# 事务传播行为

Service 层方法调用service层的其他方法，本身service方法就是一个事务，事务中的事务如何处理？

传统的做法：将session（关于数据库的内容）传递到另外一个方法中

## PROPAGATION\_REQUIRED

--支持当前事务，如果当前没有事务，就新建一个事务。这是最常见的选择。

A方法中调用b

如果a在事务中，b就不会再起新的事务

如果a 不在事务中，会创建一个事务，回到第一种

## PROPAGATION\_NESTED

Nested的事务和他的父事务是相依的，他的提交是要等和他的父事务一块提交的。也就是说，如果父事务最后回滚，他也要回滚的。Nested事务的好处是他有一个savepoint。

A 方法中调用b方法，

A在事务里，会在b调用之前设置一个savepoint,如果b失败回滚回滚，a 会回到saveponit这个位置，选择其他分支执行

A不在事务里，执行创建事务，回到第一种

# 二、Spring事务实现演变

## 编程式事务

## 声明式事务

### Xml声明式事务

* + 1. 使用原始的TransactionProxyFactoryBean
    2. 基于tx/aop命名空间的配置

### 注解声明式事务

* + 1. 使用@Transactional注解

# 三、spring 事务管理难点剖析

## 3.1 事务嵌套调用的分析

调用本方法的的其他方法，天然融合在一个事务中，调用其他service的方法，采用传播机制

## 3.2 在多线程环境下事务方法的调用问题

一个事务方法中的多个线程的事务是相互独立的

## 3.3 Spring AOP事务增强有哪些限制

Aop是基于动态代理的，动态代理是基于jdk和cglib

Jdk的方法 除了publis方法的方法都不能代理，public static 也不能增强

Cglib private,stati,finan的方法都不能增强

解决方案：private方法最终会被public的方法调用，被事务管理器管理

Public static 和public final的方法是要小心的，如果被非事务方法调用，就会在一个非事务的环境中运行