvictionPolicy="LRU"
    overflowToOffHeap="true"
    maxMemoryOffHeap="4G" />
</ehcache>
</ehcache>
```



Ehcache Tiered Storage



Advance Scaling Techniques

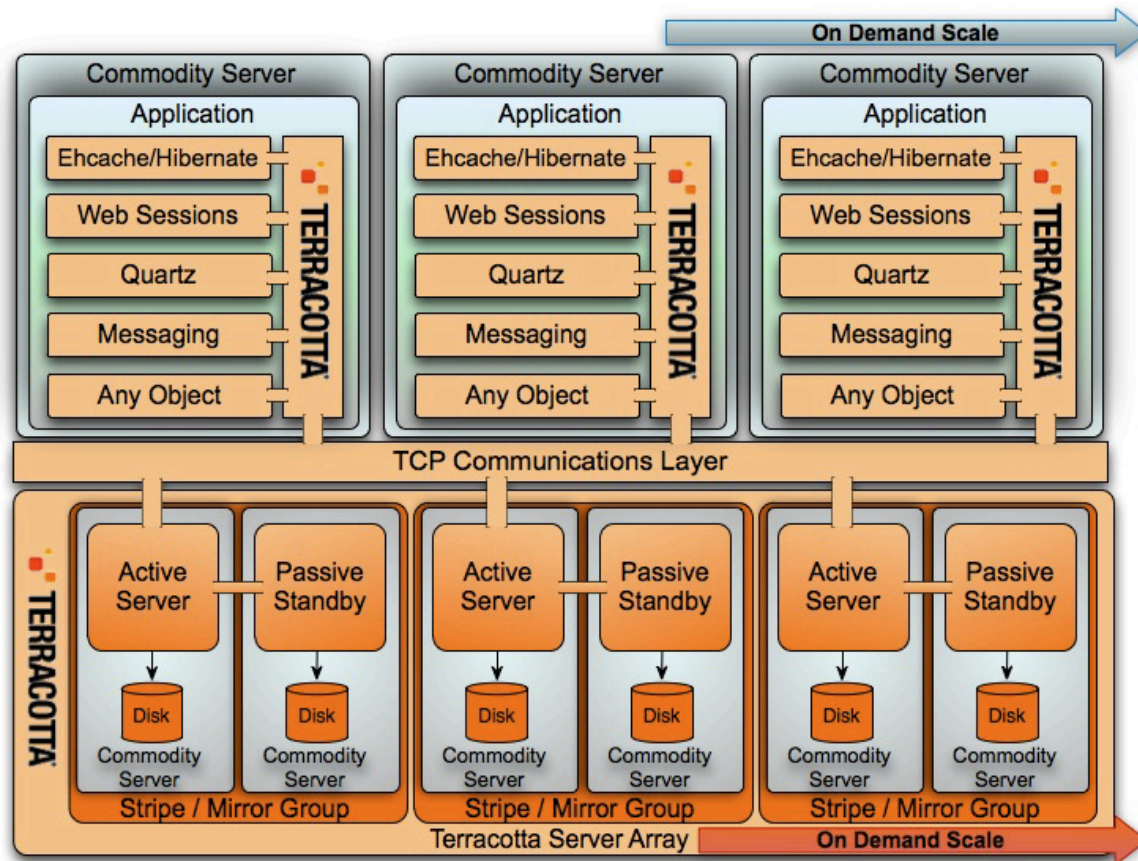
Terracotta Platform

- Enterprise class data management
 - Clustering, Distributed Caching
 - Highly Available (99.999)
 - Linear Scale Out
 - BigMemory - More scalability with less Hardware
 - ACID, Persistent to Disk (& SSD)
 - Ease of Operations
 - Flexibility with CAP tradeoffs



Terracotta Platform

- Fast & efficient
 - Ensures coherent data
 - Efficient & Smart Replication
 - Dynamic Data-locality
 - Cluster-wide coordination
- Reliable, available & scalable
 - Disk based persistence
 - Mirroring for high availability
 - Striping for linear scale



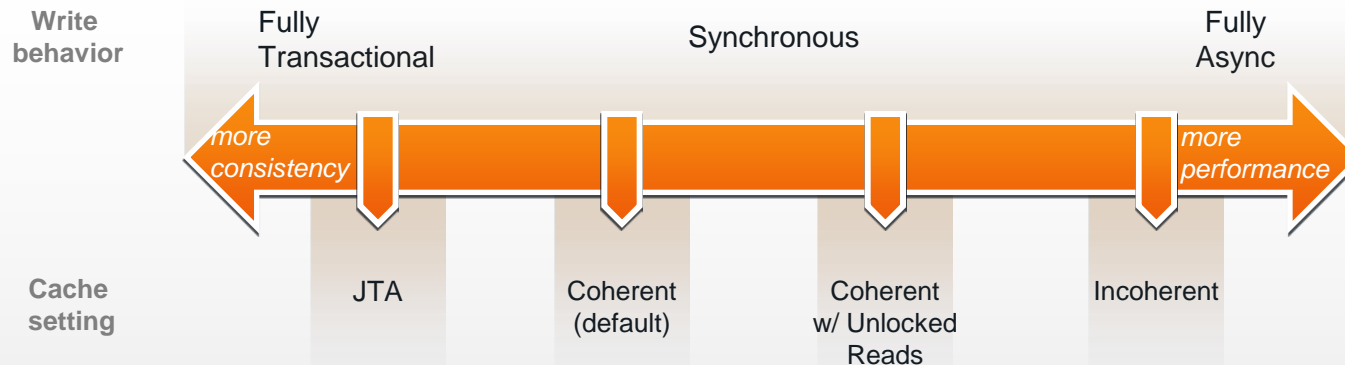
Snap In Enterprise Data Management

- Simply change two lines of configuration

```
<ehcache>
  <terracottaConfigUrl="someserver:9510"/>
  <defaultCache
    maxElementsInMemory="10000"
    eternal="false"
    timeToLiveSeconds="120" />
  <cache name="Pets"
    maxElementsInMemory="10000"
    timeToLiveSeconds="3000">
    <terracotta/>
  </cache>
</ehcache>
```



Enterprise Ehcache Operates Across the Entire Consistency-Performance Spectrum



```
<cache name="UserPreferencesCache"
  maxElementsInMemory="10000"
  timeToIdleSeconds="300"
  memoryStoreEvictionPolicy="LFU">
  <terracotta clustered="true"
  consistency="eventual" />
</cache>
```

```
<cache name="ShoppingCartCache"
  maxElementsInMemory="10000"
  timeToIdleSeconds="300"
  memoryStoreEvictionPolicy="LFU">
  <terracotta clustered="true"
  consistency="strong" />
</cache>
```

Search API

Enterprise Ehcache Search

- Full featured Search API
- Any attribute in the Value Graph can be indexed
- Supports large indices on BigMemory
- Time Complexity
 - $\log(n/\text{number of stripes})$
- Intuitive Fluent API
 - E.g. Search for 32 year old males and return the cache keys.
Results results = cache.createQuery().includeKeys()
 .addCriteria
 (age.eq(32))
 .and
 (gender.eq("male"))
 .execute();



Enterprise Ehcache Search

■ Make a cache searchable

```
<cache name="cache2" >  
  <searchable metadata="true"/>  
</cache>
```

■ What is searchable?

- Element keys, values and metadata, such as creation time
 - Attribute types: Boolean, Byte, Character, Double, Float, Integer, Long, Short, String, Date, Enum
 - Metadata: creationTime, expirationTime, lastAccessTime, lastUpdateTime, version

■ Specify attributes to index

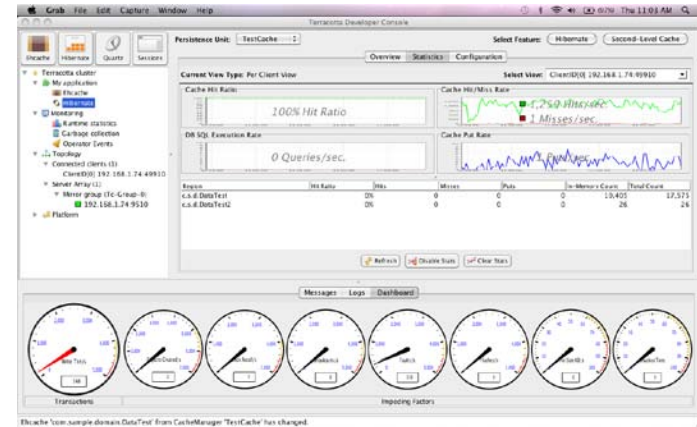
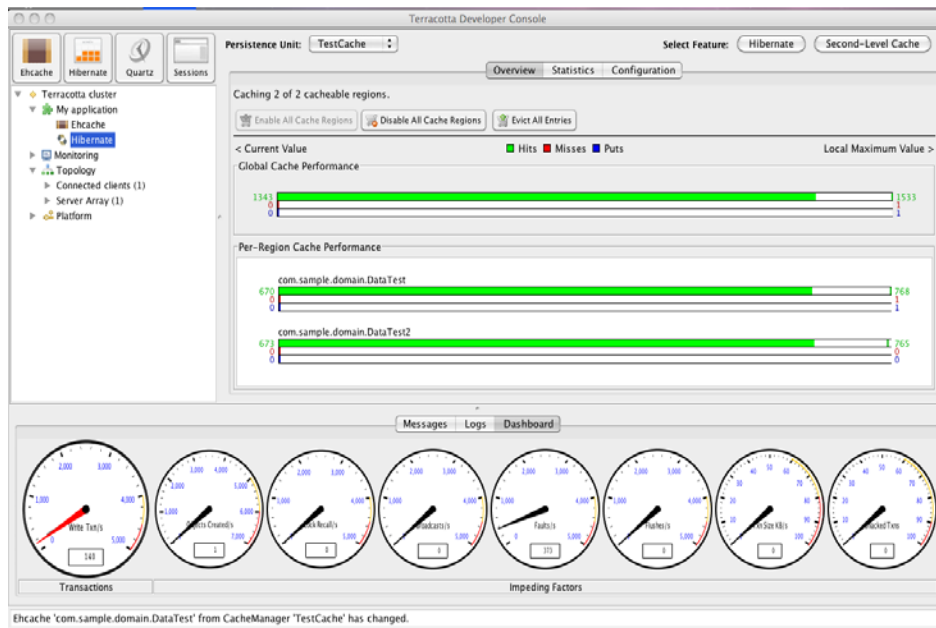
```
<cache name="cache2" maxElementsInMemory="10000" >  
  <searchable>  
    <searchAttribute name="age" class="net.sf.ehcache.search.TestAttributeExtractor"/>  
    <searchAttribute name="gender" expression="value.getGender()"/>  
  </searchable>  
</cache>
```



Developer's Console

Enterprise-class Monitoring for Ehcache

Dramatically simplifies tuning and operations, and shows the database offload.



- Real-time view of caches:
 - Cache hit / miss rates & ratios
 - Hits on the database
 - Cache puts
 - Efficiency of cache regions
 - Clear Caches on Demand.
- Dynamic cache parameters



Monitoring and Visibility

The image displays two overlapping web-based monitoring interfaces. The top interface is the EHCACHE Console, and the bottom is the Terracotta Developer Console.

EHCACHE Console

EHCache Console
a product from **TERRACOTTA**

Documentation | Licensing & Support
No license key detected

Cache Managers | Statistics | Configuration | Contents | **Cache Efficiency** | Memory Use | API

Efficiency for cache manager `192.168.1.48:60051:CacheManager@2bb5340c` and cache `o.t.s.r.l.p.t.Charters Towers`

Chart Settings

Cache Hits/Total Hits (%)

Time	Cache Hit Ratio (%)
15:29:35	40
15:29:45	35
15:29:55	55
15:30:05	45

Back

Terracotta Developer Console

Persistence Unit: `TestCache` Select Feature: `Hibernate` `Second-Level Cache`

Overview | Statistics | Configuration

Caching 2 of 2 cacheable regions.

Enable All Cache Regions | Disable All Cache Regions | Evict All Entries

< Current Value | Hits Misses Puts | Local Maximum Value >

Global Cache Performance

Metric	Value	Local Maximum Value
Hits	1172	6
Misses	6	6
Puts	6	6

Per-Region Cache Performance

Region	Hits	Misses	Puts	Local Maximum Value
com.sample.domain.DataTest	582	6	6	6
com.sample.domain.DataTest2	590	0	0	0

Messages | Logs | Dashboard

Transactions

Metric	Value
Write Txn/s	135
Objects Created/s	3
Object Recall/s	0
Evictions/s	0

Impeding Factors

Metric	Value
Faults/s	297
Flushes/s	2
Pool Size KB/s	0
Rejected Txns	0

Q & A

Stop by the Terracotta booth to sign up for the iPad 2
The drawing is during lunch.