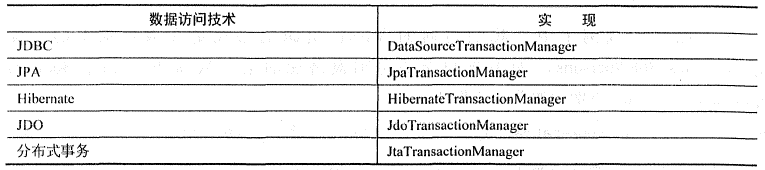
# Spring Boot核心-声明式事务

## Spring 的事务机制

而Spring 的事务机制是用统一的机制来处理不同数据访问技术的事务处理。Spring 的事务机制提供了一个PlatformTransactionManager 接口，不同的数据访问技术的事务使用不间的接口实现。



在程序中定义事务管理器的代码如下

|  |
| --- |
| @Bean **public** PlatformTransactionManager transactionManager() {  JpaTransactionManager transactionManager = **new** JpaTransactionManager();  transactionManager.setDataSource(dataSource());    **return** transactionManager; } |

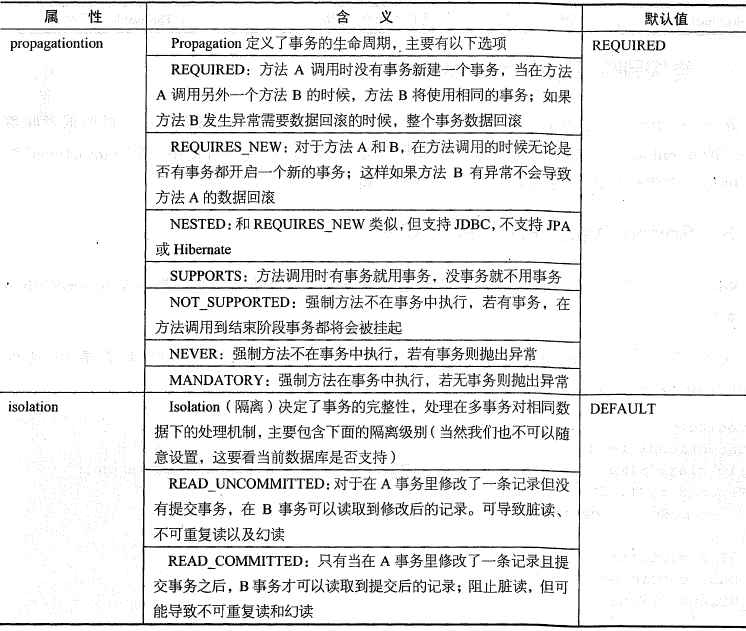
## 声明式事务

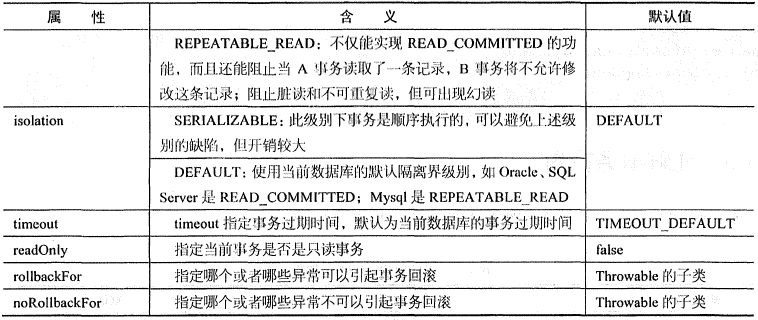
Spring 支持声名式事务，即使用注解来选择需要使用事务的方法，它使用@Transactional注解在方法上表明该方法需要事务支持。这是一个基于AOP 的实现操作，被注解的方法在被调用时， Spring开启一个新的事务，当方法无异常运行结束后， Spring 会提交这个事务。

在此处需要特别注意的是，此@Transactional 注解来自org.springframework. transaction.annotation 包，而不是javax.transaction 。Spring 提供了一个@EnableTransactionManagement 注解在配置类上来开启声明式事务的支持。使用了@EnableTransactionManagement 后， Spring 容器会自动扫描注解@Transactional。

## 注解事务行为

@Transactional通过属性来定制事务行为，如下所示。





## 类级到使用＠Transactional

@Transaction al 不仅可以注解在方法上，也可以注解在类上。注解在类上的时候意味着此类的所有public 方法都是开启事务的。如果类级别和方法级别同时使用了@Transactional 注解，则使用在类级别的注解会重载方法级别的注解。

## Spring Data JPA 的事务支持

Spring Data JPA 对所有的默认方法都开启了事务支持，且查询类事务默认启用readOnly =true 属性。

## Spring Boot 的事务支持

### 自动配嚣的事务管理器

在使用用JDBC 作为数据访问技术的时做， Spring Boot 为我们应义了PlatformTransactionManager 的实现DataSourceTransactionManager 的Bean; 配置见org.springframework.boot.autoconfigure.jdbc.DataSourceTransactionManagerAutoConfiguration 类中的定义

|  |
| --- |
| @Bean @ConditionalOnMissingBean(PlatformTransactionManager.**class**) **public** DataSourceTransactionManager transactionManager(  DataSourceProperties properties) {  DataSourceTransactionManager transactionManager = **new** DataSourceTransactionManager(  **this**.**dataSource**);  **if** (**this**.**transactionManagerCustomizers** != **null**) {  **this**.**transactionManagerCustomizers**.customize(transactionManager);  }  **return** transactionManager; } |

在使用JPA 作为数据访问技术的时候， Spring Boot 为我们了定义一个PlatformTransactionManager 的实现JpaTransaction Manager 的Bean ：配置见org.springframework.boot.autoconfigure.orm.jpa.JpaBaseConfiguration类中的定义：

|  |
| --- |
| @Bean @ConditionalOnMissingBean(PlatformTransactionManager.**class**) **public** PlatformTransactionManager transactionManager() {  JpaTransactionManager transactionManager = **new** JpaTransactionManager();  **if** (**this**.**transactionManagerCustomizers** != **null**) {  **this**.**transactionManagerCustomizers**.customize(transactionManager);  }  **return** transactionManager; } |

### 自动开启注解事务的支持

Spring Boot 专门用于配置事务的类为： org.springframework.boot.autoconfigure. transaction.TransactionAutoConfiguration ，此配置类依赖于JpaBaseConfig和DataSourceTransactionManagerAutoConfiguration 。而在DataSourceTransactionManagerAutoConfiguration 配置里还开启了对声名式事务的支持。

|  |
| --- |
| @Configuration @ConditionalOnSingleCandidate(DataSource.**class**) **static class** DataSourceTransactionManagerConfiguration {} |