【第32话：面试官问起Spring注入的几种方式时应该这样回答】

Hello小伙伴们，本节课给大家讲解一下，当面试官问起：“Spring中注入有几种方式”时，我们应该如何回答

按照官方文档说明：https://docs.spring.io/spring-framework/docs/5.3.24/reference/html/core.html#beans-dependencies

在Spring中给Bean属性赋值有两种方式：

- 构造注入（Constructor-based Dependency Injection）：通过构造方法给bean的属性赋值。所以要求bean的类中必须提供对应参数的构造方法。相当于以前创建对象时new People(1,"张三");
- 设值注入，又称setter注入（Setter-based Dependency Injection）：通过Bean的setter方法赋值。所以要求Bean中属性必须提供setter方法。相当于以前的:People peo = new People(); peo.setId(1); peo.setName("张三");

1. 构造注入

构造注入要求必须在Bean中提供有参构造方法（无参数最好也提供上，这个位置不用。但是其他位置如果没有通过构造注入时，默认是调用无参构造）。

1.1 检查Bean类中是否有有参构造方法

新建com.bjsxt.pojo.People提供有残构造方法，并且注意构造方法参数列表类型和参数顺序。

```
public class People {
    private int id;
    private String name;
    // 无参数构造
    public People() {
    }
    // 有参构造
    public People(int id, String name) {
        this.id = id;
        this.name = name;
    }
    // getter/setter
}
```

1.2 配置bean

在配置文件applicationContext.xml中可以通过`<bean>`的子标签`<constructor-arg>`设置构造方法中一个参数的值。

> 解释说明：
>
> constructor-arg 里面有5个属性，这5个属性分为2类。
>
> （1）用来确定给哪个属性进行赋值
>
> name：参数名称
>
> index：参数索引。从0开始算起。

> type：参数类型。8大基本数据类型可以直接写关键字。其他类型需要写类型的全限定路径。
>
> 这三个属性如果只需要用到一个就能精确的告诉Spring，要设置的构造方法参数是哪个可以使用一个。如果无法精确到某个构造方法参数，可以多个一起结合使用。
>
> （2）设置属性的值
>
> value：简单数据类型直接设置。Spring会自动进行类型转换。
>
> ref：需要引用另一个bean的id。也就是说这个参数是一个类类型，且这个类的对象也被Spring容器管理。

下面代码使用了type进行确认到底给构造方法哪个参数赋值。也可以尝试使用其他的属性来确认参数。

```
    <bean id="peo4" class="com.bjsxt.pojo.People">
        <constructor-arg type="int" value="1"></constructor-arg>
        <constructor-arg type="java.lang.String" value="张三"></constructor-arg>
    </bean>
```

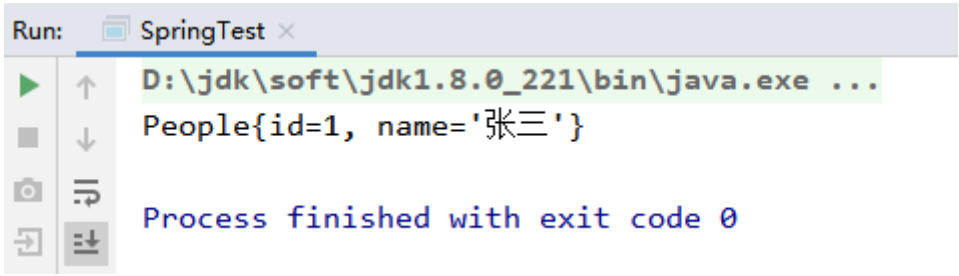
1.3 编写测试类并观察控制台运行结果

```
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("peo4", People.class);// 此处调用peo4
        System.out.println(peo);
    }
}
```

控制台输出结果包含了配置的值



2.设值注入（setter注入）

2.1 检查Bean类中是否有Setter方法

Setter注入一般都是结合无参构造方法一起使用。所以类中有无参构造方法。

```
package com.bjsxt.pojo;

public class People {
    private int id;
    private String name;
    public People() {
    }
    public People(int id, String name) {
        this.id = id;
        this.name = name;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    @Override
    public String toString() {
        return "People{" +
                "id=" + id +
                ", name='" + name + '\'' +
                '}';
    }
}
```

1.2 配置Bean

重新配置了一个bean标签，并设置id=“peo5”。

通过`<property>`标签调用类的setter方法。

- name：属性名称。
- value：属性值。
- ref：引用另一个bean标签的id属性。

```
    <bean id="peo5" class="com.bjsxt.pojo.People">
        <property name="id" value="2"></property>
        <property name="name" value="李四"></property>
    </bean>
```

1.3 编写测试类并观察控制台结果

在测试类中获取peo5的bean，并输出到控制台，可以看到结果是包含配置文件中配置的值