

Saving Time and Money with Spring-Data-Eclipse-Store



Johannes Rabauer

- B. Sc. Informatics
(Hof University of Applied Sciences)
- Atomic Power Plant
- Radar Station
- Institute for Information Systems (iisys)

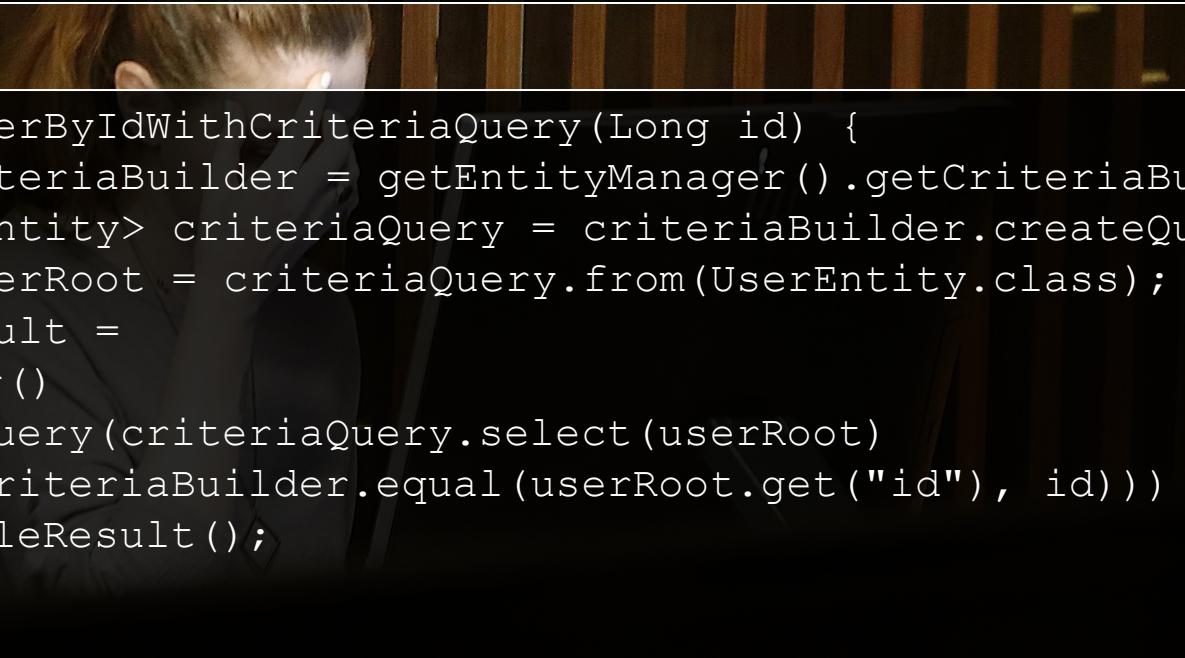
XDEV

- Open Source Contributor
- <https://github.com/JohannesRabauer>



SQL and JPA are difficult

```
Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/");
String sql = "SELECT * FROM user WHERE id = ?";
PreparedStatement ps = con.prepareStatement(sql);
ps.setString(1, "1");
ResultSet result = ps.executeQuery();
```



```
public UserEntity getUserByIdWithCriteriaQuery(Long id) {
    CriteriaBuilder criteriaBuilder = getEntityManager().getCriteriaBuilder();
    CriteriaQuery<UserEntity> criteriaQuery = criteriaBuilder.createQuery(UserEntity.class);
    Root<UserEntity> userRoot = criteriaQuery.from(UserEntity.class);
    UserEntity queryResult =
        getEntityManager()
            .createQuery(criteriaQuery.select(userRoot)
                .where(criteriaBuilder.equal(userRoot.get("id"), id)))
            .getSingleResult();
    return queryResult;
}
```

Mapping

Manual
ResultSet
.getString(,ID“)

Automatic between
Table- and Object-
Structure



EclipseStore



- Fast
- Can update parts of graph
- Different store targets
- Lazy-Loading

Example

```
final public class Root{  
    private final List<Product> products = new ArrayList<>();  
    private final Lazy<BigObject> someBigObject = Lazy.Reference(new BigObject());  
  
    public List<Product> getProducts() {...}  
}  
  
public record Product(String name) {}
```

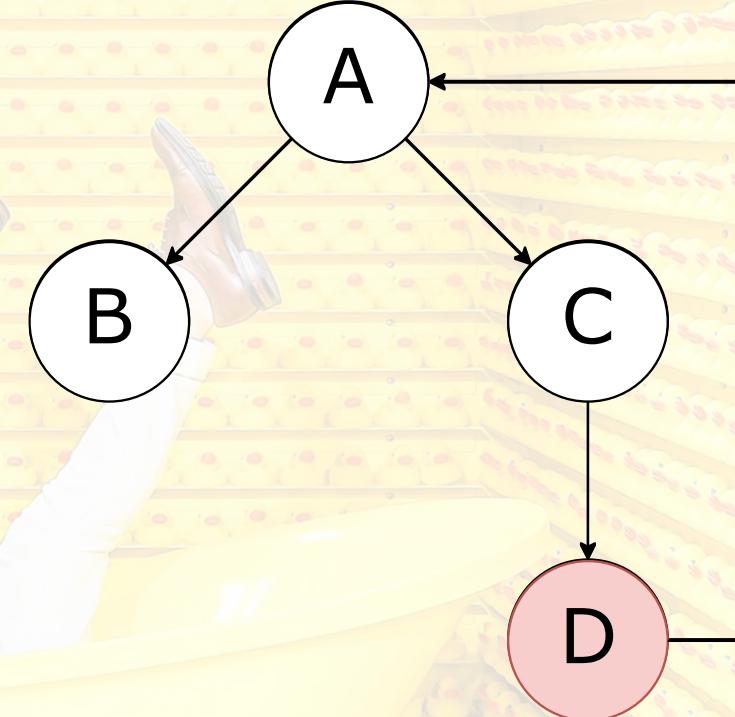
```
final Root rootObject = new Root();  
final EmbeddedStorageManager storageManager = EmbeddedStorage.start(rootObject);  
  
rootObject.getProducts().add(new Product("Chair"));  
storageManager.store(rootObject.getProducts());  
  
storageManager.shutdown();
```

Differences

	JDBC	JPA	EclipseStore
Persistence	Seperate server	Seperate server	Local, seperate server, Cloud
Querying	SQL	SQL, Criteria Query	Pure Java
Parsing/Mapping	Manual	Automatic	Automatic

Acceptance?

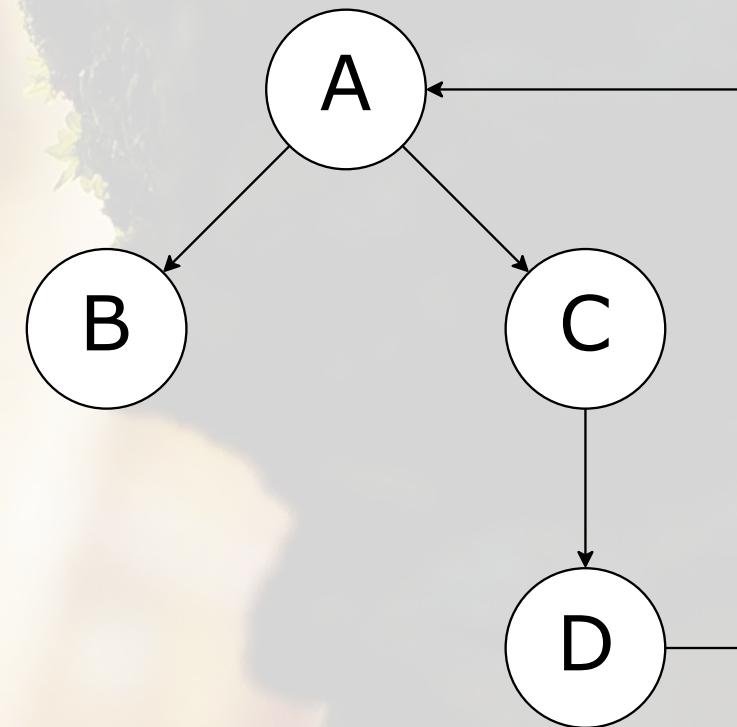
First name	Last name	Birth place
Mick	Fleetwood	Redruth
Stevie	Nicks	Phoenix



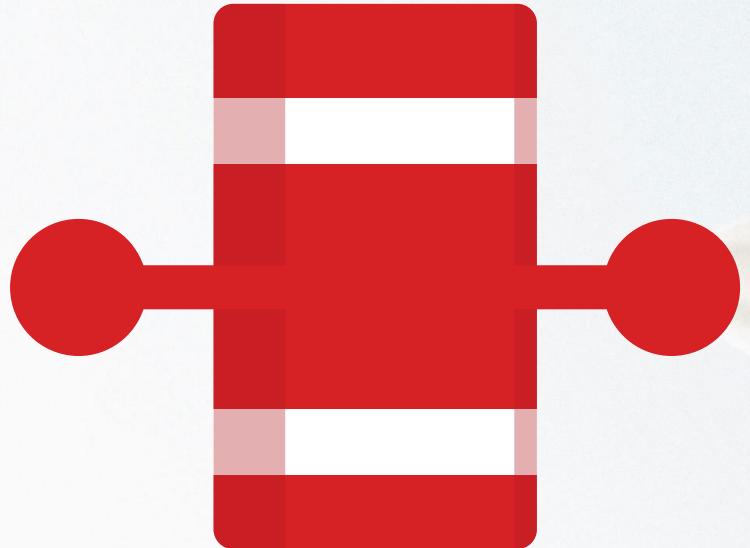
```
...  
rootObject.getProducts().add(new Product("Chair"));  
storageManager.store(rootObject.getProducts());  
...
```

Gap

First name	Last name	Birth place
Mick	Fleetwood	Redruth
Stevie	Nicks	Phoenix



X D E V



SPRING DATA ECLIPSESTORE





Spring Data

- Plenty of modules: JPA, MongoDB, Redis, ...
- Powerful repository and custom object-mapping abstractions

```
public interface SimpleUserRepository extends ListCrudRepository<User, Long> {}
```

```
simpleUserRepository.save(new User(...));
```

- Dynamic query derivation from repository method names

```
public interface SimpleUserRepository extends ListCrudRepository<User, Long> {  
    Optional<User> findByUsername(Optional<String> username);  
    List<User> findByLastname(String lastname);  
    ...}
```

- Repositories coming to Jakarta with Jakarta Data 1.0

Get started

- Download Spring Boot project with Maven and Java from <https://start.spring.io/>
- Add to pom.xml

```
<dependency>
    <groupId>software.xdev</groupId>
    <artifactId>spring-data-eclipse-store</artifactId>
    <version>1.0.2</version>
</dependency>
```

- Add to SpringBootApplication

```
@SpringBootApplication
@EnableEclipseStoreRepositories
```

Code example

```
public record Pet(String name) {}
```

```
public record Owner(String firstName, String lastName, ArrayList<Pet> pets) {}
```

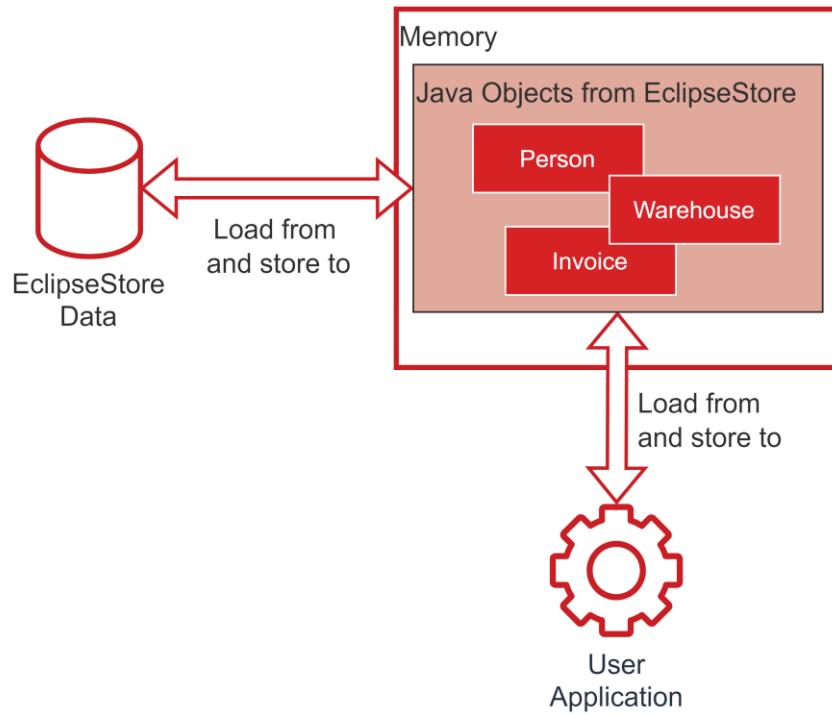
```
public interface OwnerRepository extends CrudRepository<Owner, String> {}
```

```
public class DemoApplication{
    @Override
    public void run(final String... args)
    {
        Owner stevieNicks = new Owner("Stevie", "Nicks", List.of(new Pet("Catsy")));
        this.ownerRepository.save(stevieNicks);

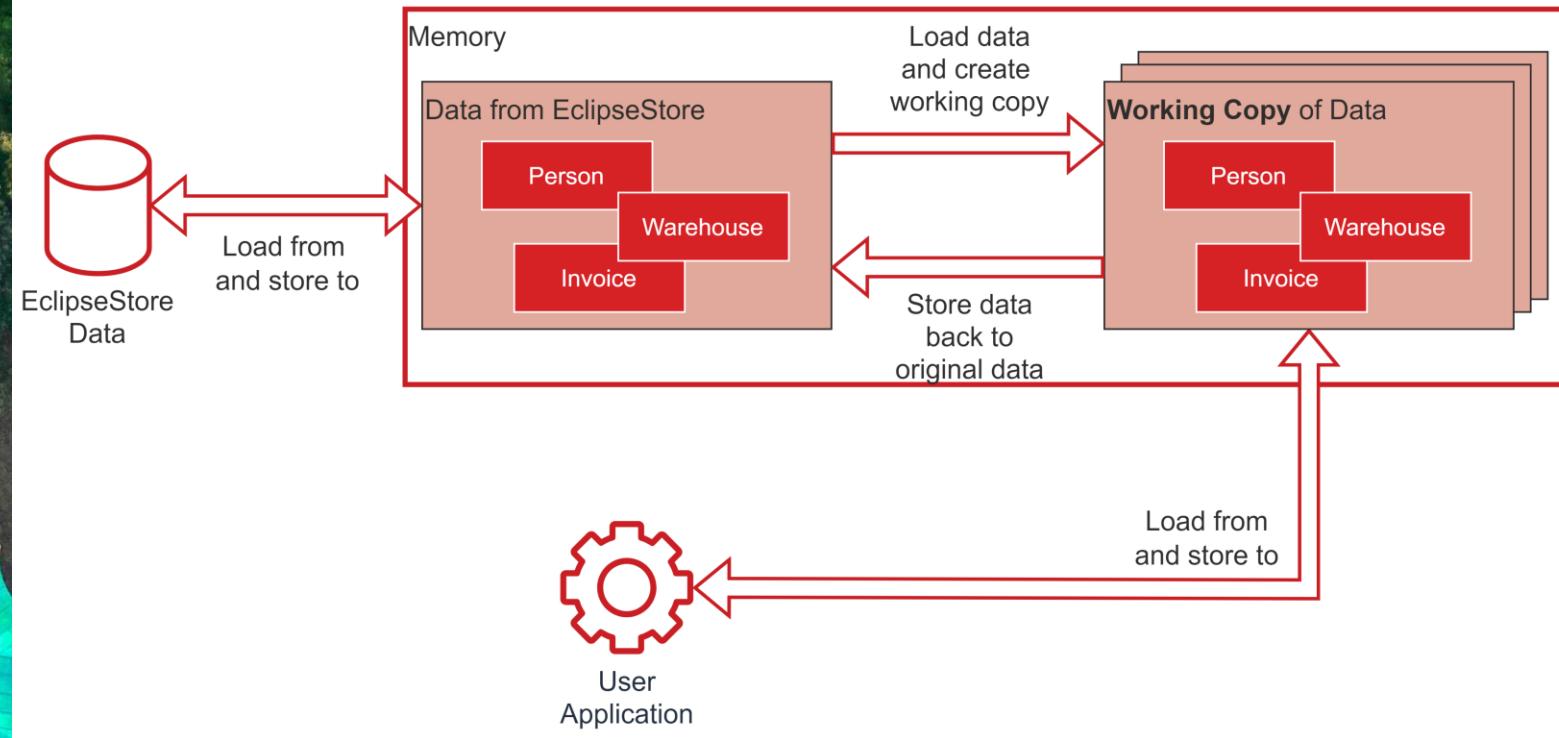
        this.ownerRepository.findAll().forEach(o -> LOG.info(o.toString()));
    ...
}
```

Core Difference

Native behavior of EclipseStore

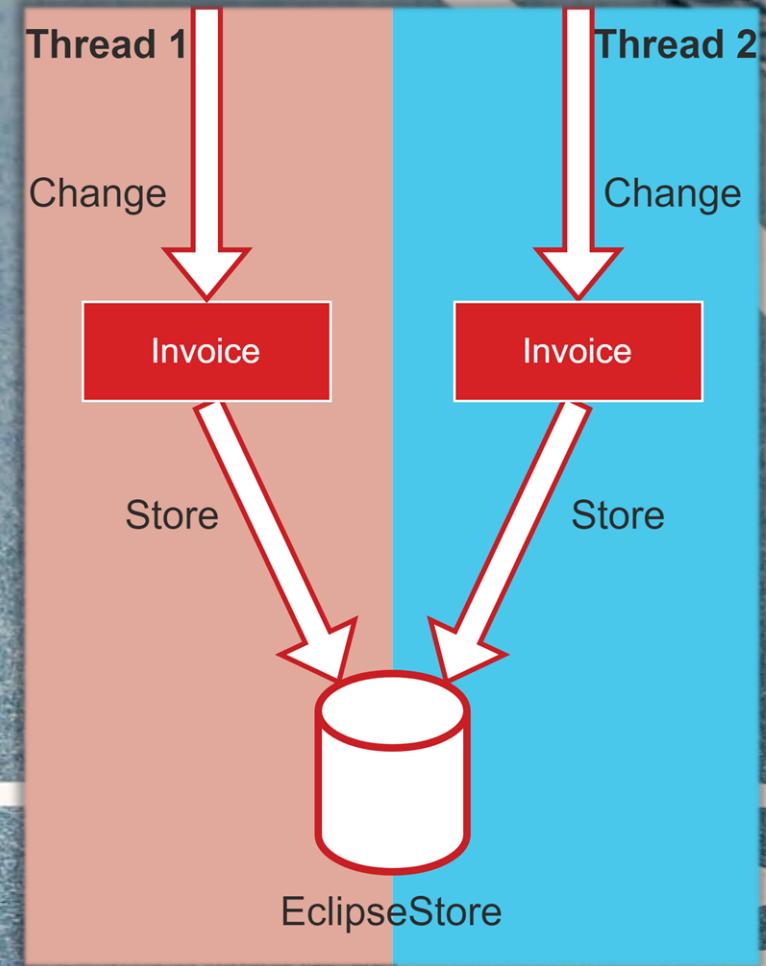
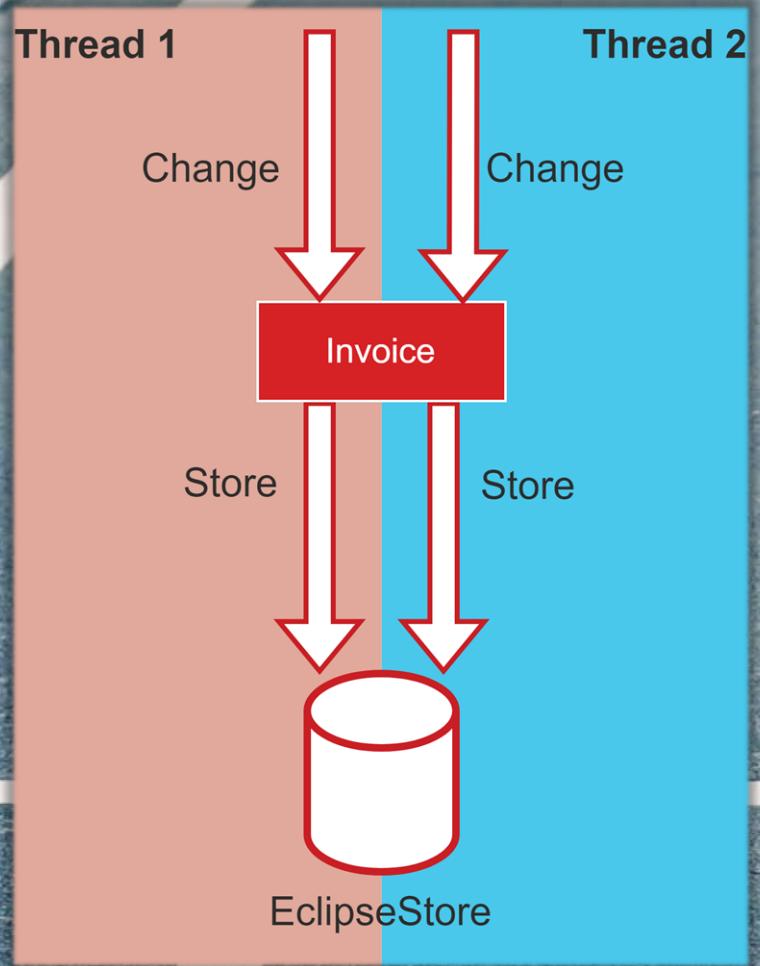


Behavior of EclipseStore with Spring-Data-Eclipse-Store



1. Principle of Least Surprise
2. Concurrency

Concurrency



Migration

- Can coexist with Spring JPA

```
public interface OwnerRepository extends CrudRepository<Owner, String>{ }
```



```
public interface OwnerRepository extends EclipseStoreCrudRepository<Owner, String>{ }
```

- Migrate Data with EclipseStoreDataImporter
- OpenRewrite Migration recipe:

<https://github.com/xdev-software/spring-data-eclipse-store-migration>

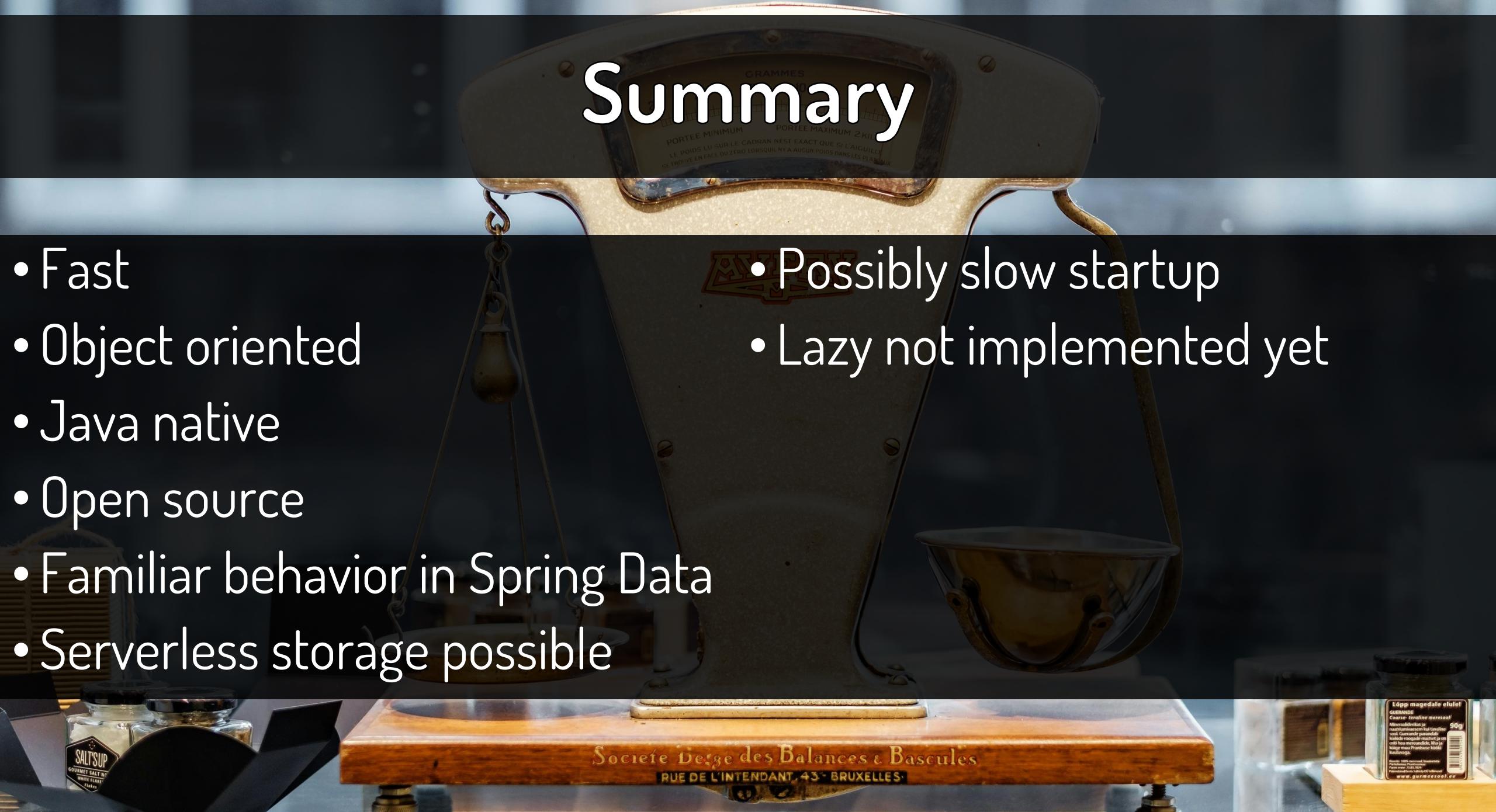
```
mvn org.openrewrite.maven:rewrite-maven-plugin:run
-Drewrite.recipeArtifactCoordinates=software.xdev:spring-data-eclipse-store-migration:1.0.0
-Drewrite.activeRecipes=software.xdev.spring.data.eclipse.store.JpaMigration
```

Downsides

- Higher memory usage than EclipseStore
- Lazy not usable (Todo)
- Queries not usable (Todo)
- Startup time can be slow with gigabytes of data

Summary

- Fast
- Object oriented
- Java native
- Open source
- Familiar behavior in Spring Data
- Serverless storage possible
- Possibly slow startup
 - Lazy not implemented yet



Sources / Further Reading

- Repository with demo and presentation:
<https://github.com/JohannesRabauer/talk-2024-spring-data-eclipse-store>
- <https://github.com/xdev-software/spring-data-eclipse-store>
- <https://eclipsestore.io/>
- <https://spring.io/projects/spring-data>
- <https://jakarta.ee/specifications/data/>
- <https://github.com/xdev-software/spring-data-eclipse-store-migration>
- Images from <https://unsplash.com/>

