实体对象一般不会使用IOC 里面涉及到数据的。

那么数据就要写在xml配置文件中。写死了。

业务层，数据层，可能会改变的情况，使用IOC，以便将来要切换实现。

jdbc

mybatis

hinernate

举业务层的例子

工厂模式的一个框架。

自己不需要通过反射机制+接口+工厂模式

只要使用spring框架就能容易的实现这个模式

IOC核心技术

ioc基础概念，获取，配置，方式，注入方式，类型，。获取的类型等等，单例的。还可以是多例的。

以前我们都是通过xml配置文件设计的

业务层，数据层，配置层等

都写在一个xml中，数据就越来越多了。

对配置文件也会进行分层

applicationContext-service.xml

applicationContext-dao.xml

applicationContext-config.xml

applicationContext.xml 总的入口配置文件，导入其他的配置文件

ref bean / local

一个是从当前这个配置文件中找bean

从全局的配置文件中找bean

# AOP面向切面

开发模式

面向过程

面向对象

面向组件

面向切面

面向服务

jdk提供的代理类，必须要使用接口

CGLIB第三方代理类，可以不用接口。

在调用目标对象之前，可以生成一个代理对象，通过反射机制，调用目标对象中的方法

周边功能，就是切面 ，把它单独放在一个类中，切面类

面向切面类编程 ，简称 面向切面

类 对象 面向对象

面向方法之前之后操作

编织 织入

切入

spring aop 如果有接口，他就使用jdk动态代理，如果没有接口，那么使用CGLIB第三方代理

切入了这个jar 不需要单独下载

切入点

在哪个包中的哪个类中的哪个方法上要用这个面向切面

之前做什么，之后做什么。

抽出来，定义一个类。定义方法，之前() 之后()

通知类型

前置通知

后置通知

环绕通知

# 整合mybatis

使用spring ioc/aop 整合mybatis的操作

session.insert()

session.selectOne()

映射到配置文件中

通过接口生成代理对象

<bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">

<property name="basePackage" value="com.bwf.mapper"/>

</bean>

扫描这个包中的接口变成对象，并且自动存入spring容器