***Day1***

什么是spring?

spring是一个容器,把每个bean(对象)与bean之间的关系,通过使用spring进行管理

核心技术:springAOP,springIOC,事务管理,DI,传播行为

事务管理只是利用了AOP的运行原理

三层架构:控制层--web层,业务逻辑层-service层,数据访问层--dao层

使用spring的好处:解耦,单例(节省内存),任何对象初始化过程,全部交给spring管理

SpringIOC:控制反转(将bean对象交给spring进行管理)

SpringIOC实验原理:反射机制(创建对象,解析xml,demo4J)

DI:依赖注入(解决对象之间的依赖关系)

Spring环境搭建:①xml②注解----主流

SpringIOC核心:反射机制

单例:JVM只能运行并创建一次对象

多例:每次运行都会创建一次对象

Spring对象默认是单例的

怎么证明对象是单例的?构造函数----反射

使用单例需要注意哪些问题?线程安全问题,可能会产生安全问题

单例:懒汉式,饿汉式

Spring使用饿汉式,线程安全

多例:不会产生线程安全问题

SpringIOC默认创建无参构造函数

Spring的注解使用

代理模式的作用:通过对目标对象的访问方式(中介)

好处:提高安全性

静态代理(需要生成代理类)与动态代理(不需要生成代理类)

动态代理又分为jdk动态代理(反射机制)和cglib动态代理(字节码)

SpringAOP:面向切面编程

应用场景:权限控制,事务管理,日志打印,性能统计

什么地方才会在项目中使用aop?代码重复

不同方法中需要做相同的操作时使用AOP

AOP:关注点:代码重复

切面:抽取重复代码

切入点:拦截哪些方法

SpringAOP注解:①aspect定义切面

②compent注入spring容器

SprinAOP的通知:前置通知,后置通知,异常通知,环绕通知

***Day2Spring基础知识***

三层架构:控制层,业务逻辑层,数据库访问层

二层架构:数据库访问层与业务逻辑层,页面控制层

一层架构:操作数据,业务逻辑,页面控制全部写在一个类里,一个方法里面

mvc架构:视图层(view),模型层(model),控制层(controller),它是一种软件架构思想,不是设计模式,而设计模式:重构代码,提高代码的复用

Spring:把每个bean与bean的关系全部交给第三方容器进行管理,那么这个容器就是spring,整个对象的生命周期由spring管理

Spring的核心:AOP(面向切面编程),IOC(控制反转),DI(依赖注入)

IOC与DI的区别:DI注入,IOC是负责创建对象

Spring集成redis

SpringIOC原理:①解析xml位置,获取bean的class地址②使用java的反射机制,进行实例化对象③返回对象

如何证明spring是单例还是多例?构造函数

Spring是单例的,但会出现线程安全问题(单例在spring中使用),可以改为不是单例的

Spring中的Springid重复怎么办? 会报错:①在xml中的beanId是不允许重复的②注解和xml同时使用不会报错

Spring中有四个生命周期:

singleton,prototype(当调用getbean方法才会初始化),request(http请求request里面使用),session(http请求session里面使用)

SpringIOC创建对象有哪些方式?①构造函数(有参,无参)②工厂③注解

DI依赖注入:如何给对象的属性赋值?①构造函数②set方法③P名称空间④自动装配⑤注解

Set(属性)注入原理?属性要有一个set方法

SpringAOP:面向切面编程,应用场景:权限,日志,控制.事务的原理就是AOP

代理设计模式:静态代理与动态代理(jdk---反射,cglib---asm,都不需要生成代理类)

**Spring注解**

Springboot简化xml配置,全部用注解代替,而且是微服务架构

@Resourse:是jdk1.6,默认以名称查找,再以类型查找

@autiwer:spring框架自带,默认以类型查找

**AOP编程**

AOP编程的作用:把代码进行分离

①关注点②切面③切入点

注解:@Aspect:指定一个切面类

@Pointut:指定切入点

@Before:前置通知

@After:后置通知

@AfterReturning:返回后置通知

@AfterThrowing:异常通知

@Around:环绕通知

**Spring事务**

什么是事务?一个执行单元

为什么要有事务?事务就是保证数据的一致性

事务的特性?一致性,原子性(不可分割的),持久性,隔离性(互不影响)

JdbcTemple:对jdbc做轻量级的封装

项目中如何使用事务?

事务分类:编程式事务,声明式事务:xml方式,注解方式

方法结束,提交事务

声明式事务,当抛出异常时才会回滚

**Spring的传播行为**

* 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。
* PROPAGATION\_SUPPORTS--支持当前事务，如果当前没有事务，就以非事务方式执行。
* PROPAGATION\_MANDATORY--支持当前事务，如果当前没有事务，就抛出异常。
* PROPAGATION\_REQUIRES\_NEW--新建事务，如果当前存在事务，把当前事务挂起。
* PROPAGATION\_NOT\_SUPPORTED--以非事务方式执行操作，如果当前存在事务，就把当前事务挂起。
* PROPAGATION\_NEVER--以非事务方式执行，如果当前存在事务，则抛出异常。