**spring**是开源的轻量级框架（轻量级：不依赖于其他东西），是一站式框架。

核心主要两部分：

1.AOP：面向切面编程，扩展功能不是通过修改源代码实现

2.IOC：控制反转，对象的创建不是通过new实现，而是交给spring配置创建类对象

一站式框架：spring在javaee三层结构中，每一层都提供不同的解决技术 1.WEB层：Springmvc

2.service层：spring的IOC 3.dao层：spring的JDBCTemplate

spring版本：1.hibernate 5.x 2.spring 4.x

Spring的IOC操作：

1.把对象的创建交给spring进行管理

2.IOC操作两部分：i>IOC的配置文件方式

ii>IOC的注解方式

IOC底层原理：

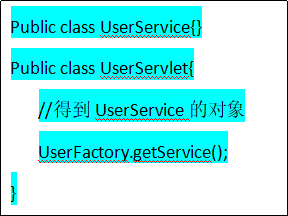
使用的技术：（1）XML配置文件

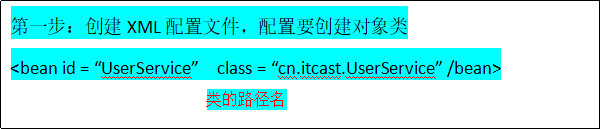
（2）DOM4J解析XML

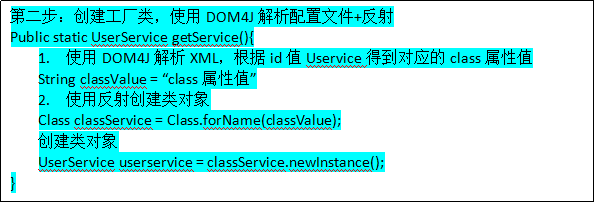
（3）工厂设计模式

（4）反射

**底层原理**

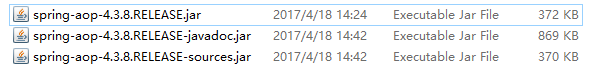




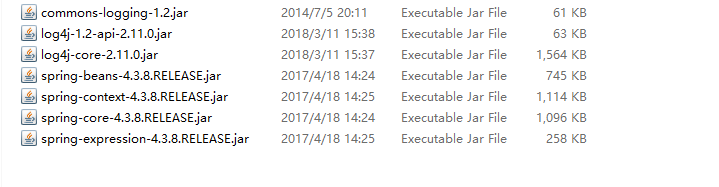


**IOC：**

1.导入jar包，做spring最基本功能的时候导入四个核心的jar包就可以了：beans core context SpEL



要导入的jar包

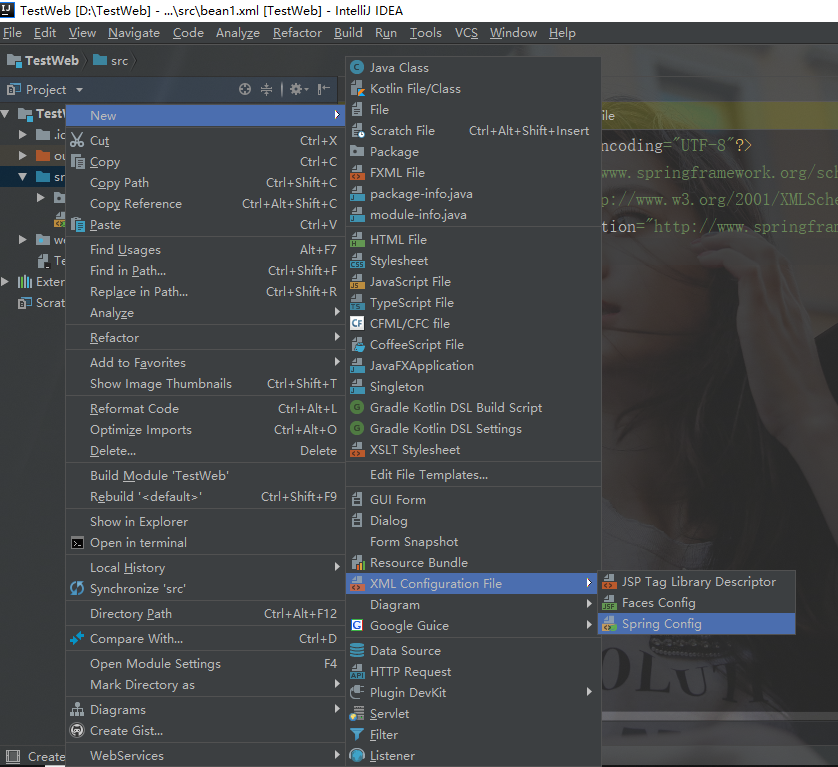


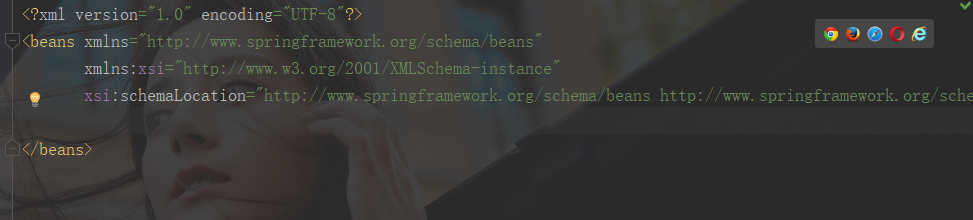
2.创建具体类

3.创建spring配置文件，配置创建类

（1）spring核心配置文件名称和位置是不固定的，建议放在src下

（2）引入schema约束（eclipse要自己引入约束，然而IDEA已经在你创建spring配置文件的时候自动帮你引入了）

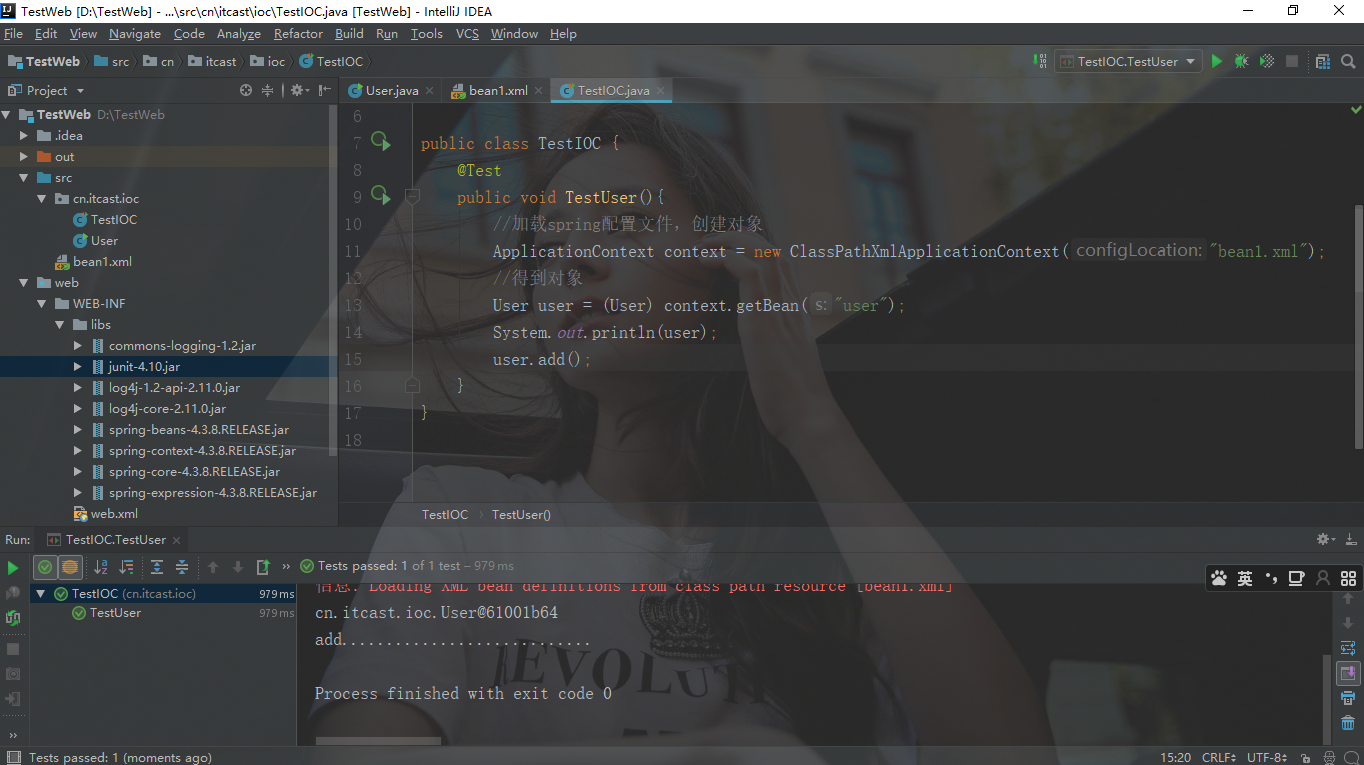




（3）配置对象创建

QQ截图20180624135605.png

4.写代码测试对象创建——这段代码在测试中使用



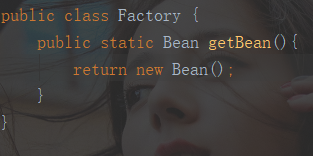
**Spring的bean管理（XML方式）**

**bean实例化的三种方式：在spring里面通过配置文件创建对象**

**1、使用类的无参数构造函数**

QQ截图20180624142522.png

**2、使用静态工厂创建**



2.png

**3、使用实例工厂创建**

Bean标签常用的属性

（1）id属性：起的一个名称，但不可以是中文，不可以包含特殊符号，根据id的值得到配置对象

（2）class属性：要创建对象的所在类的全路径

（3）name属性：功能和id一样，但是name可以包含特殊符号

（3）scope属性：bean的范围

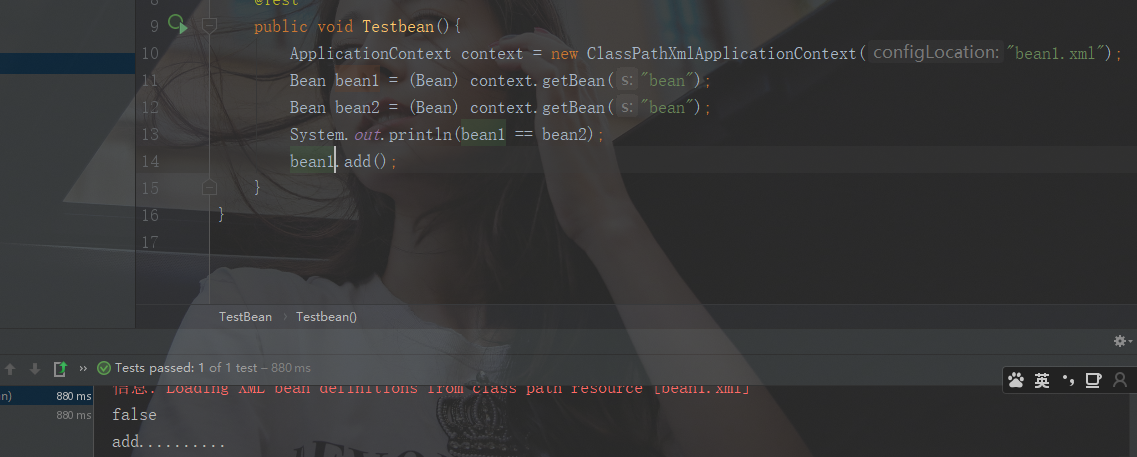
singleton(默认值)单例——只能创建一个对象

2.png



prototype 多例的 ——创建多个不同的对象

4.png



request：创建对象把对象放在request域里面

session: 创建对象把对象放在session里面

globalsession:创建对象把对象放在globalsession域里面(单点登录)

**属性注入：创建对象的时候，向对象的属性中添加值**

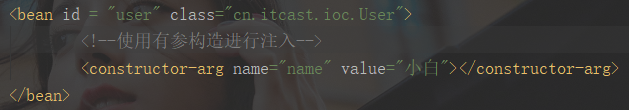
**三种方式：1.set方法（用的最多）**

**2.有参构造**

**3.使用接口注入**

**spring框架只支持前两种方法**

**有参构造注入属性**



**set方法注入属性**



**在一个类中注入对象类属性（重点）**

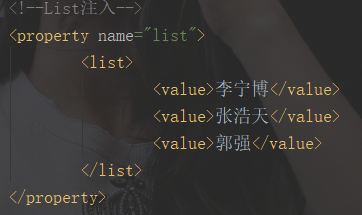




**注入复杂类型**

**数组、List、Map、Properties**







**IOC与DI的区别**

IOC：控制反转，把对象的创建交给配置文件完成

DI：依赖注入，向类的属性中设置值

两者之间的关系：DI不能单独存在，它需要在IOC的基础之上来完成

**spring整合web项目原理**

1.加载spring核心配置文件

13.png

2.实现思想：把加载配置文件和创建对象过程放在服务器启动的时候完成

3.实现原理：

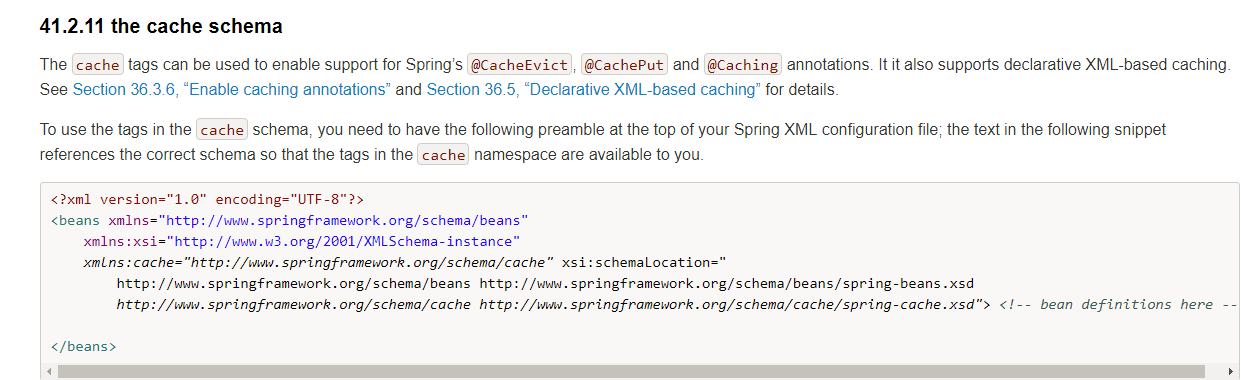
i>ServletContext对象

ii>监听器

iii>具体使用：在服务器启动时，为每一个项目创建一个ServletContext对象；在ServletContext对象创建的时候，使用监听器可以具体知道ServletContext对象什么时候创建；当监听器听到ServletContext对象创建时，加载spring配置文件，把配置文件中的配置对象创建；把创建出来的ServletContext对象放在ServletContext域中（setAttribute方法），获取对象通过ServletContext域中得到（getAttribute方法）

Spring的bean管理

约束条件：



注解：1.代码中的一些特殊的符号

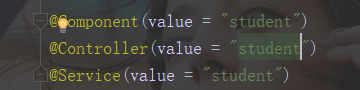
2.注解的写法：@注解名称（属性名称 = 属性值）

3.使用在类、属性、方法上面

在配置文件中开启注解扫描



注入对象



注入属性

@Autowired

@Resource

1、当启动spring容器的时候，spring容器加载了配置文件

2、在spring配置文件中，只要遇到bean的配置，就会为该bean创建对象

3、在纳入spring容器的范围内查找所有的bean,看哪些bean的属性或者方法上加 有 @Resource

4、找到@Resource注解以后，判断该注解name的属性是否为”“(name没有写)

如果没有写name属性，则会让属性的名称的值和spring中ID的值做匹配，如果匹配 成功则赋值

如果匹配不成功，则会按照类型进行匹配，如果匹配不成功，则报错

如果有name属性，则会按照name属性的值和spring的bean中ID进行匹配，匹配成功，则赋值，不成功则报错

**配置文件和注解的混合使用**

**1.创建对象操作使用配置文件方式实现**

**2.注入属性的操作使用注解方式实现**

**AOP**

**1.概念：面向切面编程，采用横向抽取机制，底层是用动态代理模式，**

**有接口情况：创建接口实现类的代理对象**

**无接口情况：创建子类的代理对象**

**2.术语：**

**（1）切入点：在类里面可以有很多方法被增强，实际增强的方法称为切入点**

**（2）增强（通知）：实际增加的逻辑功能，例如要在一个方法中添加日志功能，那么这个方法叫做切入点，日志功能叫做增强**

**前置通知：在方法之前执行**

**后置通知：在方法之后执行**

**异常通知：方法出现异常**

**最终通知：在后置通知之后执行**

**环绕通知：在方法之前和之后执行**

**（3）切面：把增强应用到具体方法上面，这个过程称为切面，即增强用到切入点的过程**

**（4）连接点：类中可以被增强的方法**

**3.Spring的AOP操作：使用aspectj（面向切面的框架）操作**

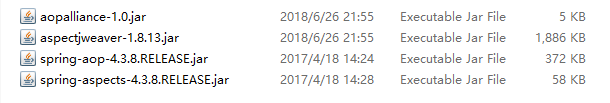
**aspectj不是spring的一部分，他是和spring一起实现AOP的，spring2.0之后才对aspectj增加了支持**

**使用aspectj实现AOP有两种方式：**

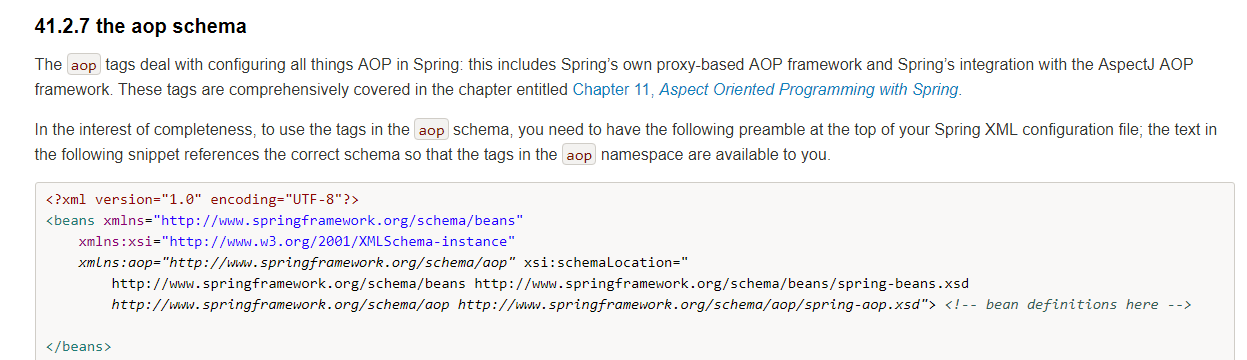
**（1）基于aspectj的xml配置**

**（2）基于aspectj的注解方式**

**AOP操作的准备：导入相关jar包**



**创建spring核心配置文件，加约束**



**使用表达式配置切入点：**

**常用的表达式：execution(<访问修饰符>?<返回类型><方法名>(<参数>)<异常>)**

**例如：匹配所有类的public方法execution(public \* \*(..))**

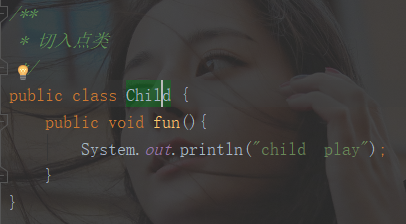
**匹配指定包下的所有方法execution(\* cn.itcast.aop.\*(..))不包含子类**

**匹配具体类的具体方法execution(\* cn.itcast.aop.Book.add(..))**

**匹配所有save开头的方法execution(\* save\*(..))**

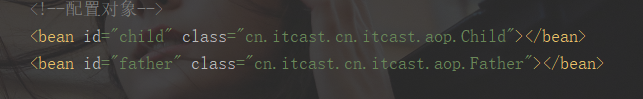
**AOP配置：**

**先准备两个类：一个切入点类，另一个增强类**





开始配置两个对象



**AOP配置**

