## Spring框架学习笔记

1. Spring框架简介

1）概述：Spring框架是由于软件开发的复杂性而创建的。Spring使用的是基本的JavaBean来完成以前只可能由EJB完成的事情。然而，Spring的用途不仅仅限于服务器端的开发。从简单性、可测试性和松耦合性角度而言，绝大部分Java应用都可以从Spring中受益。

\* 目的：解决企业应用开发的复杂性

\* 功能：使用基本的JavaBean代替EJB（重量级框架），并提供了更多的企业应用功能

\* 范围：任何Java应用

总结：Spring是一个轻量级控制反转(IoC)和面向切面(AOP)的容器框架。

2）分层概念：

\* web层：struct2、**springMVC**

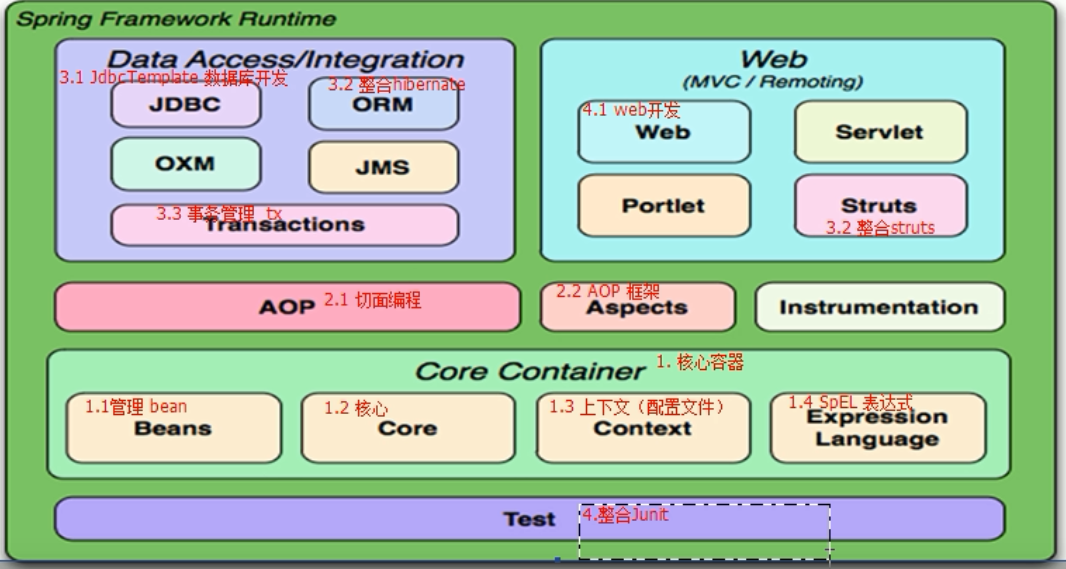
\* service层：**spring**

\* dao层：hibernate、**mybatis**、jdbcTemplate、springdata

1. Spring核心（重点）
2. spring的核心就是(IoC)控制反转和(AOP)面向切面
3. Spring优点
4. 方便解耦，简化开发（高内聚、低耦合）

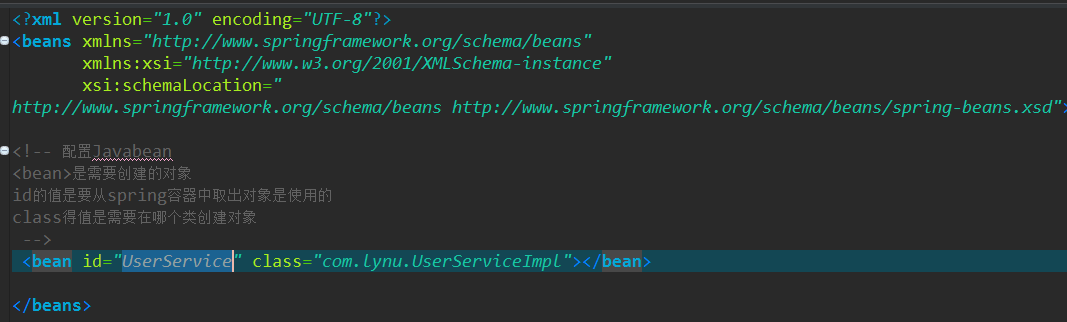
\* spring就是一个大工厂，用来生产bean

1. AOP编程的支持
2. 声明式事务的支持
3. 方便测试
4. 集成了各种优秀的框架
5. 降低了JavaEEAPI的使用难度
6. Spring体系结构图



**注：beans、core、context、expression需要重点掌握**

1. Spring IoC（控制反转）实例



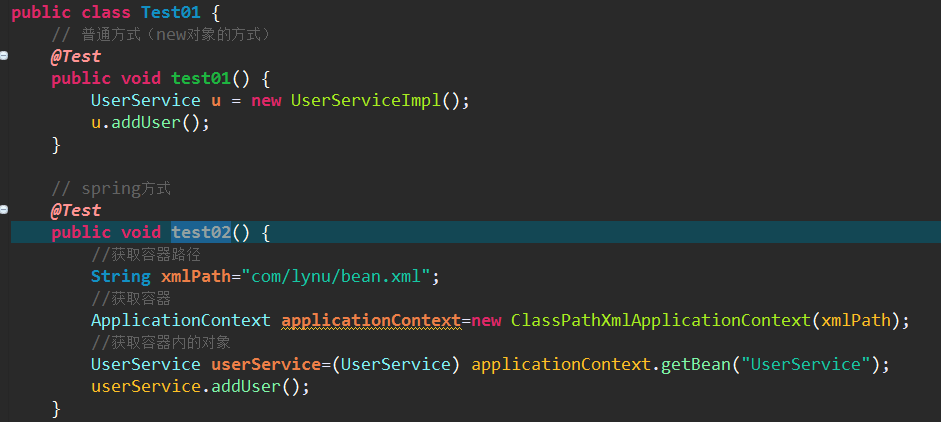
**Spring配置文件：**

**\* 位置：任意，开发中一般在src下**

**\* 名称：任意，开发中常用applicationContext.xml**

**\* 内容：schema约束**

**\* 约束文件位置：spring-framework-3.2.13.RELEASE\docs\spring-framework-reference\html 下的xsd-config.html中The equivalent file in the XML Schema-style would be...中的代码**



**注意：当测试类从spring容器中获取对象的时候，就会在xml配置文件中找，默认地从src目录下寻找，所以xmlPath要写成src下面的路径**

**总结：spring的IoC（控制反转）其实就是把需要我们自己new对象的事情反转给了spring，也就是创建对象的事情交给了spring来做，通过xml配置文件的方式。**

1. Spring DI（依赖注入）实例
2. 依赖：一个对象需要使用另一个对象

\* class B {

Private A a；//B类依赖于A类

}

1. 注入：通过setter（）方法对另一个对象实例设置，注入其实就是通过调用方法把另一个对象注入（添加）给当前对象
2. 例子：

class BookServiceImpl {

// 以前开发：接口=new 实现类

private BookService = new BookServiceImpl（）；

// spring之后：接口+setter方法

}

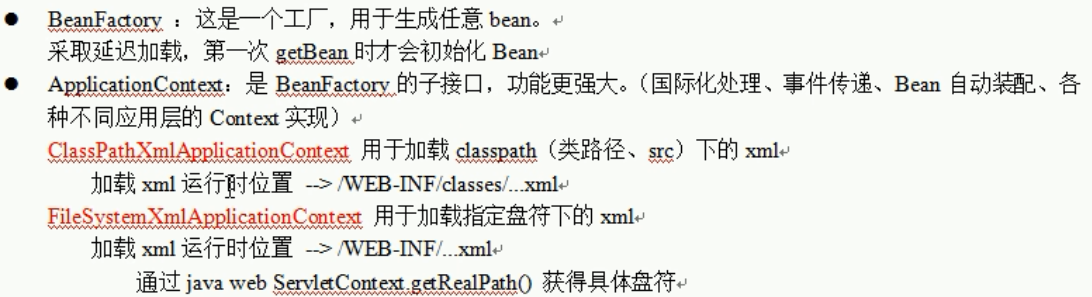
1. spring执行流程

\* 创建service实例：BookService bookService=new BookServiceImpl（）；-->IoC做

\* 创建dao实例：BookDao bookDao=new BookDaoImpl（）；-->IoC来做

\* 将bookDao设置给bookService：bookService.setBookDao （bookDao）；-->DI做

1. Spring核心API（了解即可）



一、基于XML装配Bean（重点）

1.bean的实例化方式（也就是创建对象的方式）（3种）

1）默认构造

<bean id="bookDao" class="cn.lynu.xy.di.BookDaoImpl"></bean>，这就属于默认构造

说明：id表示对象的唯一标识符，从spring中取出对象时使用；

Class：表示需要创建的对象所在的类

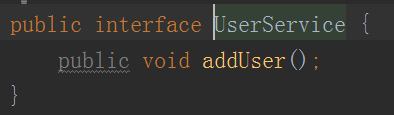
1. 静态工厂

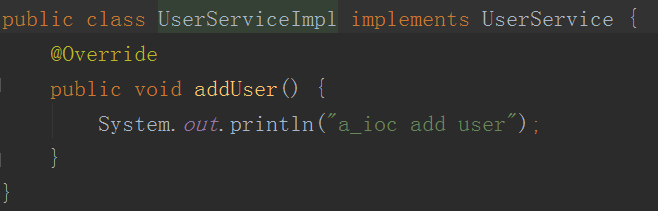
作用：常用于spring整合其他框架；用来生产实例对象，所有的方法必须是静态的

写法：<bean id="bookDao" class="工厂的全限定类名" factory-method="静态方法"></bean>

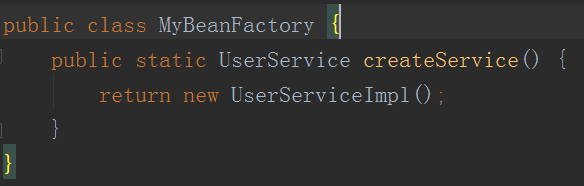
1. 使用静态工厂创建实例的步骤：

\* 编写接口和实现类

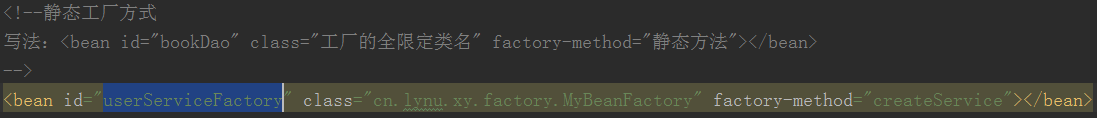




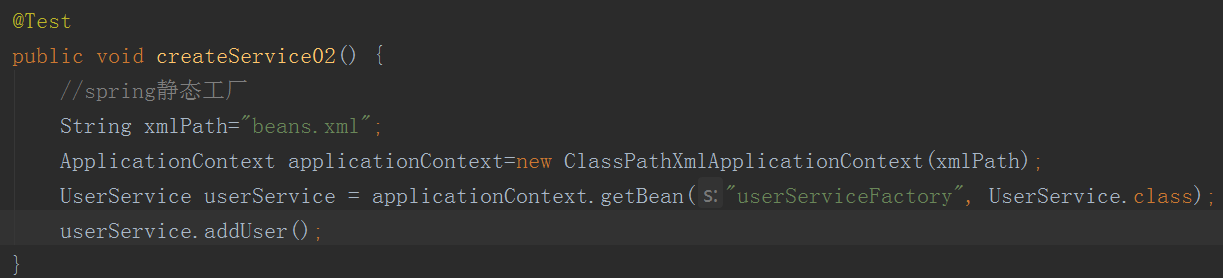
\* 创建静态工厂：



\* 在xml文件中配置



\* 测试

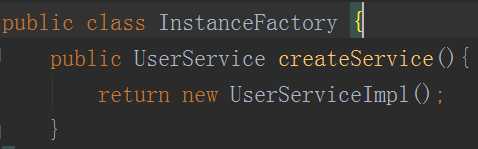


1. 实例工厂

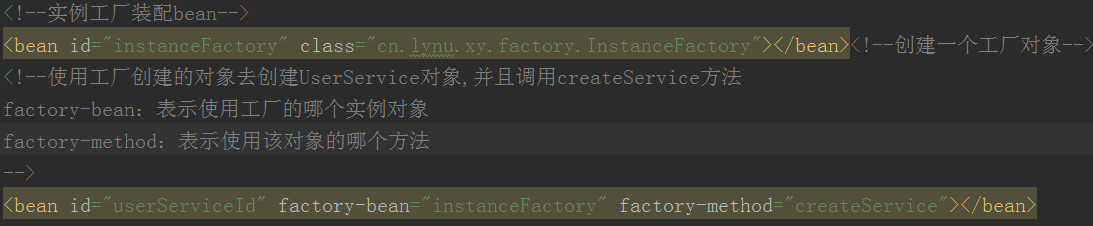
特点：必须先有工厂的实例对象，通过该对象再创建我们所需要的对象，提 供的所有方法都是“非静态的”

\* 编写接口和实现类

\* 创建实例工厂



\* xml中装配bean



\* 测试