

# MongoDB Berlin 2013: Java Persistence Frameworks for MongoDB

Tobias.Trelle@codecentric.de / @tobiastrelle

# Tobias Trelle

---



- Senior IT Consultant  
@ codecentric AG  
(official 10gen partner)
- Conference talks on MongoDB
- MongoDB user group  
Düsseldorf/Germany
- Authoring a German book on  
MongoDB

Where have all my tables gone ...

---

**ORM** is dead

long live **ODM**

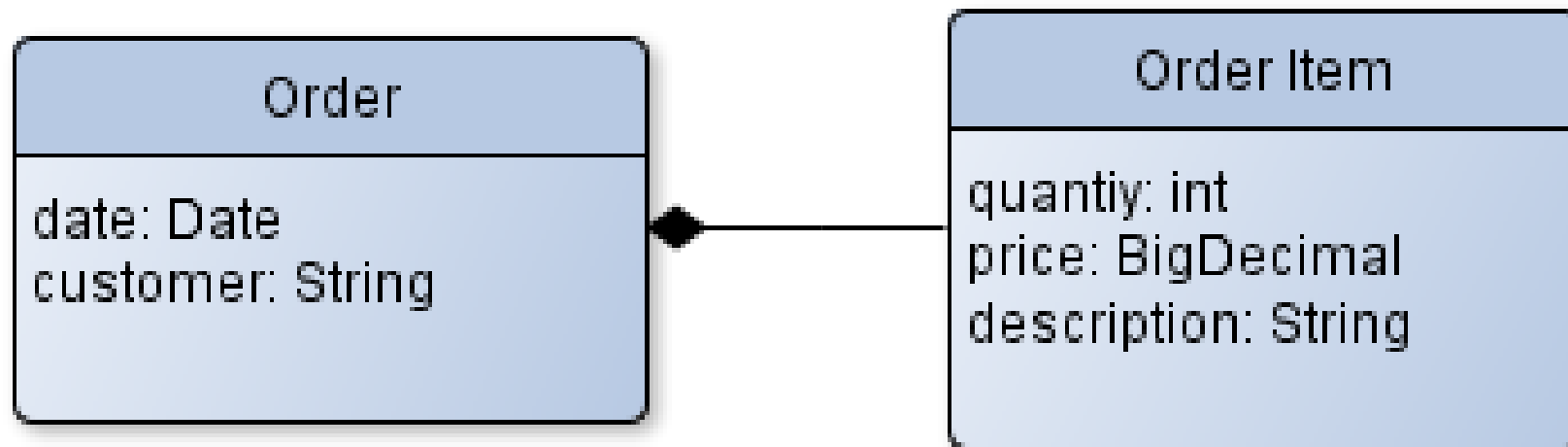
# Agenda

---

- MongoDB Java Driver
- Spring Data MongoDB
- Morphia
- Hibernate OGM

# Use Case

---



---

# Mongo Java Driver

# MongoDB Drivers

---

- **One** wire protocol for **all** client languages
- A driver implementation per language
- Responsibilities:
  - Converting language dependent data structures  $\leftarrow \rightarrow$  BSON
  - Generating ObjectId for `_id` field
- Overview: <http://www.mongodb.org/display/DOCS/Drivers>

# MongoDB Java Driver

---

- One JAR w/o further dependencies:

```
<dependency>  
  <groupId>org.mongodb</groupId>  
  <artifactId>mongo-java-driver</artifactId>  
  <version>2.10.0</version>  
</dependency>
```

- github:  
<https://github.com/mongodb/mongo-java-driver>



# Java Driver: Connect to MongoDB

```
import com.mongodb.MongoClient;

// Default: localhost:27017
mongo = new MongoClient();

// Sharding: mongos server
mongo = new MongoClient("mongos01", 4711);

// Replica set
mongo = new MongoClient(Arrays.asList(
    new ServerAddress("replicant01", 10001),
    new ServerAddress("replicant02", 10002),
    new ServerAddress("replicant03", 10003)
));
```

## Java Driver: Database / Collection

```
import com.mongodb.DB;  
import com.mongodb.DBCollection;
```

```
DB db = mongo.getDB("test");
```

```
DBCollection collection =  
    db.getCollection("foo");
```

## Java Driver: Documents

```
import com.mongodb.BasicDBObject;  
import com.mongodb.DBObject;  
  
// insert document  
DBObject doc = new BasicDBObject();  
doc.put("date", new Date());  
doc.put("i", 42);  
  
collection.insert(doc);
```

## Java Driver: Queries

```
import com.mongodb.DBCursor;

DBCursor cursor;

cursor = collection.find(); // all documents

// documents w/ {i: 42}
cursor = collection.find(
    new BasicDBObject("i", 42) );

document = cursor.next();
...
```

# Java Driver: Order Use Case

```
DB db = mongo.getDB("test");
DBCollection collection = db.getCollection("order");
DBObject order;
List<DBObject> items = new ArrayList<DBObject>();
DBObject item;

// order
order = new BasicDBObject();
order.put("date", new Date());
order.put("custInfo", "Tobias Trelle");
order.put("items", items);
// items
item = new BasicDBObject();
item.put("quantity", 1);
item.put("price", 47.11);
item.put("desc", "Item #1");
items.add(item);
item = new BasicDBObject();
item.put("quantity", 2);
item.put("price", 42.0);
item.put("desc", "Item #2");
items.add(item);

collection.insert(order);
```

---

# Spring Data MongoDB

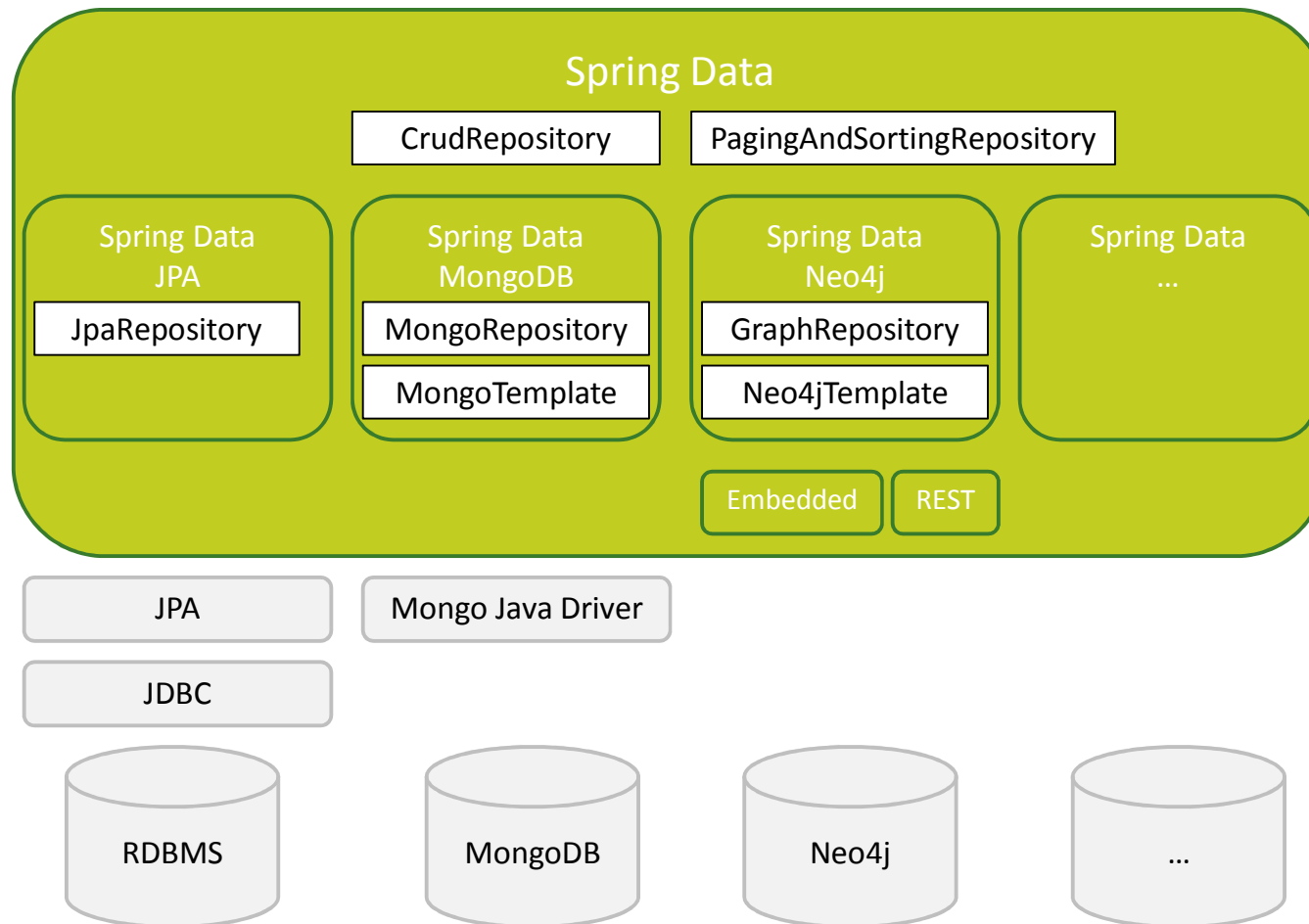
# Spring Data MongoDB – Fact Sheet

---

Vendor	VMware / SpringSource
License	Apache License, Version 2.0
Documentation	<a href="http://www.springsource.org/spring-data/mongodb">http://www.springsource.org/spring-data/mongodb</a>
Main Features	<ul style="list-style-type: none"><li>• Repository Support</li><li>• Object/Document Mapping</li><li>• Templating</li></ul>

# Spring Data

## Common patterns for RDBMS and NoSQL data stores



Quelle: <http://www.infoq.com/articles/spring-data-intro>



# Spring Data MongoDB

---

## Templating

- Resource abstraction
- Configure connections to mongod / mongos node(s)
- Collection lifecycle ( create, drop)
- Map/Reduce / Aggregation

## Object Mapping

- Annotation based: @Document, @Field, @Index etc.
- Classes are mapped to collections, Java Objects to documents

## Repository Support

- Queries are derived from methods signatures
- Geospatial Queries

# Spring Data MongoDB Template

---

## Configuration

```
<!-- Connection to MongoDB server -->
<mongo:db-factory host="localhost" port="27017" dbname="test" />

<!-- MongoDB Template -->
<bean id="mongoTemplate"
      class="org.springframework.data.mongodb.core.MongoTemplate">
    <constructor-arg name="mongoDbFactory" ref="mongoDbFactory"/>
</bean>
```

## Usage

```
@Autowired MongoTemplate template;

template.indexOps(Location.class).ensureIndex(
    new GeospatialIndex("position") );
```

# Spring Data MongoDB: Object Mapping

---

```
public class Order {  
    @Id private String id;  
    private Date date;  
    @Field("custInfo") private String customerInfo;  
    List<Item> items;    ...  
}
```

```
public class Item {  
    private int quantity;  
    private double price;  
    @Field("desc") private String description;  
    ...  
}
```

# Spring Data MongoDB: Repository Support

---

```
public interface OrderRepository extends  
    MongoRepository<Order, String> {  
  
    List<Order> findByItemsQuantity(int quantity);  
  
    List<Order> findByItemsPriceGreaterThan(double price);  
  
}
```

# Spring Data MongoDB: Repository Support

---

- Main Concept:
  - use the signature of a method to derive the query (at runtime)
- Base Implementations / abstractions for
  - CRUD operations
  - Paging
  - Sorting

# Spring Data MongoDB: Additional Goodies

---

- Map/Reduce / Aggregation framework
- Index Management
- Support for GridFS
- Geospatial indexes / queries
- Optimistic Locking

---

# Hibernate OGM

# Hibernate OGM MongoDB – Fact Sheet

---

Vendor	JBoss / Redhat
License	GNU LGPL, Version 2.1
Documentation	<a href="http://www.hibernate.org/subprojects/ogm.html">http://www.hibernate.org/subprojects/ogm.html</a>
Main Features	<ul style="list-style-type: none"><li>• JPA API (Subset)</li><li>• JPQL Query Language</li></ul>

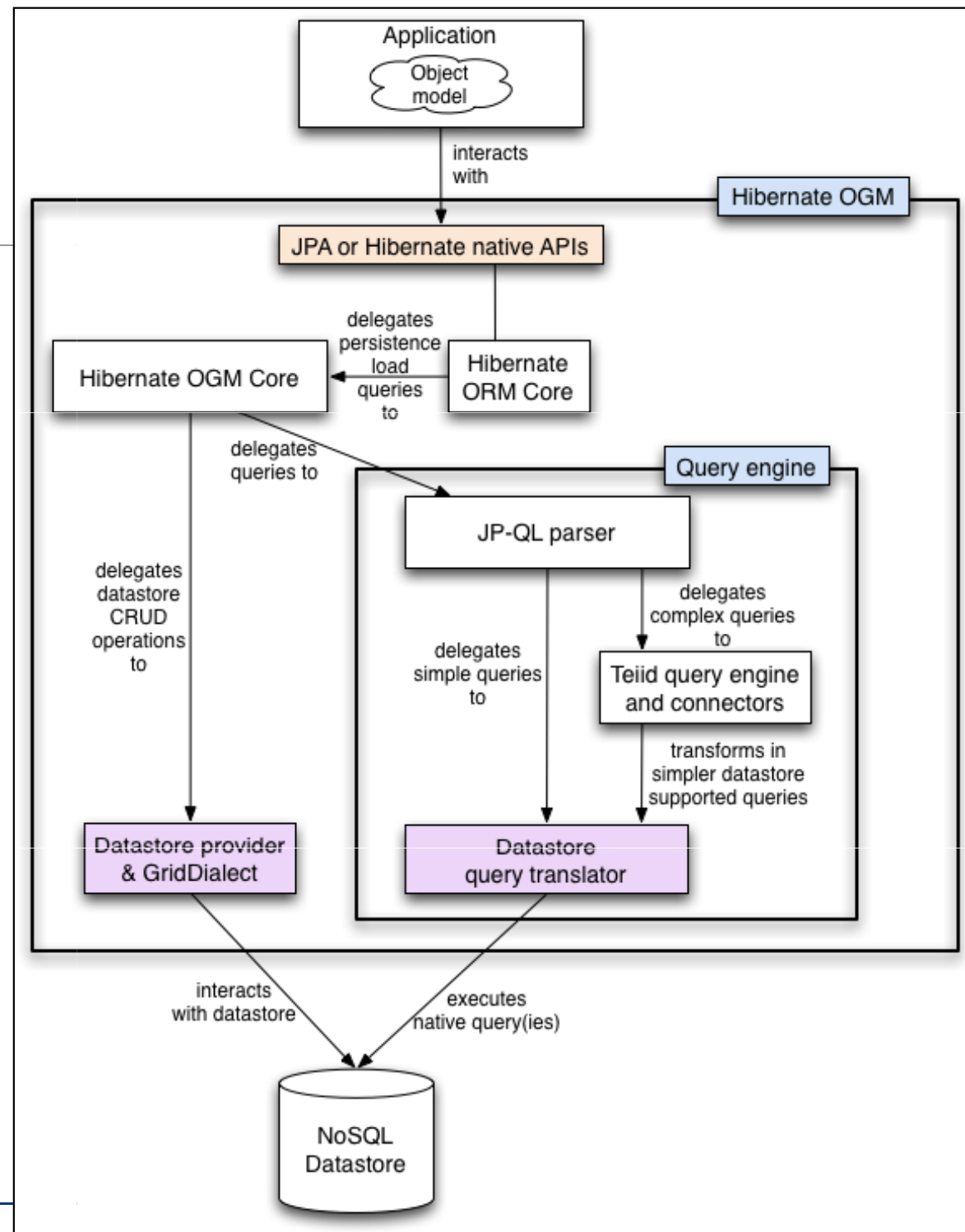


# Hibernate OGM

---

- Implements JPA API (subset)
- JP-QL query are translated to native datastore queries
- Supports Infinispan, EhCache, **MongoDB**

# Hibernate OGM Architecture



Source:

<http://docs.jboss.org/hibernate/ogm/4.0/reference/en-US/html/ogm-architecture.html#d0e409>

# Hibernate OGM MongoDB: Configuration

---

```
<persistence version="2.0" ...>
  <persistence-unit name="primary">
    <provider>org.hibernate.ogm.jpa.HibernateOgmPersistence</provider>
    <class>hibernate.Order</class>
    <class>hibernate.Item</class>
    <properties>
      <property name="hibernate.ogm.datastore.provider"
        value="org.hibernate.ogm.datastore.mongodb.impl.MongoDBDatastoreProvider"/>
      <property name="hibernate.ogm.mongodb.database" value="odm"/>
      <property name="hibernate.ogm.mongodb.host" value="localhost"/>
      <property name="hibernate.ogm.mongodb.port" value="27017"/>
    </properties>
  </persistence-unit>
</persistence>
```

# Hibernate OGM MongoDB: Object Mapping

---

```
@Entity
@NamedQuery(
    name="byItemsQuantity",
    query = "SELECT o FROM Order o JOIN o.items i WHERE i.quantity = :quantity"
)
public class Order {
    @GeneratedValue(generator = "uuid")
    @GenericGenerator(name = "uuid", strategy = "uuid2")
    @Id private String id;

    private Date date;

    @Column(name = "custInfo") private String customerInfo;

    @ElementCollection
    private List<Item> items;
```

# Hibernate OGM MongoDB: Object Mapping

---

@Embeddable

```
public class Item {
```

```
    private int quantity;
```

```
    private double price;
```

```
    @Column(name="desc") private String description;
```

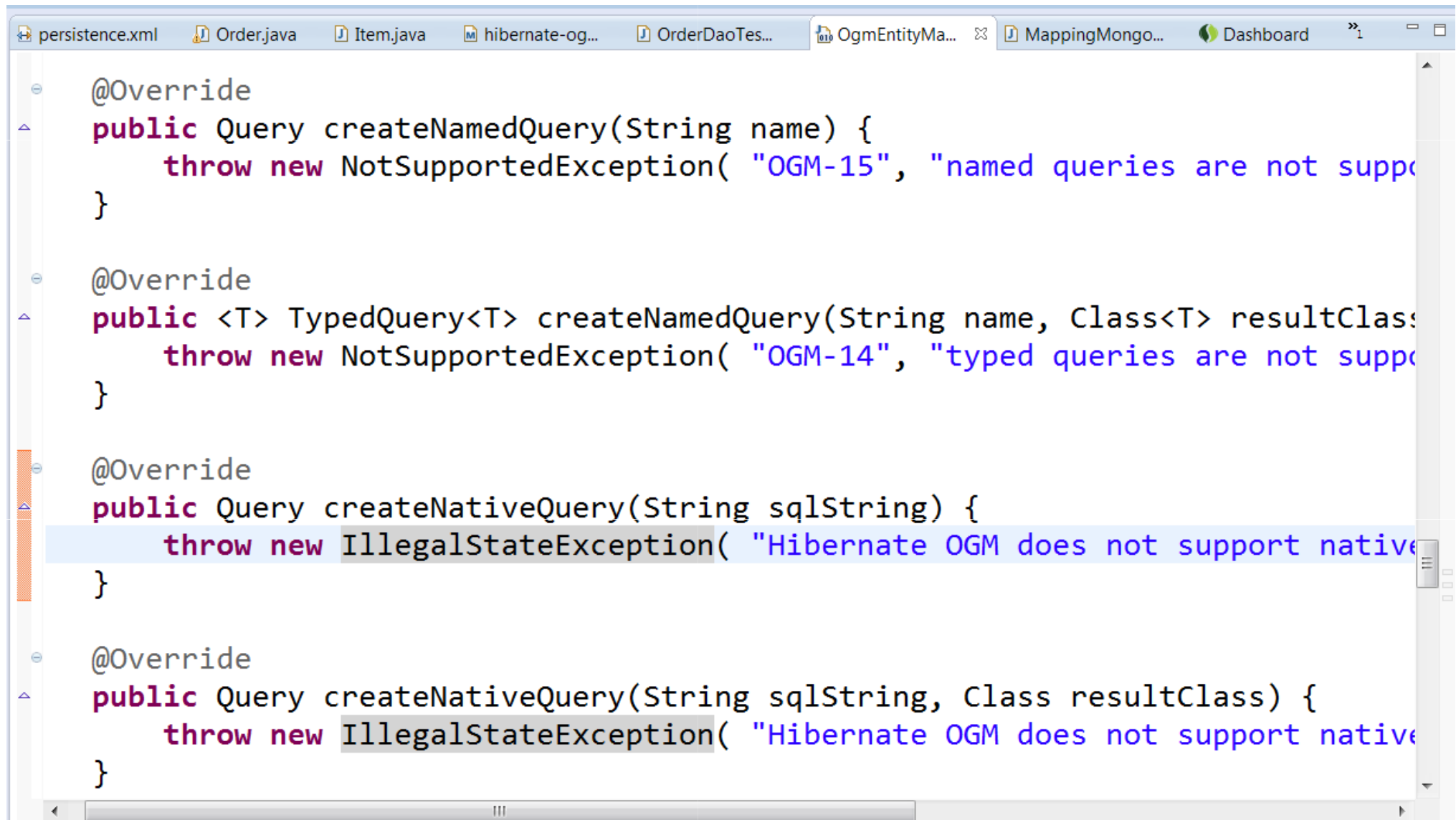
```
    ...
```

# Hibernate OGM: Summary

---

- Very early beta
- Only persist / merge / remove
- No query support (yet)
- Uses **relational** API

# Hibernate OGM: OgmEntityManager



```
persistence.xml Order.java Item.java hibernate-og... OrderDaoTes... OgmEntityMa... MappingMongo... Dashboard »1
```

```
@Override
public Query createNamedQuery(String name) {
    throw new NotSupportedException( "OGM-15", "named queries are not supported" );
}

@Override
public <T> TypedQuery<T> createNamedQuery(String name, Class<T> resultClass) {
    throw new NotSupportedException( "OGM-14", "typed queries are not supported" );
}

@Override
public Query createNativeQuery(String sqlString) {
    throw new IllegalStateException( "Hibernate OGM does not support native SQL queries" );
}

@Override
public Query createNativeQuery(String sqlString, Class resultClass) {
    throw new IllegalStateException( "Hibernate OGM does not support native SQL queries" );
}
```

---

# Morphia



# Morphia – Fact Sheet

---

Developer	Scott Hernandez, James Green
License	Apache License, Version 2.0
Documentation	<a href="https://github.com/jmkgreen/morphia/wiki/Overview">https://github.com/jmkgreen/morphia/wiki/Overview</a>
Main Features	<ul style="list-style-type: none"><li>• Object/Document Mapping</li><li>• Custom Query API</li><li>• DAO support</li></ul>

# Morphia: Object Mapping

---

```
public class Order {  
    @Id private ObjectId id;  
    private Date date;  
    @Property("custInfo") private String customerInfo;  
    @Embedded List<Item> items;  
    ...  
}  
  
public class Item {  
    private int quantity;  
    private double price;  
    @Property("desc") private String description;  
    ...  
}
```

# Morphia: Queries

---

```
public class OrderDao extends BasicDAO<Order, ObjectId> {

    List<Order> findByItemsQuantity(int quantity) {
        return
            find( createQuery().filter("items.quantity", quantity))
                .asList();
    }
    List<Order> findByItemsPriceGreaterThan(double price) {
        return
            find( createQuery().field("items.price").greaterThan(price) )
                .asList();
    }
    ...
}
```

# Morphia: Custom query syntax – why?

---

Morphia	Mongo Query
=	\$eq
!=, <>	\$neq
>, <, >=, <=	\$gt, \$lt, \$gte, \$lte
in, nin	\$in, \$nin
elem	\$elemMatch
...	....

Judge yourself ...

---

## **Spring Data MongoDB**

<https://github.com/ttrelle/spring-data-examples>

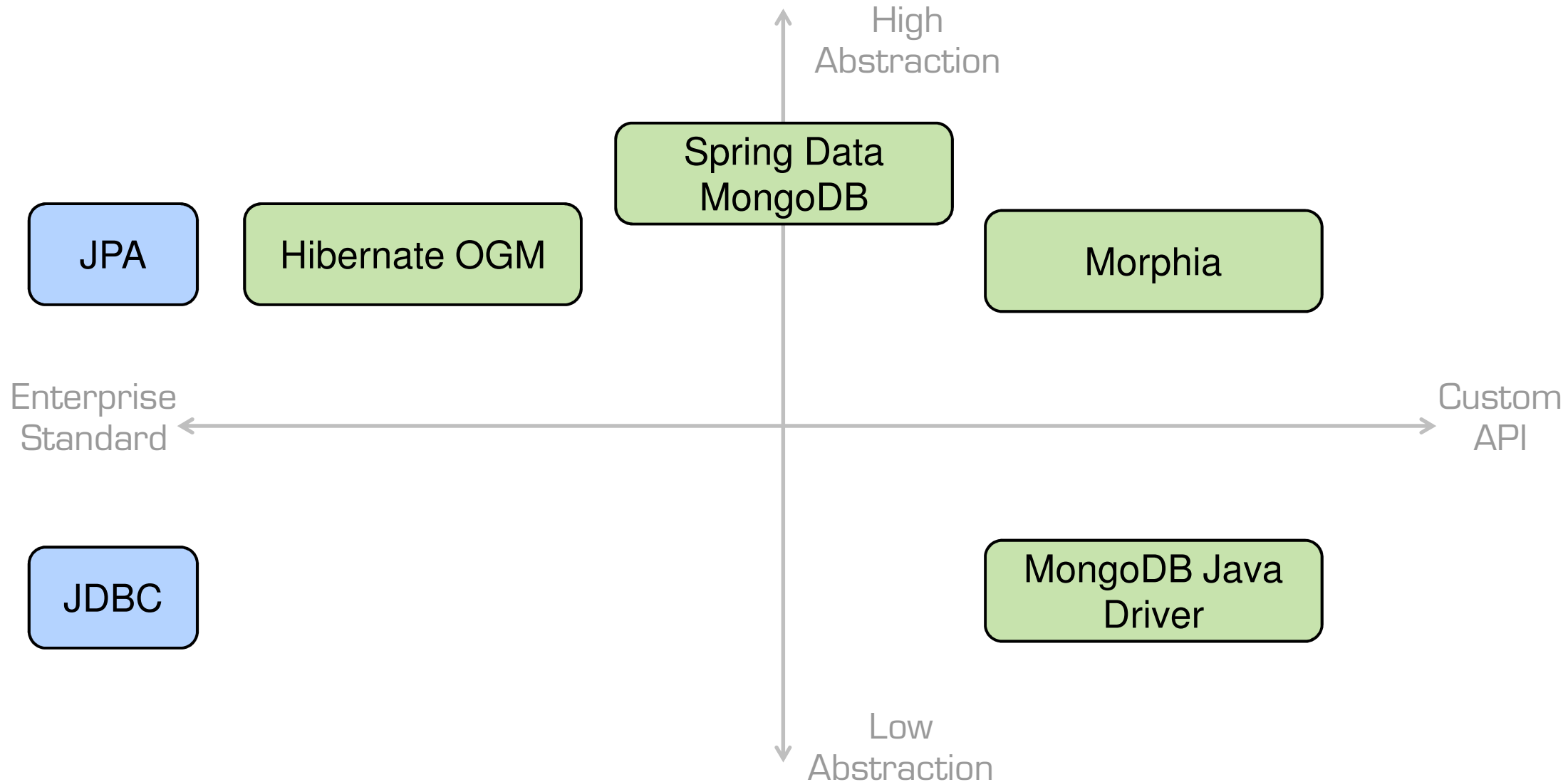
## **Hibernate OGM MongoDB**

<https://github.com/ttrelle/hibernate-ogm-examples>

## **Morphia**

<https://github.com/ttrelle/morphia-mongodb-examples>

# Which one should I use?



# German MongoDB User Groups (MUGs)

---

## **MUG Düsseldorf**

<https://www.xing.com/net/mongodb-dus>

@MongoDUS

## **MUG Berlin**

<http://www.meetup.com/MUGBerlin/>

@MUGBerlin

## **MUG Frankfurt/Main**

<https://www.xing.com/net/mongodb-ffm>

@MongoFFM

## **Hamburg MUG**

<https://www.xing.com/net/mugh>

## **MUG München**

<http://www.meetup.com/Muenchen-MongoDB-User-Group/>

@mongomuc

# QUESTIONS?

---

Tobias Trelle

codecentric AG  
Merscheider Str. 1  
42699 Solingen

tel +49 (0) 212.233628.47  
fax +49 (0) 212.233628.79  
mail [Tobias.Trelle@codecentric.de](mailto:Tobias.Trelle@codecentric.de)  
twitter @tobiastrelle

[www.codecentric.de](http://www.codecentric.de)  
[blog.codecentric.de/en/author/tobias-trelle](http://blog.codecentric.de/en/author/tobias-trelle)  
[www.xing.com/net/mongodb-dus](http://www.xing.com/net/mongodb-dus)

