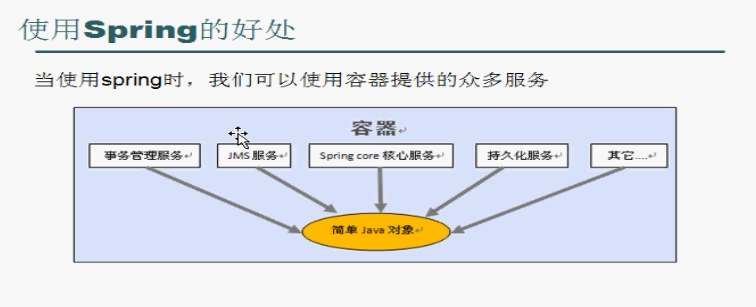
使用javaBean代替EJB

1. Spring的核心分两部分

(1). Aop： 面向切面编程， 扩展功能不是修改源代码实现

(2). Ioc：控制反转----->把类对象的创建权限交给了spring（容器）



1. spring是一站式框架
2. . 三层中，都提供了不同的解决技术
   1. Web层： springMVC
   2. Service层： ioc
   3. Dao层：spring的jdbcTemplate
3. 版本问题

使用的是4 版本

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：
   1. Ioc的配置文件方式
   2. Ioc的注解方式

IOC底层原理

1. ioc底层原理使用技术
   1. Xml配置文件
   2. Dom4j解决xml
   3. 工厂设计模式
   4. 反射
2. 画图分析ioc底层原理

直接new ---> 使用工厂模式进行解耦合(但是是service和工厂类耦合)

--------> ioc 控制反转

Ioc原理：

第一步： 创建xml配置文件，配置要创建对象类

第二步： 创建一个工厂类， 使用dom4j 解析配置文件+ 反射



ApplicationContext context = new ClasspathXmlApplicationContext(“xml路径”);

context.getBean(“xml -->id...”);

配置文件没有提示：引入文件到eclipse中

Bean 实例化方式

1. 无参构造
2. 静态工厂
3. 实例工厂

重点： 注入对象类型属性

把new的过程交给spring管理

1. 在service中 使用

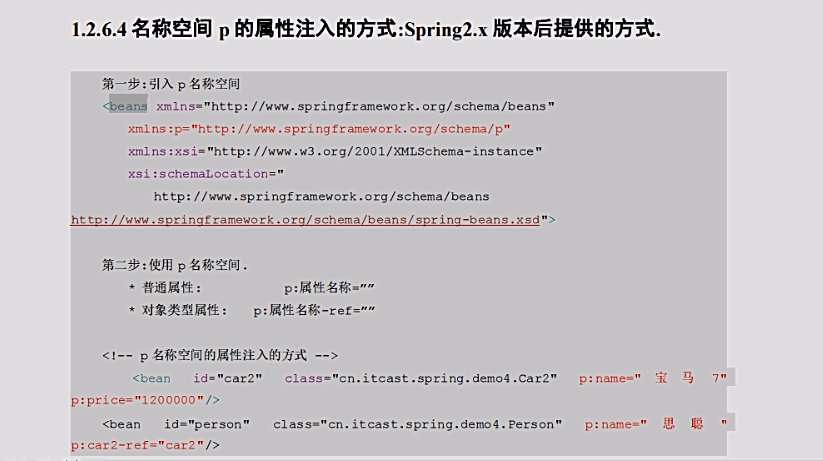
<bean ... >

<property name="filters" value="stat" />

<property name="filters2" ref="..." /> <!-- 引用 -->

</bean>

2. p 名称空间注入



注入复杂类型

1. 数组
2. list集合
3. Map
4. Properties

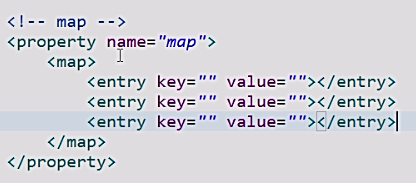
生成set方法

第一步： 创建类的对象

第二步： 属性注入









IOC 和 DI 区别

1. IOC： 控制反转, 把对象创建权交给spring进行配置

1. DI: 依赖注入， 向类里面的属性设置值
2. 关系： 依赖注入不能单独存在，需要早ioc基础之上进行操作

第一次效率特别低，

实现思想： 把加载配置文件和创建对象过程，在服务器启动时完成

实现原理：

(1). ServletContext 对象

(2). 监听器

在服务器启动时候， 为每个项目创建一个ServletContext对象

在ServletContext对象创建时候， 使用监听器可以具体到ServletContext对象在什么时候创建

使用监听器听到ServletContext对象创建时候

加载spring配置文件， 把配置文件配置对象创建

把创建出来的对象放到ServletContext域对象里面（setAttribute）

获取对象时候，到ServletContext域中获取（getAttribute）