**目的: 为了使以下代码可以走通**

ApplicationContext ctx = new ClassPathXmlApplicationContext("petstore-v1.xml");

PetStoreService petStore = (PetStoreService)ctx.getBean("petStore");

assertNotNull(petStore.getAccountDao());

assertNotNull(petStore.getItemDao());

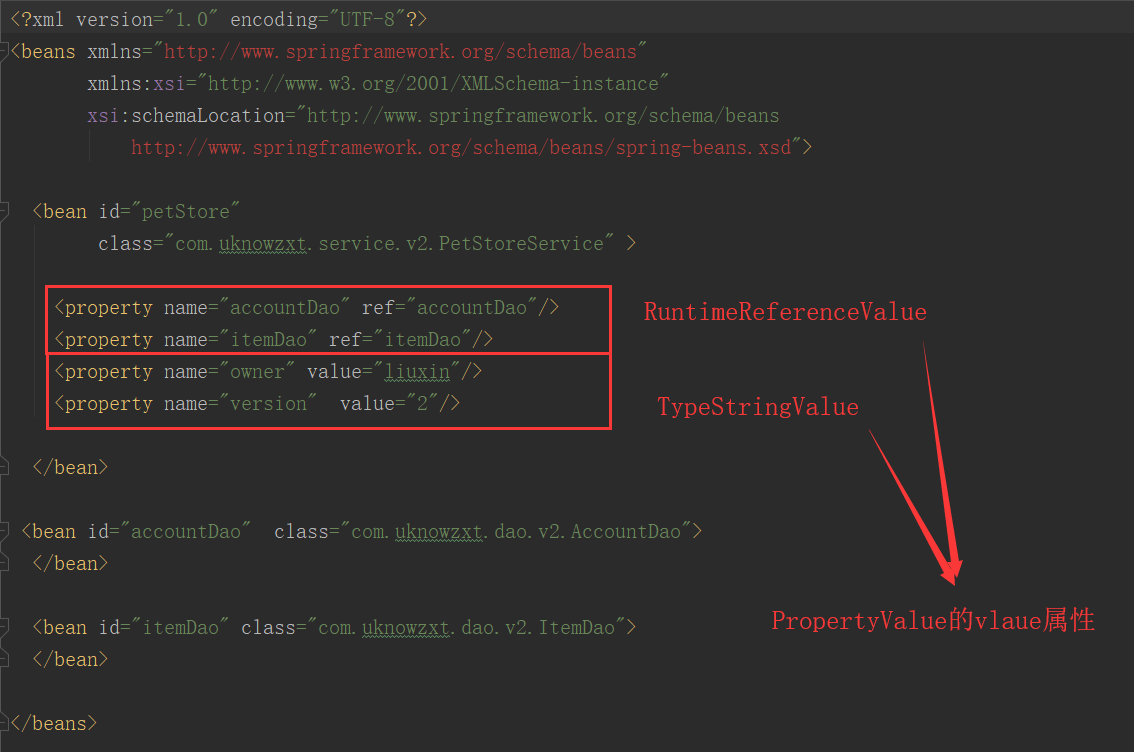
assertTrue(petStore.getAccountDao() instanceof AccountDao);

assertTrue(petStore.getItemDao() instanceof ItemDao);

assertEquals("liuxin",petStore.getOwner());

assertEquals(2, petStore.getVersion());

Xml如下所示:

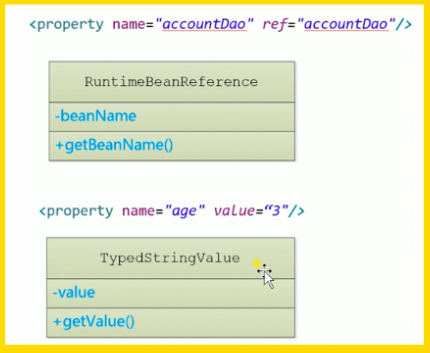
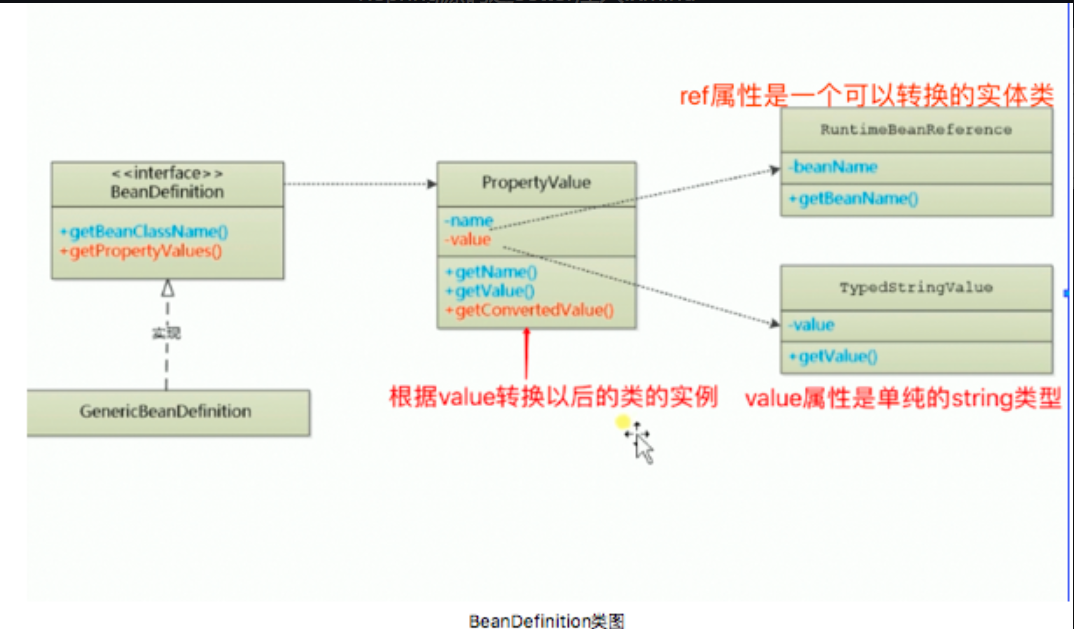


**实现思路:**

1. 从xml中解析出<bean>的属性值 . 所有的属性都解析为PropertyValue(name, value), 其中value有两种, 一种是ref标签, 可以转换为一个实体类的. 一种是value标签, 直接赋值的. 并把属性值放入BeanDefinition的List<PropertyValue> getPropertyValues()成员变量中
2. 在getBean的时候, 处理属性值, 区分是哪一种属性值, 获得该属性实际的值(RuntimeReference需要获得一个具体的bean的实例, TypeStringValue需要对各种数据类型进行转型).
3. 最终把获得的属性值, 设置给目标类

**拆解出几个点: 正叙**

Xml中新配置了关于属性的标签, property, 但是标签的属性有的写的是ref, 有的写的是value. 这决定了他们获取这个属性标签代表的真实的值的方式的不同. 所以有了以下结构:



剩下的事情就很明显了, 更新xmlBeanDefinitionReader方法, 加入属性标签的处理, 把得到的RuntimeReference或者TypeStringReference包装成PropertyValue, 加入到这个类的BeanDefinition的成员变量 List<PropertyValue>中.

具体做法:

遍历bean的时候加入parsePropertyElement(ele,bd);//解析属性,并把属性放入bd的list

二.涉及:

1.在getBean的时候, 不能只关心bean本身是什么了. 还需要关心bean的属性. 所以我们在createBean(BeanDefinition bd)之时, 也要加入populateBean(bd, bean)方法. 用来给创建好的bean设置属性.

2.不同的属性如何转化呢. 这时候需要一个单独的类对属性进行转化. 要创建一个起到转化器作用的类. 其中一个转化器是把RuntimeReference转化成代笔的bean, 把TypeStringValue转化成原始字符串.

另外一个转化器是为了处理TypeStringValue类型的值. xml中是value属性的值. 因为有一些数字类型, 但是写出来的时候依然是字符串. 这样的值直接向属性中设置是设置不进去的. 所以要把这种数字类型的字符串转化成合适的数字类型, long/float/int....

3.转化RuntimeReference的时候我们创建的是BeanDefinitionValueResolver, BeanDefinitionValueResolver我们对它的期待是, 把propertyVaule的value(RuntimeReference/typeStringValue)传进去, 就可以得到一个转化好的对象. 转化好指的是, RuntimeReference转换成一个实体bean. TypeStringValue直接获得它代表的字符串

4.而转化原本是 TypeStringValue类型的value的时候(这个时候value已经被BeanDefinitionValueResolver转化为字符串了), 我们创建的是SimpleTypeConverter, SimpleTypeConverter我们对他的期待是, 传入待转化的字符串以及要转化出的类型, 得到正确的类型的值. -------如果给的字符串就是要转化出的类型, 那么直接转化直接返回. 不然的话取到一些editor. 这些editor是java原生的,根据类型选取对应的editor, 像editor中设置字符串, 最终就会转化出对应类型字符串代表的值.

5.现在所有的障碍都不存在了.

三 最终要做的就是调用bean对应属性的Setter方法, 把属性值设置进去了