Echo 的发布评论是怎么做的

原创 飞天小牛肉 [飞天小牛肉](javascript:void(0);)

**飞天小牛肉**

微信号 CS-Wiki

功能介绍 InfoQ & 阿里云签约作者，分享原创技术干货和成长经验

2021-05-06[原文](https://mp.weixin.qq.com/s?__biz=MzI0NDc3ODE5OQ==&mid=2247486744&idx=2&sn=8033de0f2e939e1d843ec7929567da1d&chksm=e959d8a0de2e51b67f1272d0ead3098c5522ad97f97318e61916806fe6902a62390cebf5de7b&scene=178&cur_album_id=1744497649518493697#rd) 发表于

收录于合集 #Echo 学习教程 18个

先梳理下发布评论这个逻辑需要做哪些事情：

* 步骤 1：增加评论数据（向评论表 comment 中添加记录行）
* 步骤 2：修改帖子的评论数量（帖子表 discuss\_post 中内置了 comment\_count 字段）

这样，处理发布评论这个逻辑的 Service 层方法就需要执行两次 DML 操作，所谓 DML 就是数据操纵语言， 属于 SQL 语言四大分类（数据查询语言 DQL、数据操纵语言 DML、数据定义语言 DDL、数据控制语言 DCL）中的其中一个，简单来说，对数据库进行添加 insert、修改 update 和删除 delete 操作的就是 DML 操作。

那么，如果步骤 1 执行成功了，而步骤 2 执行失败了，就相当于评论添加成功了但是帖子的评论数量没有修改；如果步骤 1 执行失败而步骤 2 执行成功了，就相当于帖子的评论数量增加了但是评论却没有被添加进来。

因此，我们需要保证这两个步骤要么全部成功，要么全部失败。

也就是说我们需要保证**原子性**。为此，在发布评论这块，我们需要引入**事务管理**。

**前置知识：Spring 事务管理**

Spring 中支持两种方式事务管理：

1）基于 TransactionTemplate 的编程式事务，实际中很少使用

2）基于 xml 配置或者注解 @Transactional 的声明式事务。

声明式事务管理实际是通过 AOP 实现的，其本质是对方法前后进行拦截，然后在目标方法开始之前创建或者加入一个事务，在执行完目标方法之后根据执行情况提交或者回滚事务。

显然声明式事务管理要优于编程式事务管理，这正是 Spring 倡导的非侵入式的开发方式。声明式事务管理使业务代码不受污染，只要简单的在方法上加上 @Transactional  注解就可以获得完全的事务支持。

不过，和编程式事务相比，声明式事务确实存在一些不足，后者的最细粒度只能作用到方法级别。

当然，对于我们发布评论这个逻辑来说，作用到方法级别已经足够了。

另外，我们还可以通过 @Transactional  注解的 isolation 参数配置**隔离级别**、以及通过 propagation 参数配置**传播行为**。

示例用法如下：

@Transactional(isolation = Isolation.READ\_COMMITTED, propagation = Propagation.REQUIRED)

隔离级别主要定义了如下 5 种：

public enum Isolation {  
    DEFAULT(-1),  
    READ\_UNCOMMITTED(1),  
    READ\_COMMITTED(2),  
    REPEATABLE\_READ(4),  
    SERIALIZABLE(8);

后面四种就不多说，耳熟能详，读取未提交、读取已提交、可重复读以及可序列化。

第一个 DEFAULT 表示使用底层数据库的默认隔离级别。比如我在 Echo 这个项目中使用的数据库是 MySQL，引擎是 InnoDB，其默认隔离级别就是可重复读 REPEATABLE\_READ。

传播行为主要有 7 种，意思就是说如果在开始当前事务之前，一个事务上下文已经存在了，那么你有这 7 种选择可以指定当前事务接下来的执行行为：

public enum Propagation {  
    REQUIRED(0),  
    SUPPORTS(1),  
    MANDATORY(2),  
    REQUIRES\_NEW(3),  
    NOT\_SUPPORTED(4),  
    NEVER(5),  
    NESTED(6);

* REQUIRED：如果当前存在事务，则加入该事务；如果当前没有事务，则创建一个新的事务。
* SUPPORTS：如果当前存在事务，则加入该事务；如果当前没有事务，则以非事务的方式继续运行。
* MANDATORY：如果当前存在事务，则加入该事务；如果当前没有事务，则抛出异常。
* REQUIRES\_NEW：创建一个新的事务，如果当前存在事务，则把当前事务挂起。
* NOT\_SUPPORTED：以非事务方式运行，如果当前存在事务，则把当前事务挂起。
* NEVER：以非事务方式运行，如果当前存在事务，则抛出异常。
* NESTED：如果当前存在事务，则创建一个事务作为当前事务的嵌套事务来运行；如果当前没有事务，则该取值等价于REQUIRED。

**Service 层**

看 CommentService，就是这个方法执行了两次 DML，需要使用事务管理。



**表现层**

以下代码在 CommentController 中，只截取了一部分，其他无关代码我就没截了：



逻辑很简单，为这条评论赋值（发布人的 Id，评论的状态，发布时间）然后调用 Service 层方法，有些小白同学可能会纳闷，还有评论的内容 comment、评论针对的实体类型 entityType、实体 ID entityId、以及这条评论是针对哪个用户的（targetId），这些字段在哪里赋值了呢？

是这样的，SpringMVC 可以自动将 JSON 数据转化为 Java 对象，所以，在使用 form 表单进行提交的时候，如果前端页面的属性名（name）和实体类（Comment 类）的属性名一致，那么后端就可以直接使用实体类作为参数接收前端传值。

下面是发布对帖子（在 CommunityConstant 中定义了其实体类型为 1）的评论的部分前端代码：



下面是发布对评论（在 CommunityConstant 中定义了其实体类型为 2）的回复的部分前端代码：





* 博主小硕在读，深耕 Java，目前在维护一个教程类仓库 CS-Wiki「Gitee 官方推荐项目，现已 1.6k+ star，仓库地址：https://gitee.com/veal98/CS-Wiki」，公众号上的文章也会在此同步更新，欢迎各位前来交流学习。
* 准备春招秋招的小伙伴可以参考我的这个论坛项目 Echo「Gitee 官方推荐项目，现已 700+ star，仓库地址：https://gitee.com/veal98/Echo」。配套教程正在同步更新中，公众号后台回复 "Echo" 即可免费获取。
* 另外，虽然现在本号仍然很小，不过我还是建了一个交流群『小牛肉和它的小伙伴们』，感兴趣的各位可以下方扫码加我微信回复 "进群"，我拉你进群：





### 精选留言

暂无...