Using Transactions with Spring

Kesha Williams
SOFTWARE ENGINEERING MANAGER & PROFESSOR

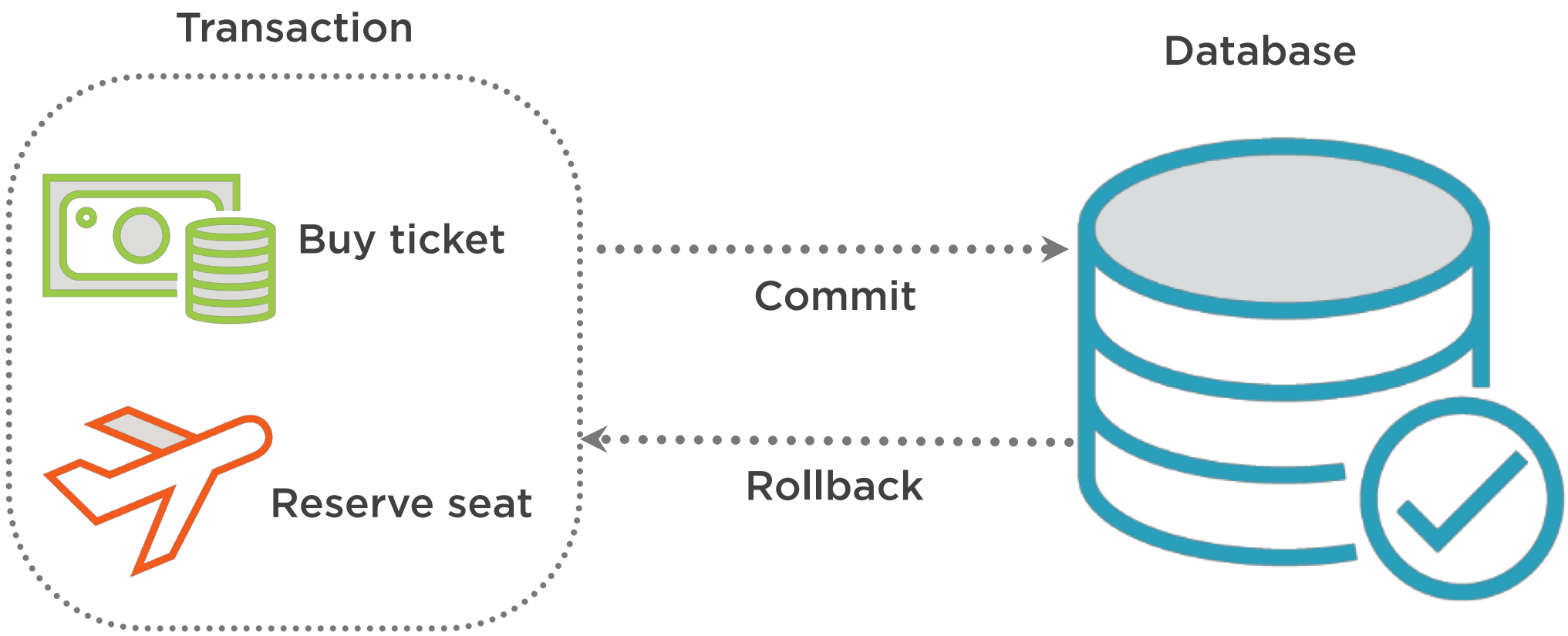
@KeshaWillz www.kesha.tech

Overview

- Advantages of Spring
- Programmatic transaction management
- Declarative transaction management
- Extra support provided by Spring Boot
- Entity manager vs. database transactions

Advantages of the Spring Framework

Spring Support



Manage Transactions

Java Transaction API	Java Database Connectivity	Hibernate
Java Persistence API	Java Data Objects	Java Message Service

Consistent Programming Model

Phase 1

Java Data Objects
JDO

Phase 2

Hibernate
ORM

Spring Framework

Uniform API
Simple

Manual Transaction Management

```
public void saveTicket(Ticket ticket) {

    Session session = sessionFactory.getCurrentSession();
    session.getTransaction().begin();
    session.save(ticket);
    session.getTransaction().commit();

}
```

Spring Transaction Management

```
@Transactional
public void saveTicket(Ticket ticket){
    sessionFactory.getCurrentSession().save(ticket);
}
```

```
@Transactional

public void scheduleRelease(Ticket ticket, Release release)
{

    ticketDao.saveTicket(ticket);
    assignToRelease(ticket);
    doOtherStuff(ticket);

}
```

Complex Transaction

Automatic rollback during an exception

Spring provides support for programmatic and declarative transaction management.

Programmatic

```
public void saveTicket(Ticket ticket) {

    Session session = sessionFactory.getCurrentSession();
    session.getTransaction().begin();
    session.save(ticket);
    session.getTransaction().commit();

}
```

```
@Transactional

public void
scheduleRelease(Ticket ticket,
Release release) {

   ticketDao.saveTicket(ticket);
   assignToRelease(ticket);
   doOtherStuff(ticket);

}
```

- Begin
- Suspend
- Commit
- Rollback
- Transactional parameters

Advantages of Declarative Transactions

- Container managed
- Time saving
- Cleaner separation
- Easier to maintain

Programmatic Transaction Management

Programmatic Transaction Management

Transaction template

Similar to Spring templates like JdbcTemplate and other available templates

Platform transaction manager

Handles transactions across Hibernate, JDBC, JPA, JMS, etc.

Spring Transaction Managers

Platform

Transaction manager

Jta

Transaction manager

Hibernate

Transaction manager

DataSource

Transaction manager

Jpa

Transaction manager

HibernateTransactionManager

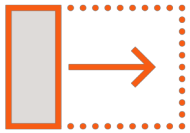
A uniform API that uses the features of Hibernate's transactions, while maintaining the advantage of Spring's unified transaction abstractions.

Declarative Transaction Management

@Transactional

This annotation is an example of declarative transaction management and allows us to easily work with database transactions.

@Transactional Annotation



Proxy created to hold transaction management code

Annotation used at the class, interface, or method level

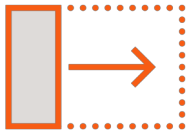
Transaction propagation handled automatically

Spring Boot and Transaction Management

@EnableTransactionManagement

Annotation that enables Spring's annotation-driven transaction management capability.

Automatic Configuration



@Transactional works without configuration

Declarative transaction management works

No platform transaction manager configured

```
<dependency>
 <groupId>org.springframework.boot</groupId>
 <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

Complex Transaction

Automatic rollback during an exception

Spring Configuration

Jpa Transaction Manager

Registers a transaction manager for us.

Additional Configuration

Datasource, entity manager, repositories

Spring Data Repositories

Calls are surrounded by a transaction

@Transactional annotations within your code are only evaluated when you use the @EnableTransactionManagement annotation.

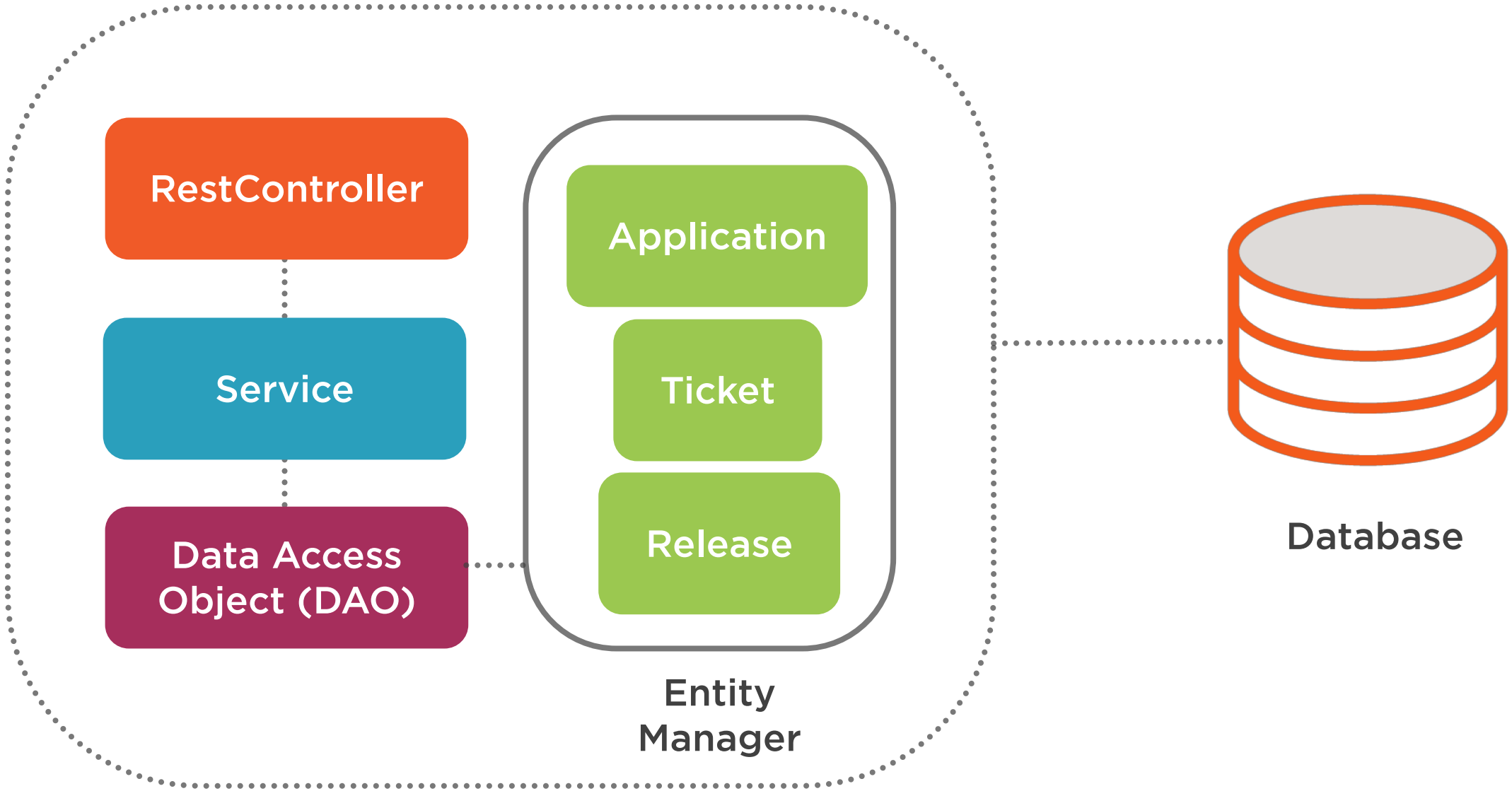
You can easily trace transaction behavior by adding logging.level.org.springframework.transaction.interceptor=TRACE

You can get access to the transaction manager with:

```
@Autowired
private PlatformTransactionManager
transactionManager
```

Entity Manager vs. Database Transactions

Entity Manager



Lifecycle and Scope

Database Transaction

@Transactional defines a single transaction; in the scope of a persistence context

Persistence Context

Defined in JPA; handles a set of entities that contain data to be persisted

Demo

Transaction pitfalls

TrackZilla

- Enhance to handle transactions
- Add resiliency

Summary

- Spring advantages
- Programmatic transaction management
- Declarative transaction management
- Spring Boot support
- Entity manager
- Database transactions