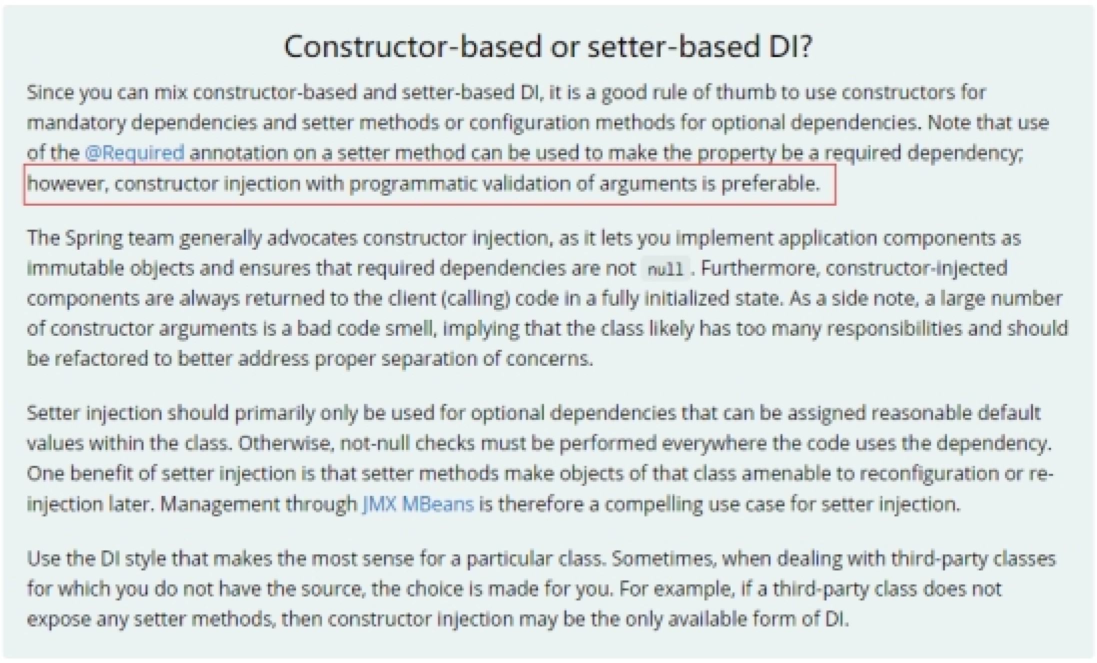
【第40话：Spring 循环注入问题，这事好办也不好办】

Hello 小伙伴们，这节课给大家讲一下常见面试题之“如何解决Spring 循环注入问题”。

Spring 循环注入问题在Spring 官网文档中明确给出了说明：循环注入即多个类相互依赖，产生了一个闭环。

网址：https://docs.spring.io/spring-framework/docs/5.3.24/reference/html/core.html#beans-dependencies



Constructor-based or setter-based DI?

Since you can mix constructor-based and setter-based DI, it is a good rule of thumb to use constructors for mandatory dependencies and setter methods or configuration methods for optional dependencies. Note that use of the @Required annotation on a setter method can be used to make the property be a required dependency; however, constructor injection with programmatic validation of arguments is preferable.

The Spring team generally advocates constructor injection, as it lets you implement application components as immutable objects and ensures that required dependencies are not `null`. Furthermore, constructor-injected components are always returned to the client (calling) code in a fully initialized state. As a side note, a large number of constructor arguments is a bad code smell, implying that the class likely has too many responsibilities and should be refactored to better address proper separation of concerns.

Setter injection should primarily only be used for optional dependencies that can be assigned reasonable default values within the class. Otherwise, not-null checks must be performed everywhere the code uses the dependency. One benefit of setter injection is that setter methods make objects of that class amenable to reconfiguration or re-injection later. Management through JMX MBeans is therefore a compelling use case for setter injection.

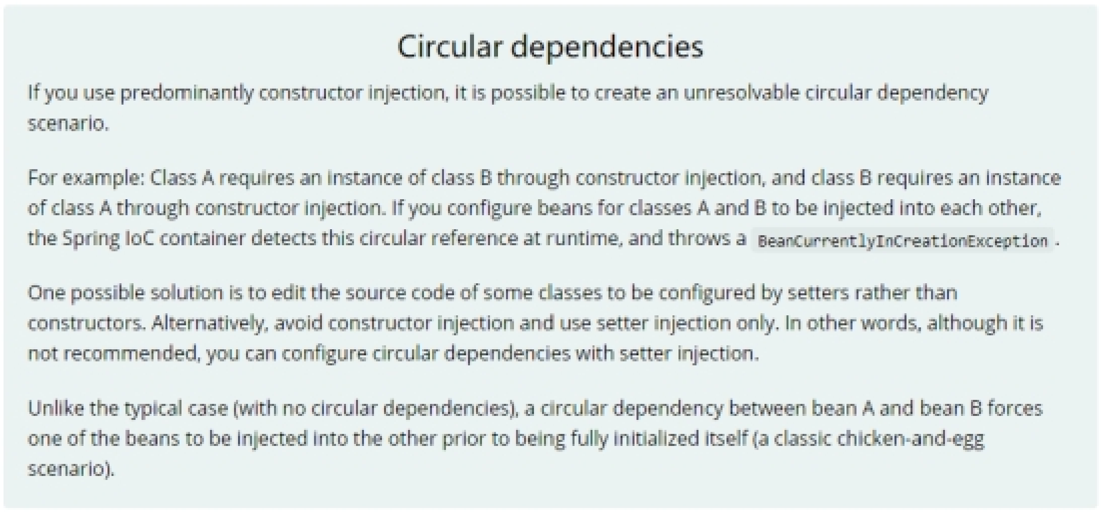
Use the DI style that makes the most sense for a particular class. Sometimes, when dealing with third-party classes for which you do not have the source, the choice is made for you. For example, if a third-party class does not expose any setter methods, then constructor injection may be the only available form of DI.

其实Spring循环注入问题并不是我们开发者去解决的，而是Spring本身会根据我们的代码进行解决。但是其中有的情况能解决，有的会直接报异常。汇总如下：

- 第一种：两个Bean都是用构造注入时是无法解决循环注入问题的。
- 第二种：如果Bean的scope属性为prototype时，使用设值注入也是无法解决循环注入问题的。
- 第三种：如果Bean的scope属性为默认值singleton时，使用设值注入Spring可以解决循环注入问题。

下面我们先看看Spring 官方中对构造注入时出现循环注入的解释。

当两个类都是用构造注入时，没有等当前类实例化完成就需要注入另一个类，而另一个类没有实例化完整还需要注入当前类，所以这种情况是无法解决循环注入问题的的。会出现BeanCurrentlyInCreationException异常。


Circular dependencies

If you use predominantly constructor injection, it is possible to create an unresolvable circular dependency scenario.

For example: Class A requires an instance of class B through constructor injection, and class B requires an instance of class A through constructor injection. If you configure beans for classes A and B to be injected into each other, the Spring IoC container detects this circular reference at runtime, and throws a `BeanCurrentlyInCreationException`.

One possible solution is to edit the source code of some classes to be configured by setters rather than constructors. Alternatively, avoid constructor injection and use setter injection only. In other words, although it is not recommended, you can configure circular dependencies with setter injection.

Unlike the typical case (with no circular dependencies), a circular dependency between bean A and bean B forces one of the beans to be injected into the other prior to being fully initialized itself (a classic chicken-and-egg scenario).

下面通过代码给小伙伴们演示一下构造注入时循环注入的效果。

在搭建好Spring环境的项目中新建两个类:

先新建Teacher类代表老师

```java
package com.bjsxt.pojo;

public class Teacher {
    private Student student;

    public Teacher(Student student) {
        this.student = student;
    }

    public Teacher() {
    }

    public Student getStudent() {
        return student;
    }

    public void setStudent(Student student) {
        this.student = student;
    }
}
```

然后在新建个Student类，代表学生

```java
package com.bjsxt.pojo;

public class Student {
    private Teacher teacher;

    public Student() {
    }

    public Student(Teacher teacher) {
        this.teacher = teacher;
    }

    public Teacher getTeacher() {
        return teacher;
    }

    public void setTeacher(Teacher teacher) {
        this.teacher = teacher;
    }
}
```

在Spring的配置文件applicationContext.xml中设置两个Bean的循环注入

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
        https://www.springframework.org/schema/beans/spring-beans.xsd"  >

    <bean id="teacher" class="com.bjsxt.pojo.Teacher">
        <constructor-arg name="student" ref="student"></constructor-arg>
    </bean>

    <bean id="student" class="com.bjsxt.pojo.Student">
        <constructor-arg name="teacher" ref="teacher"></constructor-arg>
    </bean>
</beans>
```

最后在测试类中编写测试代码

```java
package com.bjsxt.test;

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

public class Test {
    public static void main(String[] args) {
        ApplicationContext applicationContext = new
ClassPathXmlApplicationContext("applicationContext.xml");
        Teacher teacher = applicationContext.getBean("teacher", Teacher.class);
        System.out.println(teacher);
    }
}
```

运行测试类后会发现IDEA控制台出现异常。最后一个Cased by的异常类型是Caused by: org.springframework.beans.factory.BeanCreationException，代表着发生了循环注入问题。

```
D:\jdk\soft\jdk1.8.0_221\bin\java.exe -
javaagent:D:\idea\soft\lib\idea_rt.jar=51603:D:\idea\soft\bin -
Dfile.encoding=UTF-8 -classpath
D:\jdk\soft\jdk1.8.0_221\jre\lib\charsets.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\d
eploy.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\access-bridge-
64.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\cldrdata.jar;D:\jdk\soft\jdk1.8.0_22
1\jre\lib\ext\dnsns.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\jaccess.jar;D:\jdk\
soft\jdk1.8.0_221\jre\lib\ext\jfxrt.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\loc
aledata.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\nashorn.jar;D:\jdk\soft\jdk1.8.
0_221\jre\lib\ext\sunec.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\sunjce_provider
.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\sunmscapi.jar;D:\jdk\soft\jdk1.8.0_221
\jre\lib\ext\sunpkcs11.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\ext\zipfs.jar;D:\jdk
\soft\jdk1.8.0_221\jre\lib\javaws.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\jce.jar;D
:\jdk\soft\jdk1.8.0_221\jre\lib\jfr.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\jfxswt.
jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\jsse.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\m
anagement-
agent.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\plugin.jar;D:\jdk\soft\jdk1.8.0_221\j
re\lib\resources.jar;D:\jdk\soft\jdk1.8.0_221\jre\lib\rt.jar;D:\idea\ideaws\spri
ng1\target\classes;C:\Users\smallming\.m2\repository\org\springframework\spring-
context\5.3.16\spring-context-
5.3.16.jar;C:\Users\smallming\.m2\repository\org\springframework\spring-
aop\5.3.16\spring-aop-
5.3.16.jar;C:\Users\smallming\.m2\repository\org\springframework\spring-
beans\5.3.16\spring-beans-
5.3.16.jar;C:\Users\smallming\.m2\repository\org\springframework\spring-
core\5.3.16\spring-core-
5.3.16.jar;C:\Users\smallming\.m2\repository\org\springframework\spring-
jcl\5.3.16\spring-jcl-
5.3.16.jar;C:\Users\smallming\.m2\repository\org\springframework\spring-
expression\5.3.16\spring-expression-5.3.16.jar com.bjsxt.test.Test
五月 23, 2022 3:54:36 下午
org.springframework.context.support.AbstractApplicationContext refresh
警告: Exception encountered during context initialization - cancelling refresh
attempt: org.springframework.beans.factory.BeanCreationException: Error creating
bean with name 'teacher' defined in class path resource [applicationContext.xml]:
Cannot resolve reference to bean 'student' while setting constructor argument;
nested exception is org.springframework.beans.factory.BeanCreationException:
Error creating bean with name 'student' defined in class path resource
[applicationContext.xml]: Cannot resolve reference to bean 'teacher' while
setting constructor argument; nested exception is
org.springframework.beans.factory.BeanCurrentlyInCreationException: Error
creating bean with name 'teacher': Requested bean is currently in creation: Is
there an unresolvable circular reference?
Exception in thread "main"
org.springframework.beans.factory.BeanCreationException: Error creating bean with
name 'teacher' defined in class path resource [applicationContext.xml]: Cannot
resolve reference to bean 'student' while setting constructor argument; nested
exception is org.springframework.beans.factory.BeanCreationException: Error
creating bean with name 'student' defined in class path resource
[applicationContext.xml]: Cannot resolve reference to bean 'teacher' while
setting constructor argument; nested exception is
org.springframework.beans.factory.BeanCurrentlyInCreationException: Error
creating bean with name 'teacher': Requested bean is currently in creation: Is
there an unresolvable circular reference?
```

```
    at
org.springframework.beans.factory.support.BeanDefinitionValueResolver.resolveRef
erence(BeanDefinitionValueResolver.java:342)
    at
org.springframework.beans.factory.support.BeanDefinitionValueResolver.resolveVal
ueIfNecessary(BeanDefinitionValueResolver.java:113)
    at
org.springframework.beans.factory.support.ConstructorResolver.resolveConstructor
Arguments(ConstructorResolver.java:707)
    at
org.springframework.beans.factory.support.ConstructorResolver.autowireConstructo
r(ConstructorResolver.java:198)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.aut
owireConstructor(AbstractAutowireCapableBeanFactory.java:1372)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.cre
ateBeanInstance(AbstractAutowireCapableBeanFactory.java:1222)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.doC
reateBean(AbstractAutowireCapableBeanFactory.java:582)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.cre
ateBean(AbstractAutowireCapableBeanFactory.java:542)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.lambda$doGetBean$0
(AbstractBeanFactory.java:335)
    at
org.springframework.beans.factory.support.DefaultSingletonBeanRegistry.getSingle
ton(DefaultSingletonBeanRegistry.java:234)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.doGetBean(Abstract
BeanFactory.java:333)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.getBean(AbstractBe
anFactory.java:208)
    at
org.springframework.beans.factory.support.DefaultListableBeanFactory.preInstanti
ateSingletons(DefaultListableBeanFactory.java:953)
    at
org.springframework.context.support.AbstractApplicationContext.finishBeanFactory
Initialization(AbstractApplicationContext.java:918)
    at
org.springframework.context.support.AbstractApplicationContext.refresh(AbstractA
pplicationContext.java:583)
    at org.springframework.context.support.ClassPathXmlApplicationContext.<init>
(ClassPathXmlApplicationContext.java:144)
    at org.springframework.context.support.ClassPathXmlApplicationContext.<init>
(ClassPathXmlApplicationContext.java:85)
    at com.bjsxt.test.Test.main(Test.java:9)
```

```
Caused by: org.springframework.beans.factory.BeanCreationException: Error
creating bean with name 'student' defined in class path resource
[applicationContext.xml]: Cannot resolve reference to bean 'teacher' while
setting constructor argument; nested exception is
org.springframework.beans.factory.BeanCurrentlyInCreationException: Error
creating bean with name 'teacher': Requested bean is currently in creation: Is
there an unresolvable circular reference?
    at
org.springframework.beans.factory.support.BeanDefinitionValueResolver.resolveRef
erence(BeanDefinitionValueResolver.java:342)
    at
org.springframework.beans.factory.support.BeanDefinitionValueResolver.resolveVal
ueIfNecessary(BeanDefinitionValueResolver.java:113)
    at
org.springframework.beans.factory.support.ConstructorResolver.resolveConstructor
Arguments(ConstructorResolver.java:707)
    at
org.springframework.beans.factory.support.ConstructorResolver.autowireConstructo
r(ConstructorResolver.java:198)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.aut
owireConstructor(AbstractAutowireCapableBeanFactory.java:1372)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.cre
ateBeanInstance(AbstractAutowireCapableBeanFactory.java:1222)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.doC
reateBean(AbstractAutowireCapableBeanFactory.java:582)
    at
org.springframework.beans.factory.support.AbstractAutowireCapableBeanFactory.cre
ateBean(AbstractAutowireCapableBeanFactory.java:542)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.lambda$doGetBean$0
(AbstractBeanFactory.java:335)
    at
org.springframework.beans.factory.support.DefaultSingletonBeanRegistry.getSingle
ton(DefaultSingletonBeanRegistry.java:234)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.doGetBean(Abstract
BeanFactory.java:333)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.getBean(AbstractBe
anFactory.java:208)
    at
org.springframework.beans.factory.support.BeanDefinitionValueResolver.resolveRef
erence(BeanDefinitionValueResolver.java:330)
    ... 17 more
Caused by: org.springframework.beans.factory.BeanCurrentlyInCreationException:
Error creating bean with name 'teacher': Requested bean is currently in creation:
Is there an unresolvable circular reference?
    at
org.springframework.beans.factory.support.DefaultSingletonBeanRegistry.beforeSin
gletonCreation(DefaultSingletonBeanRegistry.java:355)
```

```
    at
org.springframework.beans.factory.support.DefaultSingletonBeanRegistry.getSingle
ton(DefaultSingletonBeanRegistry.java:227)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.doGetBean(Abstract
BeanFactory.java:333)
    at
org.springframework.beans.factory.support.AbstractBeanFactory.getBean(AbstractBe
anFactory.java:208)
    at
org.springframework.beans.factory.support.BeanDefinitionValueResolver.resolveRef
erence(BeanDefinitionValueResolver.java:330)
    ... 29 more

Process finished with exit code 1
```

第一种情况演示完成后，下面演示下第二种情况：如果Bean的scope属性为prototype时，循环注入的效果。

我们先把applicationContext.xml中的配置修改一下,注入的方式修改为设置注入，并设置Bean的scope="prototype"

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
        https://www.springframework.org/schema/beans/spring-beans.xsd"  >

    <bean id="teacher" class="com.bjsxt.pojo.Teacher" scope="prototype">
        <property name="student" ref="student"></property>
    </bean>

    <bean id="student" class="com.bjsxt.pojo.Student" scope="prototype">
        <property name="teacher" ref="teacher"></property>
    </bean>
</beans>
```

运行测试类，发现依然会产生循环注入问题。控制台还是出现了BeanCurrentlyInCreationException异常。

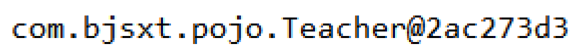
最后在来演示一下第三种情况：Bean的scope属性为默认值singleton时，循环注入的效果。

只需要修改配置文件applicationContext.xml中，把`<bean>`的scope属性删除掉就可以了。

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
        https://www.springframework.org/schema/beans/spring-beans.xsd"  >

    <bean id="teacher" class="com.bjsxt.pojo.Teacher">
        <property name="student" ref="student"></property>
    </bean>
    <bean id="student" class="com.bjsxt.pojo.Student">
        <property name="teacher" ref="teacher"></property>
    </bean>
</beans>
```

运行测试类会发现程序正确运行，控制台成功输出Teacher对象的toString()内容。



```
com.bjsxt.pojo.Teacher@2ac273d3
```

这种方式之所以可以成功运行是因为单例默认下有三级缓存(DefaultSingletonBeanRegistry)，可以暂时缓存没有被实例化完成的Bean。这样就不用考虑Bean实例化时先后问题，也就不会出现循环注入问题了。

通过这些演示后小伙伴们知道了只要Bean的scope="singleton"就不会出现循环注入问题。那么在平时我们进行代码编写时，尽量避开循环注入。如果实在无法避开，类中涉及到两个类的相互引用。例如：双向多对一、双向一对一的关系中就必须有双向引用。这时最好使用设值注入，并且scope设置为singleton。

Spring 循环注入问题回答示范

Spring 循环注入是因为多个类相互依赖，产生闭环了。

Spring 的循环注入问题是由Spring框架进行解决，开发者只能通过躲避无法解决的循环注入问题进行规避。

当获取的bean是通过setter注入，且scope为singleton时，其他bean可以是构造注入，也可以是setter注入，但是scope取值不能是singleton。因为Spring有缓存机制，可以对当前获取bean做临时缓存，先去对引用的bean进行实例化。

当bean的scope取值为prototype时是无法解决循环注入的。

当获取的bean是构造注入，也无法解决循环注入问题。