Caching mit JSR-107

Michael Bräuer

Systemberatung, BU Server Technologies FMW
Oracle Deutschland B.V. & Co KG

Herbst/Winter 2015

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Services …

It is something that provides something for others what they want or need

- Interface (how to interact with)
- Data (what is processed)
- Implementation (how it is processed)

- Best case/ideal world: There is no state. Easy scale-out and high availability by redundancy of service instance.

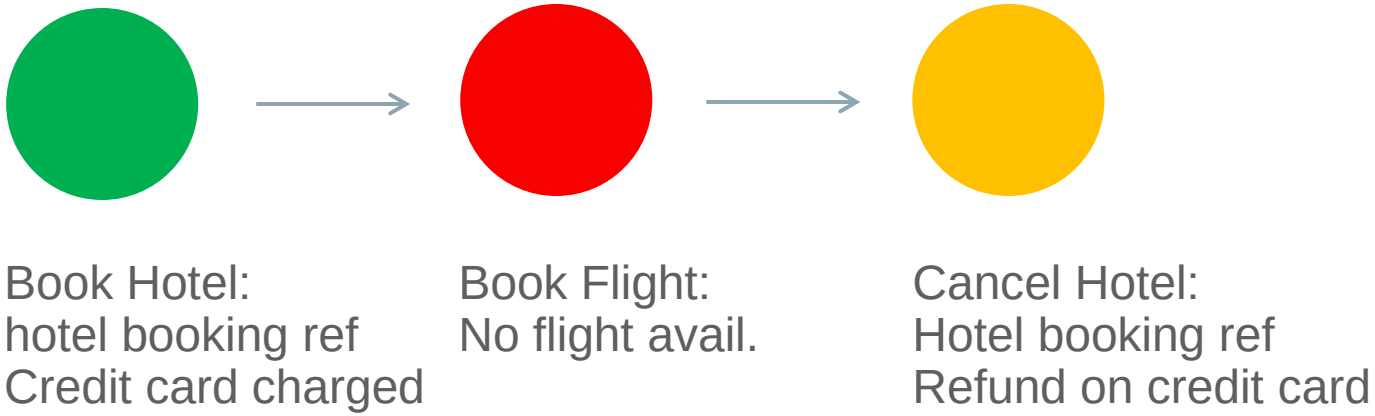
Agenda

1. State in Services
2. JSR-107 Overview
3. Need more than Caching ?
4. How to Use

Real[it|l]y?

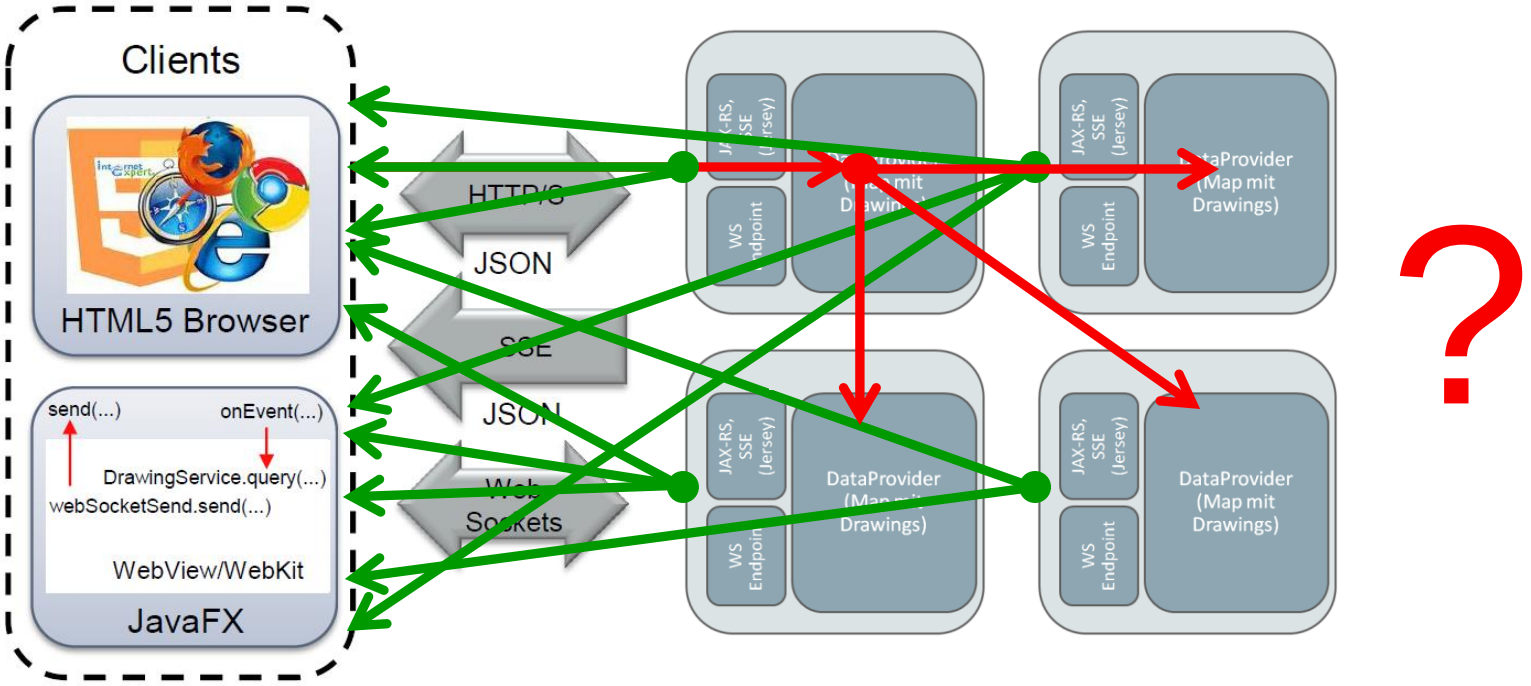
Examples of State

- Booking of Hotel and Flight
  - Business transaction: book hotel then book flight
  - Pieces of data that represents booking is needed for undo operation (no available flight → cancel hotel)



Examples of State

- Scaling-out services that use WebSocket, SSE, e.g. Drawing Board



Examples of State

- Shared data (read-only, read-mostly)
  - Don‘t access data source frequently → cache!
  - BUT: you can‘t cache everything in one JVM → scale-out, availability?

JSR-107

“The Java Caching API is an API for interacting with caching systems from Java programs”

JSR | Community | Expert Group

Summary | Proposal | Detail (Summary & Proposal)

JSRs: Java Specification Requests

JSR 107: JCACHE - Java Temporary Caching API

Stage	Access	Start	Finish
Final Release	Download page	18 Mar, 2014	
Final Approval Ballot	View results	04 Mar, 2014	17 Mar, 2014
Proposed Final Draft	Download page	24 Oct, 2013	
Public Review Ballot	View results	27 Aug, 2013	09 Sep, 2013
Public Review	Download page	05 Jul, 2013	26 Aug, 2013
Early Draft Review	Download page	23 Oct, 2012	22 Nov, 2012
Expert Group Formation		20 Mar, 2001	04 Apr, 2001
JSR Review Ballot	View results	06 Mar, 2001	19 Mar, 2001

Status: Final
JCP version in use: 2.9
Java Specification Participation Agreement version in use: 2.0

Description:
Specifies API and semantics for temporary, in memory caching of Java objects, including object creation, shared access, spooling, invalidation, and consistency across JVM's.

Expert Group Transparency:
Public Communications

Introduction to JCache

- Maven API Coordinates:

```
<dependency>
    <groupId>javax.cache</groupId>
    <artifactId>cache-api</artifactId>
     <version>1.0.0</version>
</dependency>
```

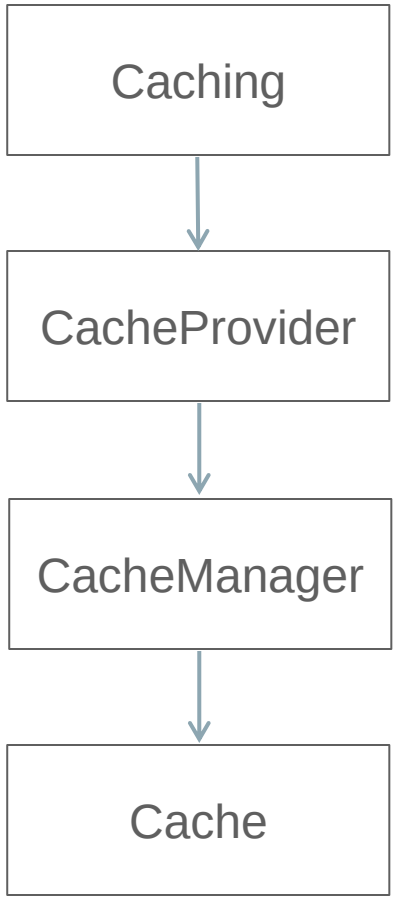
- Java Doc:
  - http://www.javadoc.io/doc/javax.cache/cache-api/1.0.0

Concepts

- Caching?
  - “Caching involves keeping a temporary copy of information in a low-latency structure for some period of time so that future requests for the same information may be performed faster.” (Spec, p.
- Cache?
  - (key_1, value_1)(key_2,value_2), ... ,(key_n,value_n)
  - put()/get() ... and more
- Is it a Map?
  - EntryProcessor, CacheEntryListener, CacheLoader, CacheWriter, Expiration, Eviction, ...

API

How do I get a Cache?



API

How do I use a Cache?

- An Example ...

API

- Read-Though/Write-Through
  - CacheLoader (Read-Though), CacheWriter (Write-Through)
- EntryProcessors
  - Lock free API
  - Sends code to data
- Events and Listeners

Oracle Coherence and JCache

- Since 12.1.3
- Support for all JCache configurations and APIs
- Additionally numerous compliant caching topologies (eg: local and partitioned/distributed)
- Provides “passthrough” configurations to support accessing existing Coherence-native Caches as JCache caches
- Use POF serialization

Oracle Coherence and JCache

- **Step 1: Application Dependencies and Class Path Requirements**
  - cache-api.jar (The JCache API)
  - coherence.jar (Oracle Coherence)
  - coherence-jcache.jar (Provides Oracle Coherence JCache Support)
- **Step 2: Configuring JCache for use with Coherence**
  - Approach 1: Use the example Coherence Cache Configuration for JCache
    `-Dtangosol.coherence.cacheconfig=coherence-jcache-cache-config.xml`
  - Approach 2: Introduce the JCache Namespace into an existing Coherence Cache Configuration

```
<cache-config …
   xmlns:jcache="class://com.tangosol.coherence.jcache.JCacheNamespace">
```

Oracle Coherence and JCache

- **Step 3: Set specific Cache Topology – if needed**
  - `-Dtangosol.coherence.jcache.configuration.classname=partitioned`
  - `-Dtangosol.coherence.jcache.configuration.classname=passthrough`
    - `-Dcoherence.cacheconfig=my-cache-config.xml`
  - `-Dtangosol.coherence.jcache.configuration.classname=remote (12.2.1)`
    - `-Dcoherence.cacheconfig=coherence-jcache-server-proxy-cache-config.xml`
- **Step 4: Acquire a Cache Manager from the Caching Provider**

```
// acquire the default CachingProvider
CachingProvider provider = Caching.getCachingProvider();

// acquire the default CacheManager
CacheManager manager = provider.getCacheManager();
```

Oracle Coherence and JCache

- **Step 5: Configure JCache Caches**

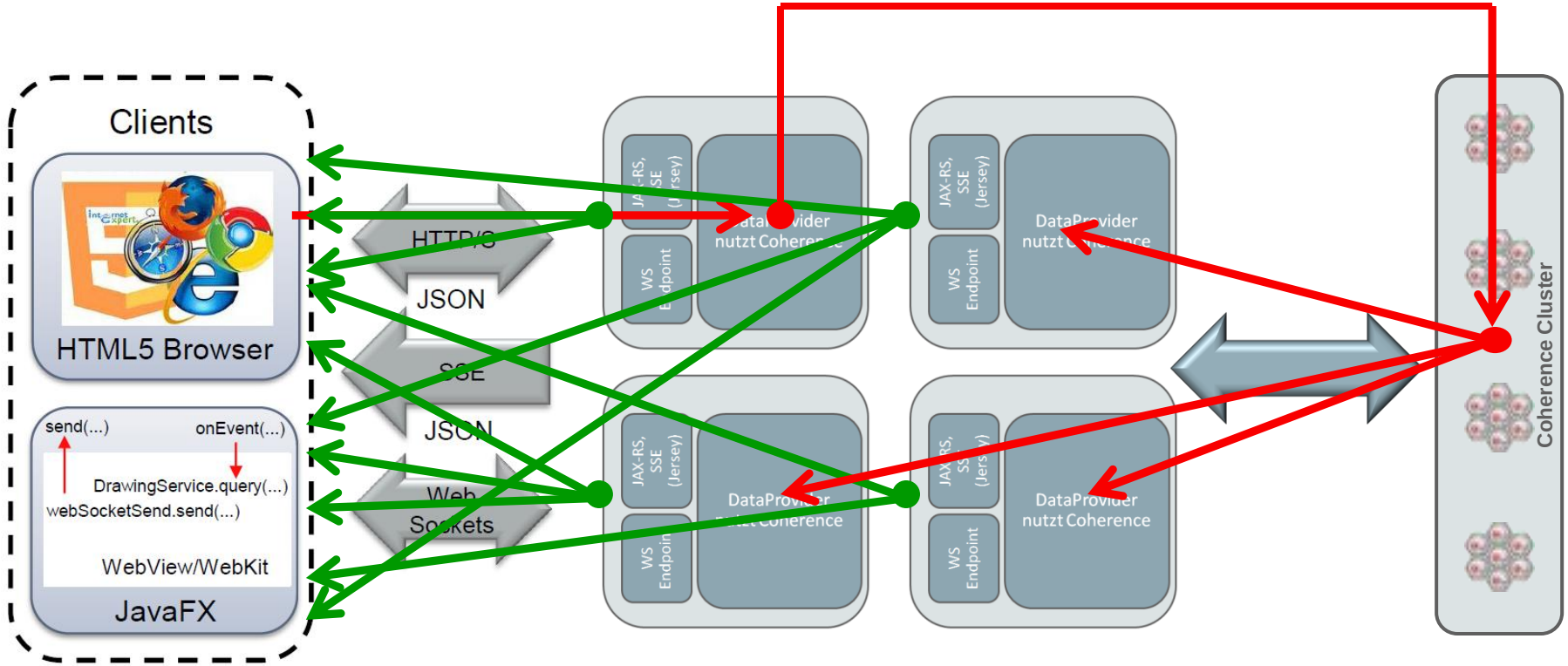
```
// define a standard JCache Cache Configuration if applicable
MutableConfiguration<String, Integer> config = new MutableConfiguration<>();

config.setStoreByValue(true)
      .setTypes(String.class, Integer.class);

// create the cache
Cache<String, Integer> cache = manager.createCache(”Ages", config);
```

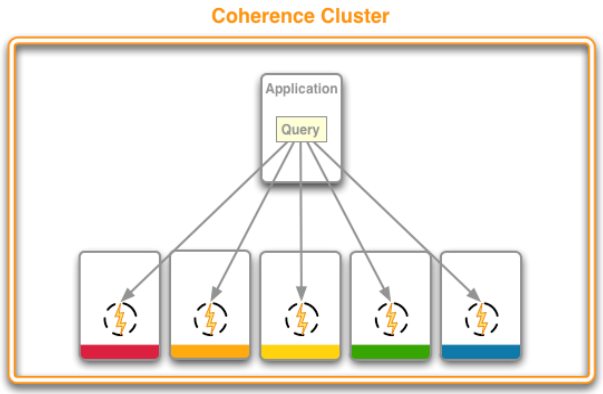
Example: Scale-Up Drawingboard

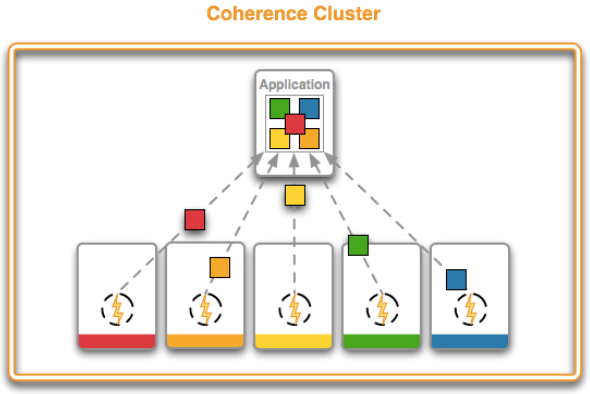
Propagate state to other JVMs → WebSpockets and SSE will inform client about state



Need more than Caching?

- In-Memory Data Grids vs. Caching
  - Not only cache but also filter and process data (in parallel)





Need more than Caching?

- Coherence provides
  - Filter, ValueExtractors, Aggregates, Indexes
  - Lambdas and Streams (Coherence 12.2.1)
  - many other features for resilience, database integration (database changes capture), high availablity
  - …

How to use it … back to Services

- Data needed by other Services: Exposing data with help of
  - REST
    - Custom based services (e.g. REST)
    - Coherence REST (built-in)
  - Remote Cache (Coherence*Extend)
  - Separate Coherence Services
- In Service: In-Process vs. Out-of-Process, e.g. Near Cache Schema
- Coherence nodes need coherence libraries + config (e.g. xml, System properties) - not more!

How to use it … back to Services

- WebLogic Multitenancy
  - Partitions/Micro-Containers integrated with Coherence!

Summary

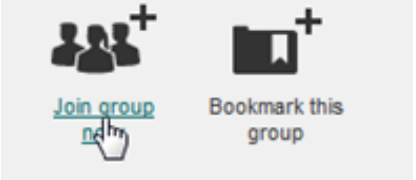
- Don‘t ignore state and shared data
- Plan for high availability and scalability (e.g. increase capacity)
  - Consider partitioned caches
- Caching is one aspect, processing data is closely related!

Java Application Server Plattform Community

Eine Community von ORACLE für Kunden, Partner und Interessierte

Code Camps
Demos
WebLogic Server
GlassFish Server
Community Treffen
Java EE
JSRs
Vorträge
Serverseitige Entwicklung
Administration
Wissensaustausch
Coherence

Registrierung: https://www.xing.com/net/oraclejavaappserver
Blog: http://fmwtech.wordpress.com
Ansprechpartner: michael.braeuer@oracle.com
peter.doschkinow@oracle.com



ORACLE

Integrated Cloud

Applications & Platform Services


ORACLE®