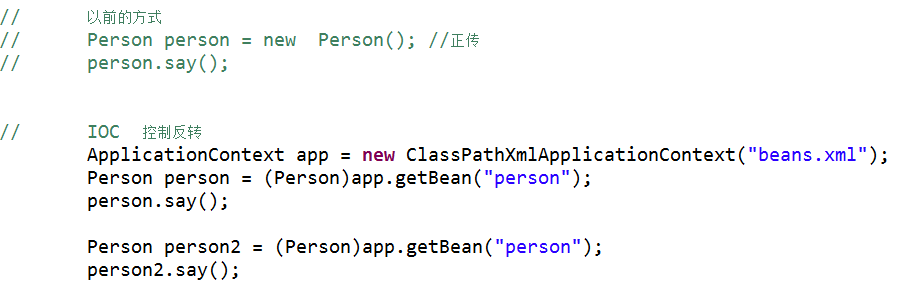
# note总结：

今日重点

## 1：IOC 控制反转

**所谓控制反转就是应用本身不负责依赖对象的创建及维护，依赖对象的创建及维护是由外部容器spring负责的**

<bean id=*"personService"* class=*"com.offcn.service.impl.PersonServiceImpl"*></bean>



那么反转就是不用去new对象，而是通过反射获取,class中的person对象

## 2：DI依赖注入

所谓依赖注入就是指：在运行期，由外部容器动态地将依赖对象注入到另一个对象的组件中。

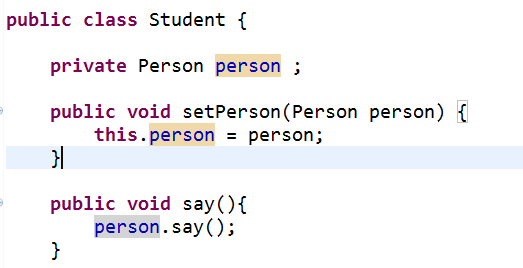
beans.xml中的配置：

<bean id="student" class="com.offcn.pojo.Student">

//ref是在bean的内部引用另一个bean对象：这就是依赖注入

<property name="person" ref="person"></property>

</bean>



## 3注解

主要成员：

**@Service用于service层、相当于 xml配置中得bean**

**服务层组件，用于标注业务层组件,表示定义一个bean，自动根据bean的类名实例化一个首写字母为小写的bean，例如Chinese实例化为chinese，如果需要自己改名字则:@Service("你自己改的bean名")。**

**@Controller用于controller层 用于标注控制层组件(如struts中的action)**

**@Repository用于dao层 持久层组件，用于标注数据访问组件，即DAO组件**

**@Component任意哪一层 当组件不好归类的时候，我们可以使用这个注解进行标注。**

@Autowired从容器中获取对象，先匹配类型没在匹配名字

@Resource 从容其中获取对象，先匹配名字，再匹配类型

getBean 的默认名称是类名（头字母小写），如果想自定义，可以@Service(“aaaaa”) 这样来指定

bean默认是单例的，如果想改变，可以使用@Service(“beanName”) @Scope(“prototype”)来改变。

可以使用以下方式指定初始化方法和销毁方法（方法名任意）：

@PostConstruct

@PreDestroy

Spring不但支持自己定义的@Autowired注解，还支持几个由JSR-250规范定义的注解，它们分别是@Resource、@PostConstruct以及@PreDestroy。

@Resource的作用相当于@Autowired，只不过@Autowired按byType自动注入，而@Resource默认按 byName自动注入罢了。@Resource有两个属性是比较重要的，分是name和type，Spring将@Resource注解的name属性解析为bean的名字，而type属性则解析为bean的类型。所以如果使用name属性，则使用byName的自动注入策略，而使用type属性时则使用byType自动注入策略。如果既不指定name也不指定type属性，这时将通过反射机制使用byName自动注入策略。

@Autowired 与@Resource的区别：

1、 @Autowired与@Resource都可以用来装配bean.都可以写在字段上,或写在setter方法上。

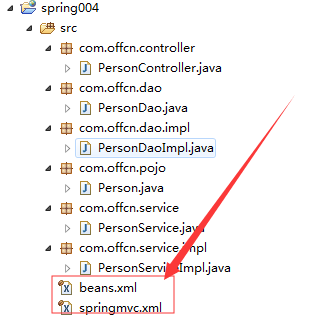
2、 @Autowired默认按类型装配（这个注解是属于spring的），默认情况下必须要求依赖对象必须存在，如果要允许null值，可以设置它的required属性为false

3、@Resource（这个注解属于J2EE的），默认安装名称进行装配，名称可以通过name属性进行指定，如果没有指定name属性，当注解写在字段上时，默认取字段名进行安装名称查找，如果注解写在setter方法上默认取属性名进行装配。当找不到与名称匹配的bean时才按照类型进行装配。但是需要注意的是，如果name属性一旦指定，就只会按照名称进行装配。

推荐使用：@Resource注解在字段上，这样就不用写setter方法了，并且这个注解是属于J2EE的，减少了与spring的耦合。这样代码看起就比较优雅。

测试：

第一步：项目结构图：



第二步：

web.xml中的配置

<!-- 加载springmvc -->

<servlet>

<servlet-name>cms</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:springmvc.xml</param-value>

</init-param>

</servlet>

<servlet-mapping>

<servlet-name>cms</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>

<!-- spring监听 -->

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:beans.xml</param-value>

</context-param>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

beans.xml中的配置

<context:annotation-config></context:annotation-config>

<context:component-scan base-package=*"com.offcn.\*"*>

</context:component-scan>//扫描其他包

springmvc.xml的配置

<!--开启注解： -->

<context:annotation-config></context:annotation-config>

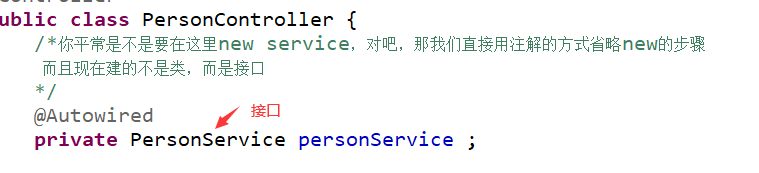
<!-- <mvc:annotation-driven></mvc:annotation-driven> -->

<context:component-scan base-package=*"com.offcn.controller"*>

</context:component-scan>//扫描*controller*包

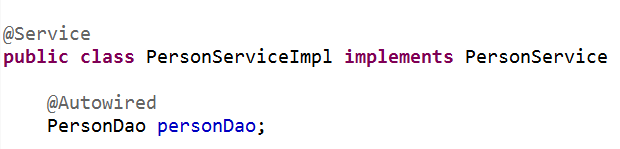
第三步：类中的代码

controller层：



service实现类层

service实现类 要用service注解，**相当于 xml配置中得bean**



dao层

@Repository ==//<bean id="personDao" class="PersonDpa">



@Autowired相当于new对象了 将Spring容器中的bean自动的和我们需要这个bean的类组装在一起协同使用

它可以对类成员变量、方法及构造函数进行标注，完成自动装配的工作。 通过 @Autowired的使用来消除 set ，get方法。

**@Repository解修饰哪个类，则表明这个类具有对对象进行CRUD（增删改查）的功能，而且@Repository是@Component注解的一个派生品，所以被@Repository注解的类可以自动的被@ComponentScan 通过路径扫描给找到。**