



EHCACHE

wotif.com

JavaOne

Distributed Caching Using the JCACHE API and ehcache, Including a Case Study on Wotif.com

Greg Luck

Maintainer, ehcache—<http://ehcache.sf.net>

Member, JSR 107 JCACHE Expert Group

Chief Architect, Wotif.com—<http://wotif.com>

TS-6175

Goal of This Talk

What You Will Gain

Learn how to:

- Use ehcache to make your apps fast
- Use General, Hibernate, and Web caches
- Configure caches as distributed caches for horizontal scaling
- Abstract the Cache Provider with Java™ Specification Request (JSR) 107 JCache
- Apply ehcache to real-world problems

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

Introduction

About ehcache

- Since 2003
- Probably the most widely used Java platform Cache
- Apache 2.0 License
- Integrated with Spring and Hibernate
- Java Development Kit (JDK™) 1.4 and higher (from ehcache 1.2.4)
- Web Caching, Persistence Caching, General Purpose and Cache Constructs
- Distributed Caching implementation in 2006
- JCACHE implementation released 2007

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

Make Your Application 10–1,000 Times Faster

How much faster will caching make an application?

- It depends on a multitude of factors being:
 - How many times a cached piece of data can and is reused by the application
 - The proportion of the response time that is alleviated by caching
 - In applications that are I/O bound, which is most business applications, most of the response time is getting data from a database; Therefore the speed up mostly depends on how much reuse a piece of data gets

Make Your Application 10–1,000 Times Faster

Cache Efficiency

- Factors which affect the efficiency of a cache are:
 - Required Liveness of Data
 - Proportion of total data cached
 - Read/Write ratio
 - Caching in a single VM vs. Caching Clusters

Make Your Application 10–1,000 Times Faster

How to calculate entire system speedup: Amdahl's Law

- Amdahl's law, after Gene Amdahl, is used to find the system speed up from a speed up in part of the system
 - $1 / ((1 - \text{Proportion Sped Up}) + \text{Proportion Sped Up} / \text{Speed up})$
- Things to Watch:
 - For a web application the “system” should include browser render time and network latency



Make Your Application 10–1,000 Times Faster

Speed up from a Web Page Cache

Uncached page time: 2 seconds

Cache retrieval time: 2ms

Proportion: 100%

The expected server-side system speedup is thus:

$$1 / ((1 - 1) + 1 / 1000)$$

$$= 1 / (0 + .001)$$

$$= 1,000 \text{ times system speedup}$$

The to the browser “system” speedup is much less

Make Your Application 10–1,000 Times Faster

Speed up from a Database Level Cache

Uncached page time: 2 seconds

Database time: 1.5 seconds

Cache retrieval time: 2ms

Proportion: 75% (1.5/2)

The expected system speedup is thus:

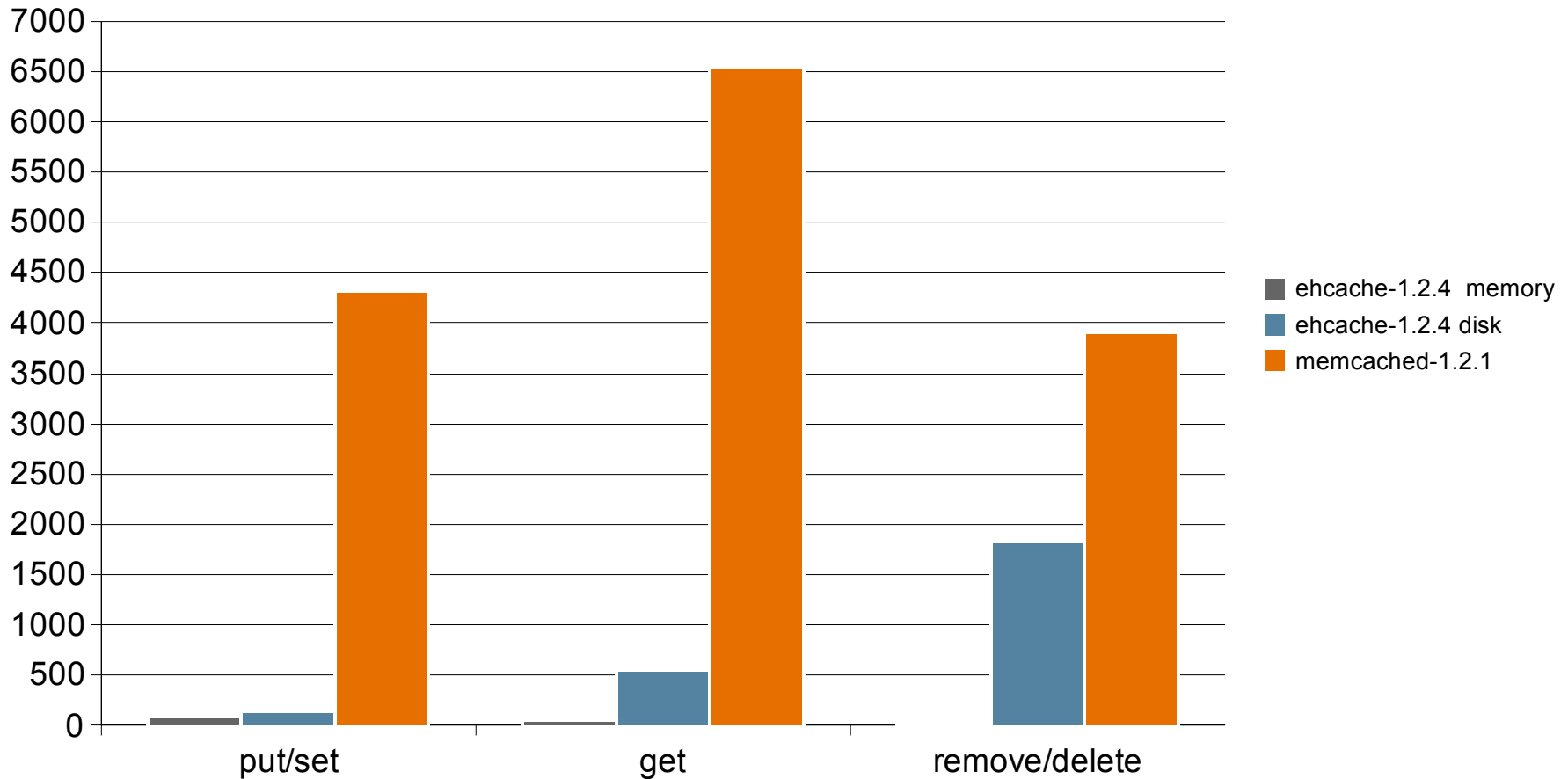
$$1 / ((1 - .75) + .75 / (1500/2))$$

$$= 1 / (.25 + .75/750)$$

$$= 3.98 \text{ times system speedup}$$

Make Your Application 10–1,000 Times Faster

Speed compared with Memcached (smaller is better)



Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

General Purpose Caching

Step by Step

1. Add ehcache Java Archive (JAR) to your classpath
2. Place configuration in ehcache.xml and add it to your classpath
3. Create a CacheManager
`CacheManager manager = CacheManager.create();`
4. Reference a Cache
`ehcache sampleCache = manager.getCache();`
5. Use it
`sampleCache.put(new Element("key", "value"));`
`sampleCache.get("key");`

General Purpose Caching

ehcache.xml Configuration

```
<ehcache>
```

```
  <diskStore path="java.io.tmpdir"/>
```

```
  ...
```

```
  <cache name="sampleCache1"  
    maxElementsInMemory="10000"  
    maxElementsOnDisk="1000"  
    eternal="false"  
    overflowToDisk="true"  
    timeToIdleSeconds="300"  
    timeToLiveSeconds="600"  
    memoryStoreEvictionPolicy="LFU"  
  />
```

```
</ehcache>
```

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

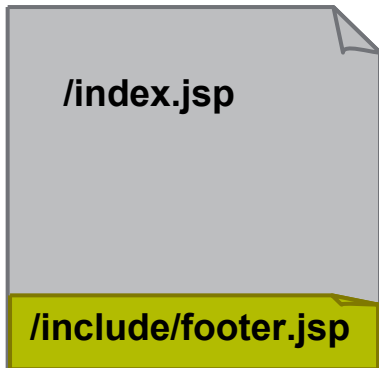
Web Caching

Step by Step

1. Use or Subclass SimpleCachingFilter or SimplePageFragmentCachingFilter
2. Configure filter in web.xml
3. Create a mapping in web.xml
4. Configure a cache entry matching the filter name

Web Caching

Example Scenario



Filter Configuration in web.xml

Full Page Cache

```
<filter>
<filter-name>SimplePageCachingFilter</filter-name>
<filter-class>net.sf.ehcache.constructs.web.filter.
    SimplePageCachingFilter
</filter-class>
</filter>

<filter-mapping>
    <filter-name>SimplePageCachingFilter</filter-name>
    <url-pattern>/index.jsp</url-pattern>
    <dispatcher>REQUEST</dispatcher>
    <dispatcher>INCLUDE</dispatcher>
    <dispatcher>FORWARD</dispatcher>
</filter-mapping>
```

Add Cache to ehcache

Configuration

Full Page Cache

```
<ehcache>
```

```
  <diskStore path="java.io.tmpdir"/>
```

```
  ...
```

```
  <cache name="SimplePageCachingFilter"
    maxElementsInMemory="10000"
    maxElementsOnDisk="1000"
    eternal="false"
    overflowToDisk="true"
    timeToIdleSeconds="300"
    timeToLiveSeconds="600"
    memoryStoreEvictionPolicy="LFU"
  />
```

```
</ehcache>
```

Filter Configuration in web.xml

Page Fragment Cache

```
<filter>
<filter-name>SimplePageFragmentCache</filter-name>
<filter-class>net.sf.ehcache.constructs.web.filter.
    SimplePageFragmentCachingFilter
</filter-class>
</filter>

<!-- Page Fragment Cache -->
<filter-mapping>
    <filter-name>SimplePageFragmentCachingFilter
</filter-name>
    <url-pattern>/include/Footer.jsp</url-pattern>
</filter-mapping>
```

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

JPA/Hibernate Caching

Overview

- Hibernate can either be used directly either or via the Hibernate Java Persistence API Provider in Hibernate 3.2
- ehcache is a CacheProvider for Hibernate 2.x and 3.x

Example

- Simple Country persistent object/Entity

JPA/Hibernate Caching

Step by Step

1. Configure Hibernate to use ehcache
2. Configure the Country object's cacheability; this can be done in a number of ways
3. Decide whether to use defaults, or to configure specific cache settings for the Country object cache in ehcache.xml



JPA/Hibernate Caching

Configure Hibernate to use ehcache

Add the following to hibernate.properties:

```
hibernate.cache.provider_class=net.sf.ehcache.hibernate.EhCacheProvider  
  
<!-- optional configuration file parameter -->  
net.sf.ehcache.configurationResourceName=/name_of_configuration_resource
```

JPA/Hibernate Caching

Native Hibernate Mapping File Configuration

```
<hibernate-mapping>
```

```
<class
```

```
    name="com.somecompany.someproject.domain.Country"
```

```
    table="ut_Countries"
```

```
    dynamic-update="false"
```

```
    dynamic-insert="false"
```

```
>
```

```
    <cache usage="read-write|nonstrict-read-write|read-only" />
```

```
</class>
```

```
...
```

```
</hibernate-mapping>
```

```
<cache
```

```
    usage="transactional|read-write|nonstrict-read-write|read-only"
```

```
    region="RegionName"
```

```
    include="all|non-lazy"
```

```
/>
```



JPA/Hibernate Caching

Entity Configuration With JPA and Hibernate Annotations

```
@Entity
...
@Cache(usage = CacheConcurrencyStrategy.NONSTRICT_READ_WRITE)
public class Country {

    private String name;

    ...

}
```

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

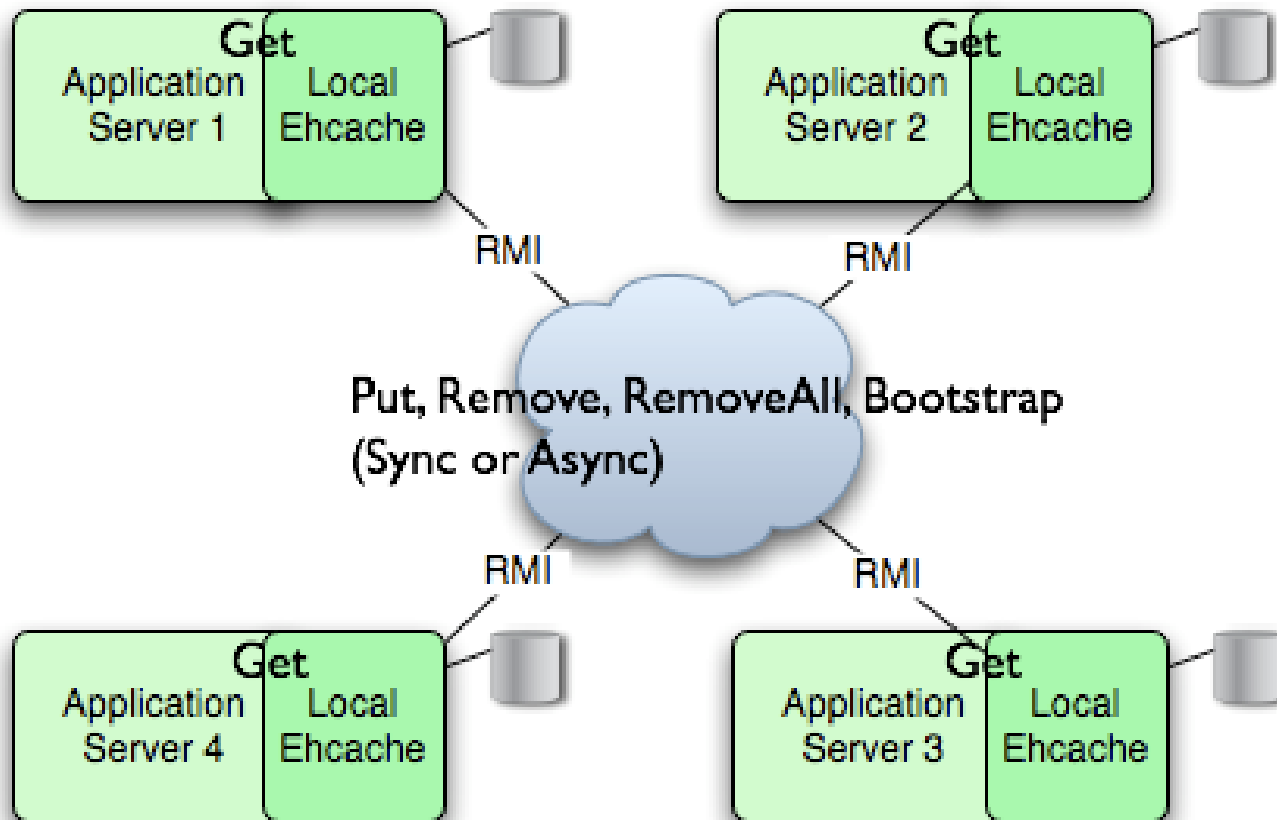
Distributed Caching for Clusters

Features

1. Pluggable distribution mechanism.
2. Comes with an RMI-based replicator. JGroups coming. Easy to add others.
3. Multiple “clusters”
4. Replication set per Cache. Options:
 1. Replicate adds
 2. Replicate removes
 3. Replicate update by copy or invalidate
 4. Async or Sync
5. Bootstrapping from existing Caches in a cluster

Distributed Caching for Clusters

Architecture



Distributed Caching for Clusters

RMI-Based Replication

1. Multicast Peer Discovery

```
<cacheManagerPeerProviderFactory class=  
"net.sf.ehcache.distribution.  
RMICacheManagerPeerProviderFactory"  
properties="peerDiscovery=automatic,  
multicastGroupAddress=230.0.0.1,  
multicastGroupPort=4446"/>
```

2. RMI Listener

```
<cacheManagerPeerListenerFactory class=  
"net.sf.ehcache.distribution.RMICacheManagerPeerLis  
tenerFactory"properties="port=40001"/>
```

Distributed Caching for Clusters

Set Replication Per Cache

```
<cache ...>
```

```
<cacheEventListenerFactory
```

```
class="net.sf.ehcache.distribution.RMICacheReplicatorFactory"
```

```
properties="replicateAsynchronously=true,
```

```
replicatePuts=true,
```

```
replicateUpdates=true,
```

```
replicateUpdatesViaCopy=true,
```

```
replicateRemovals=true,
```

```
asynchronousReplicationIntervalMillis=1000"/>
```

```
...
```

```
</cache>
```


Distributed Caching for Clusters

Set Bootstrapping Per Cache

```
<cache ...>  
  <bootstrapCacheLoaderFactory class="  
    net.sf.ehcache.distribution.RMIBootstrapCacheLoaderFactory"  
    properties="bootstrapAsynchronously=true,  
              maximumChunkSizeBytes=5000000"/>  
</cache>
```

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

JSR 107—JCACHE: Java Temporary Caching API

Description and Status

- Formed 20/3/2001
- Abstracts user from the Cache Providers in much the same way as Java DataBase Connectivity (JDBC™) does for databases
- Many think it is moribund
- Too important not to be finalized
- Forthcoming ehcache-1.3 Implementation can be used with interfaces in `net.sf.jsr107cache`
- JCache expert group has reactivated and will be conducting meetings this week on the spec

JSR 107—JCACHE

Basic Usage

1. Add ehcache-1.3 and jsr107cache jar to your classpath
2. Place configuration in ehcache.xml and add it to your classpath
3. Specify a CacheFactory in META-INF/services/net.sf.jsr107cache.CacheFactory
net.sf.ehcache.jcache.JCacheFactory
4. Create a CacheManager
`CacheManager manager = CacheManager.getInstance();`

JSR 107—JCACHE

Basic Usage (Cont.)

5. Create a Cache

```
Map env = new HashMap();
env.put("name", "test4");
env.put("maxElementsInMemory", "1000");
env.put("overflowToDisk", "true");
env.put("eternal", "true");
env.put("timeToLiveSeconds", "0");
env.put("timeToIdleSeconds", "0");
cache = manager.getCacheFactory().createCache(env);
```

6. Reference a Cache

```
Cache sampleCache = manager.getCache();

or manager.addCache(ehcache);
Cache cache = new JCache(ehcache, null);
```

7. Use it

```
sampleCache.put("key", "value");
sampleCache.get("key");
```

Agenda

Introduction

Make Your Application 10–1,000 Times Faster

General Purpose Caching

Web Caching

Java Persistence API (JPA)/Hibernate Caching

Distributed Caching for Clusters

Step-by-Step Coding Demo

Do It With Standards—JSR 107 JCACHE

Making It Real: A Wotif.com Case Study

Wotif.com Case Study



Key facts and figures

- Australia's No. 1 Hotel Booking Site
- Rapidly Growing elsewhere
- 1,000,000 users
- 10,000 concurrent users
- 3.6 million room nights sold per year
- 9,000 hotels
- Very Large Pages by design (some > 1.5MB)

Source: <http://blogs.sun.com/stories/entry/wotif>
<http://www.networksasia.net/ena/article/articleDetail.jsp?id=393040>
http://info.wotif.com/pdf/Wotif.com_Facts_Figures.pdf

Wotif.com Case Study

Screen shots—Home Page

The screenshot displays the Wotif.com website interface. At the top left is the Wotif.com logo with a tagline '28 days of great deals!'. Navigation tabs include Home, About Us, Contact Us, Investors, and Suppliers. A banner below the navigation reads 'GREAT RATES for a broad range of hotels, motels, apartments, resorts and bed & breakfasts in over 35 countries. Book last-minute accommodation or up to 28 days in advance.'

The main content area is divided into several sections:

- Search Accommodation:** A central search box with three main criteria:
 - Country:** A dropdown menu showing Australia, Austria, Belgium, and Brazil.
 - Destination:** A dropdown menu with a '(browse all)' link, listing Sydney, Sydney CBD, Melbourne, Melbourne CBD, Brisbane, Gold Coast, Sunshine Coast, Adelaide, Perth, Canberra, Darwin, Hobart, and NSW.
 - Time:** Radio buttons for 'All Days' (selected) and 'Weekends (Fri-Sun)'.
 A 'Search' button is located below these criteria.
- Advanced Search Options:** A section for filtering results based on specific properties. It includes:
 - Property type: 'All types' (dropdown)
 - Room type: 'All rooms' (dropdown)
 - Min rating: 'All' (dropdown)
 - Max. price: '\$' (input field) per night
 - Hotel name: (input field)
 - Checkboxes for: Restaurant, Kitchen/ette, Pool, Fitness Centre, Balcony, 24hr front desk, Broadband Internet, Satellite/cable TV, and In-room Spa.
 - An 'Advanced Search' button.
- Wot's On? (Left Sidebar):** A list of featured deals including 'FINA World Swimming Championships Melbourne', 'Adelaide Fringe', 'Autumn Bed & Breakfast deals', and 'INXS Accommodation'. Below this is a 'Gift voucher?' icon.
- Wot's Hot? (Right Sidebar):** A list of featured hotel deals with descriptions and prices:
 - Melbourne **Easystay The Bayside Motel** AU\$75 economy room, great location on vibrant Fitzroy St, room incl. tv, fridge, tea/coffee & ensuite!
 - Adelaide **Oaks Embassy** AU\$182 deluxe 1 bedroom, incl. use of the pool, gym, spa, sauna & steam room, spacious apartment!
 - Perth **Hyatt Regency Perth** AU\$235 Hyatt Guest Room, incl. newspaper & use of the Hotels Gym and Sauna!
 - Newcastle **Quest Newcastle** AU\$160 1 bedroom apartment, incl. a bottle of Hunter wine on arrival and parking!
 - Melbourne **Punt Hill South Yarra - Punt Road** AU\$115 one bedroom, spacious apartment with fully equipped kitchen, tv and dvd player!
 - Launceston and Tamar Valley (TAS) **Country Comfort Coach House** AU\$99 queen/twin special, incl. unlimited broadband and Auster, don't miss out!
 - Sunshine Coast **Australis Noosa Lakes** AU\$99 one bedroom 3 nights, incl. parking, enjoy the use of 3 lagoon style pools or relax on your balcony!

At the bottom of the page, there are logos for Visa, MasterCard, American Express, and Discover, along with links for Terms & Conditions, Privacy, Security, Feedback, and Help.

Source: <http://wotif.com>

Wotif.com Case Study

Screen shots—Search Results



wotif.com 28 days of great deals!

Home About Us Contact Us Investors Suppliers

Choose Accommodation

Advanced Search Options: Get more specific results by selecting to display only properties matching these criteria:

Property type: All types Room type: All rooms Min. rating: All Max. price: \$ per night Hotel name:

Restaurant Pool Balcony Broadband Internet In-room Spa
 Kitchen/ette Fitness Centre 24hr front desk Satellite/cable TV

Advanced Search

Showing properties in: Sydney Go

Currency Converter Now showing all days Change to weekends Next Seven Days 27 Mar - 9 Apr

Sydney	Full Rate	Tue 20 Mar	Wed 21 Mar	Thu 22 Mar	Fri 23 Mar	Sat 24 Mar	Sun 25 Mar	Mon 26 Mar	Tue 27 Mar	Wed 28 Mar	Thu 29 Mar	Fri 30 Mar	Sat 31 Mar	Sun 1 Apr	Mon 2 Apr
All rates are TAX inclusive and per Room listed in AU\$															
Amora Hotel Jamison ★★★★★	\$536	SOLD	\$350	\$300	\$220	\$250	\$220	\$300	\$300	\$300	\$300	\$210	\$220	\$220	\$250
BLUE Sydney - A Taj Hotel SELF RATED	\$520	\$420	\$300	\$300	\$280	\$280	\$220	\$420	SOLD	\$280	\$280	\$280	\$280	\$250	\$220
Establishment Hotel SELF RATED	\$415	SOLD	SOLD	SOLD	SOLD	SOLD	\$290	\$320	SOLD	SOLD	\$350	SOLD	SOLD	\$290	\$330
Four Seasons Hotel Sydney ★★★★★	\$440	\$450	\$450	SOLD	\$300	SOLD	\$300	\$450	\$450	\$450	\$450	\$300	\$300	\$260	\$350
Fraser Suites Sydney SELF RATED	\$420	\$266	\$266	\$266	\$294	\$294	\$252	\$266	\$266	\$266	\$266	\$294	\$294	\$252	\$266
Hilton Sydney ★★★★★	\$530	\$430	\$430	\$345	\$295	\$295	\$295	\$325	\$325	\$325	\$325	\$345	\$345	\$245	\$395
InterContinental Sydney ★★★★★	\$465	SOLD	\$485	\$440	SOLD	\$440	\$440	\$485	\$410	\$410	\$410	\$485	\$410	\$320	\$320
Kirketon Boutique Hotel SELF RATED	\$220	SOLD	SOLD	SOLD	SOLD	SOLD	\$145	\$145	SOLD	SOLD	SOLD	SOLD	SOLD	SOLD	SOLD
Marriott Sydney ★★★★★	\$520	SOLD	\$385	\$330	\$275	\$275	\$245	\$245	\$330	\$295	\$229	\$229	\$229	\$229	\$229

- + 15 more screenfuls for a total of 1.5MB
- Support many of these per second

Source: <http://www.wotif.com/search/Simple?refine=simpleSearch&country=1®ion=1&viewType=all>

Wotif.com Case Study



Technology Platform

- Sun AMD64 Servers
- RHEL5
- 64-bit Java technology
- GlassFish™ Project
- Open Message Queue
- Oracle and MySQL
- Java Platform, Enterprise Edition (Java EE platform) 5 with Enterprise JavaBeans™ (EJB™) 3, JavaServer Pages™ (JSP™) 2
- Also lots of open source

Wotif.com Case Study



ehcache

- 150 caches
- Hibernate 2 and Hibernate 3 Persistent Object and Collection Caches
- Distributed Page Caches in a “Customer” cluster
- AJAX XML Response Caches
- Distributed Java Object Caches in a “Supplier” cluster
- Uses SelfPopulating and Blocking Cache
- Web Cache Full Page and Fragment Web Filters
- Disk Store and Persistent Disk Stores

Wotif.com Case Study



How we came to implement a Cache

- Started with Apache JCS. It had some serious threading bugs 3 years ago. Submitted a patch which was ignored by the maintainer.
- Mentioned on the Hibernate mailing list about JCS problems and linked the patch for people to apply.
- Gavin King suggested forking JCS. On the basis that Hibernate would use the forked version ehcache was born. It originally stood for Easy Hibernate Cache.
- ehcache's evolution mirrors Wotif.com's own need for caching features with the progression of MemoryStore -> DiskStore -> SelfPopulatingCache -> Web Response Caching -> Persistent Disk Store -> Distributed Caching.

DEMO

1. Web App on GlassFish Project without caching
2. Add caching to a single instance
3. Use distributed caching with n instances

Summary

- Most apps will benefit from caching
- You can calculate the speed up
- ehcache is easy to use directly
- You can also benefit from ehcache indirectly through frameworks that use it
- Distributed ehcache is a small configuration step
- Abstract yourself from Caching Providers using JCACHE

For More Information

- Project Website:
<http://ehcache.sf.net>
- Downloadable Book:
<http://ehcache.sourceforge.net/EhcacheUserGuide.pdf>
- My Email:
gluck@gregluck.com



Q&A

Greg Luck
gluck@gregluck.com





EHCACHE

wotif.com

JavaOne

Distributed Caching Using the JCACHE API and ehcache, Including a Case Study on Wotif.com

Greg Luck

Maintainer, ehcache—<http://ehcache.sf.net>

Member, JSR 107 JCACHE Expert Group

Chief Architect, Wotif.com—<http://wotif.com>

TS-6175