【第33话：Spring中Bean实例化的几种方式】

Hello小伙伴们，下面给大家讲解下，当面试官问我们："Spring中Bean实例化的几种方式"时，我们应该如何进行回答。

在Spring中实例化Bean有两种方式：

- 通过构造方法进行实例化。默认情况时使用无参构造。这种方式和以前new的方式是等效的。只需要在XML中通过`<bean>`的class属性指定类的全限定路径，然后就可以实例化对象。
- 通过工厂进行实例化。可以通过静态工厂和实例工厂进行实例化。这种方式完全是根据设计模式中工厂模式的思想而研发出的。Spring考虑到如果需要频繁实例化某个类的对象，工厂模式无疑是一个好选择。

使用构造实例化

使用构造器注入必须保证类中包含对应的构造器。

如果类中包含无参构造器，可以直接使用`<bean>`标签进行实例化。

具体代码就是这样的，这时会实例化People类，但是不会给类中属性设置值。

```
<bean id="people" class="com.bjsxt.pojo.People"></bean>
```

也可以使用类的有构造方法。结合`<constructor-arg>`子标签对构造器的参数设置值。`<constructor-arg>`里面除了name是通过名称对应参数，也可以通过type与参数类型对应，index与参数索引位置对应。

```
    <bean id="people" class="com.bjsxt.pojo.People">
        <constructor-arg name="id" value="1"></constructor-arg>
        <constructor-arg name="name" value="smallming"></constructor-arg>
    </bean>
```

使用工厂默认实例化

使用工厂模式又分为静态工厂和实例工厂两种方式，所以下面分成两部分进行演示。

1.使用实例工厂实例化bean

1.1 创建工厂类

实例工厂方式需要实例化工厂类。所以工厂中创建bean的方法是一个实例方法。

在项目中创建实例工厂类，并编写方法返回创建的实例对象。

示例中创建com.bjsxt.factory.PeopleFactory

```
package com.bjsxt.factory;

import com.bjsxt.pojo.People;

public class PeopleFactory {
    People peo = new People();
    public People getInstance(){
        return peo;
    }
}
```

1.2 配置配置文件

在applicationContext.xml中先配置工厂实例，然后通过工厂实例产生自己想要的bean对象。

> 小提示：
>
> 实例工厂方式peo2的`<bean>`标签没有class属性。

```
    <!-- 创建工厂实例 -->
    <bean id="factory" class="com.bjsxt.factory.PeopleFactory"></bean>
    <!-- factory-bean:工厂对象的id    factory-method：创建当前bean的方法名称 -->
    <bean id="peo2" factory-bean="factory" factory-method="getInstance"></bean>
```

1.3 在测试类中测试效果

运行SpringTest类，发现控制台可以输出People对象信息。

```
public class SpringTest {
    public static void main(String[] args) {
        ApplicationContext ac = new
ClassPathXmlApplicationContext("applicationContext.xml");
        People peo = ac.getBean("peo2", People.class);
        System.out.println(peo);
    }
}
```

2. 使用静态工厂实例化bean

2.1 创建工厂类

静态工厂和实例工厂类最主要的区别是，创建bean的方法是static修饰的。

创建静态工厂类：com.bjsxt.factory.PeopleStaticFactory

```java
package com.bjsxt.factory;

import com.bjsxt.pojo.People;

public class PeopleStaticFactory {
    private static People peo = new People();

    public static People newInstance(){
        return peo;
    }
}
```

2.2 配置配置文件

在applicationContext.xml中配置bean，注意bean标签中属性含义：

- class：静态工厂类的全限定路径
- factory-method：静态工厂中创建bean的静态方法。

之所以可以这样配置，是因为Java中静态方法是通过类名直接调用，不需要创建工厂的实例。

```xml
<bean id="peo3" class="com.bjsxt.factory.PeopleStaticFactory" factory-
method="newInstance"></bean>
```

2.3 在测试类中测试效果

运行SpringTest类，发现控制台可以输出People对象信息。

```java
package com.bjsxt.test;

import com.bjsxt.pojo.People;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class SpringTest {
    public static void main(String[] args) {
        ApplicationContext ac = new
ClassPathXmlApplicationContext("applicationContext.xml");
        People peo = ac.getBean("peo3", People.class);
        System.out.println(peo);
    }
}
```