Spring01

今日学习目标：

***spring的IOC的容器***

***spring的IOC的入门案例（spring中容器是怎么创建对象的）***

***实例化Bean的三种方式***

***Bean的作用域配置：scope的配置***

***Bean的生命周期的配置***

***spring的依赖注入（DI）***

***spring的IOC的容器***

1：理解程序的耦合和解耦

（1）什么是程序的耦合

在A类中，new B就是耦合

* 案例一：使用反射技术创建JDBC连接对象



* 案例二：使用工厂模式+反射技术创建对象（使用工厂类读取xml的配置，底层使用反射完成解耦）

（2）解决程序耦合的思路一：反射

（3）解决程序耦合的思路二：工厂模式解耦

在resource下，创建applicationContext.xml



1.创建对象

​ 1.使用xml文件去描述对象属性的值，以及对象之间的关系(定义一个id作为对象标识)

​ 2.写一个工具类BeanFactory.java，读取xml文件(applicationContext.xml)的内容，解析并创建对象

​ 3.将对象存入到内存中（map<id,对象实例>），（表示对象创建,IOC）

​ 4.读取对象描述的属性，如果有对象关联就将对应的对象传入该对象（表示对象注入,DI）

2.对象的获取/使用

​ 1.在工具类里面创建静态getbean方法，通过id获取对象实例

​ 2.在外部调用工具类getbean方法，获取对象实例





2：控制反转IOC

（1）ioc概念，控制反转，将对象的创建交给spring去管理，我们需要的时候去spring 容器中取就可以了

（2）好处:实现了解耦 AccountDao dao = new AccountDao(); AccountService service = new AccountService (); 现在就是要什么直接去容器中取。

***spring的IOC的入门案例（spring中容器是怎么创建对象的）***

BeanFactory 才是Spring 容器中的顶层接口。

ApplicationContext 是它的子接口。

BeanFactory 和ApplicationContext 的区别：

创建的方式都表示单例对象。

创建对象的时间点不一样。

ApplicationContext：只要一读取配置文件，默认情况下就会创建对象。（立即加载）

BeanFactory：什么时候使用什么时候创建对象。（延迟加载）

ApplicationContext 接口的实现类

（1）ClassPathXmlApplicationContext： （重点）

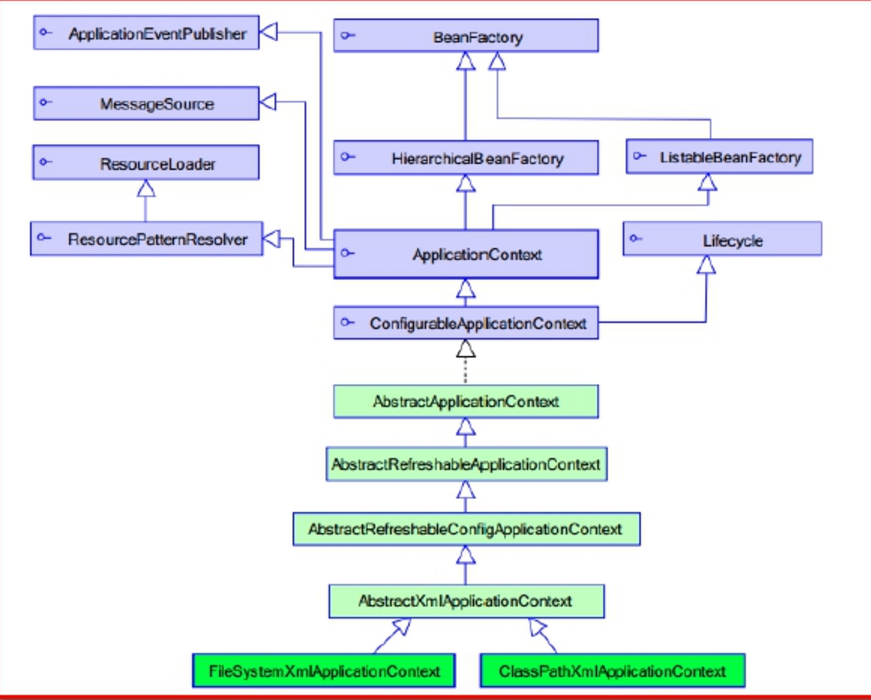
它是从类的根路径下加载配置文件 推荐使用这种

（2）FileSystemXmlApplicationContext：

它是从磁盘路径上加载配置文件，配置文件可以在磁盘的任意位置。

（3）AnnotationConfigApplicationContext: （第2天讲）

当我们使用注解配置容器对象时，需要使用此类来创建spring 容器。它用来读取注解。



***实例化Bean的三种方式***





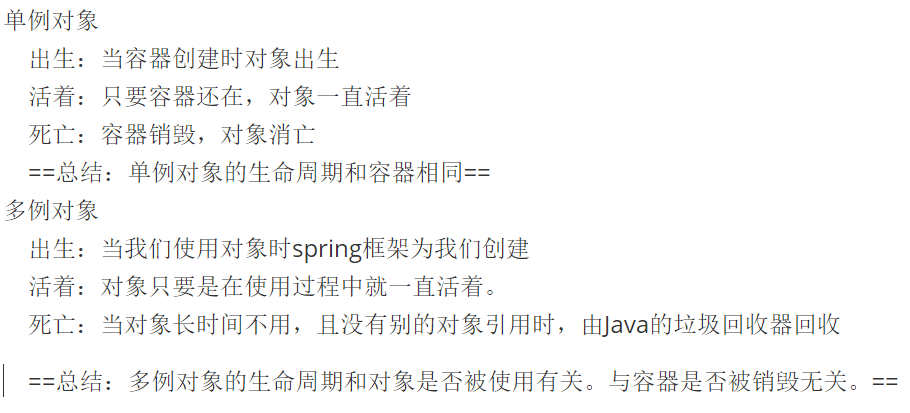




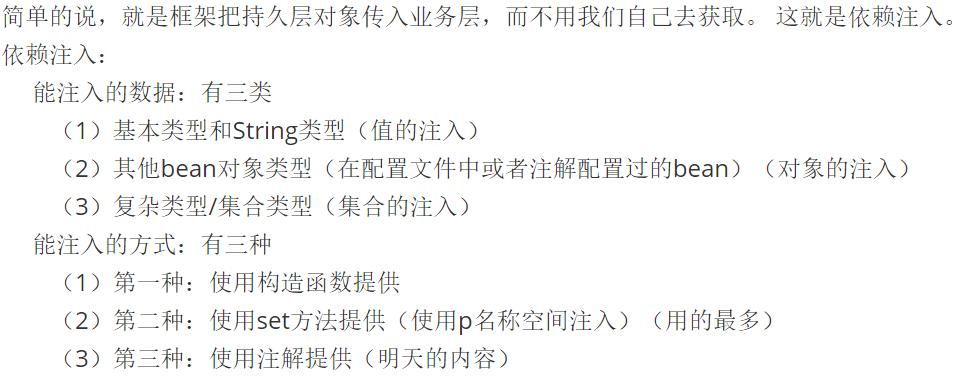
***Bean的作用域配置：scope的配置***



***Bean的生命周期的配置***



***spring的依赖注入（DI）***



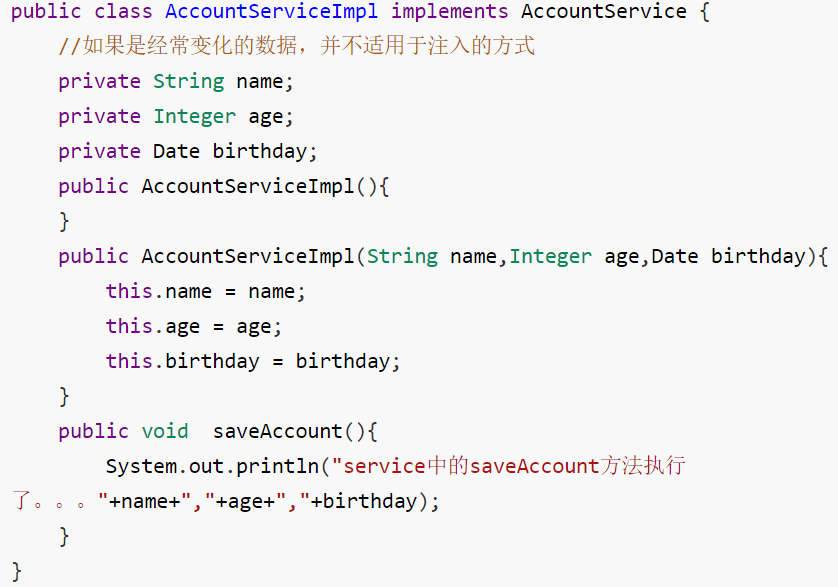
**第一种：构造函数注入 （了解）**

1：AccountServiceImpl.java提供传递参数的构造方法

2：applicationContext.xml（注入属性值）

3：测试类Client.java

第一步：AccountServiceImpl.java提供传递参数的构造方法。



第二步：applicationContext.xml



第三步：测试类Client.java



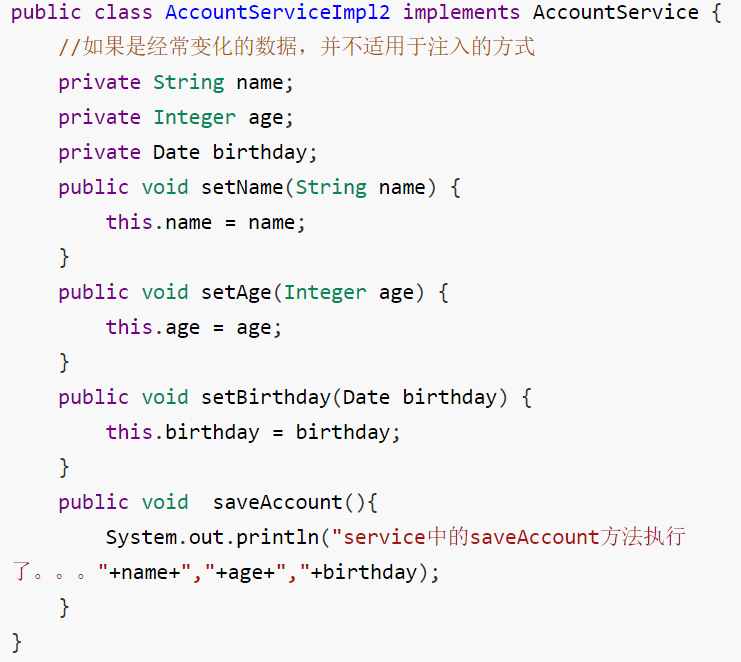
第二种：**set方法注入 （推荐使用）**

1：AccountServiceImpl2.java提供属性的set方法。

2：applicationContext.xml（注入属性值）

3：测试类Client.java

第一步：AccountServiceImpl2.java提供属性的set方法。



第二步：applicationContext.xml



***使用p名称空间注入数据（本质还是调用set方法） （了解）***



***注入集合属性（复杂类型）***



第二步：applicationContext.xml

<!-- 复杂类型的注入/集合类型的注入

用于给List结构集合注入的标签：

list array set

用于个Map结构集合注入的标签:

map props

结构相同，标签可以互换

-->

<bean id="accountService3" class="com.itheima.service.impl.AccountServiceImpl3">

<!--数组-->

<!--在spring的集合注入中，array，list，set是可以通用的-->

<property name="arrays">

<set>

<value>张三</value>

<value>22</value>

<ref bean="date"></ref>

</set>

</property>

<!--list集合-->

<property name="list">

<set>

<value>李四</value>

<value>20</value>

<ref bean="date"></ref>

</set>

</property>

<!--set集合-->

<property name="set">

<set>

<value>王五</value>

<value>25</value>

<ref bean="date"></ref>

</set>

</property>

<!--map集合-->

<property name="map">

<map>

<entry key="key001">

<value>赵六</value>

</entry>

<entry key="key002" value="23"></entry>

<entry key="key003">

<ref bean="date"></ref>

</entry>

</map>

</property>

<!--properties集合，和map集合很相似，也是键值对，键和值只能是String-->

<!--集合属性的应用场景：初始化系统中使用常量-->

<property name="properties">

<props>

<prop key="driver">com.mysql.jdbc.Driver</prop>

<prop key="url">jdbc:mysql:///itcastspring</prop>

<prop key="username">root</prop>

<prop key="password">root</prop>

</props>

</property>

</bean>

***在Service中，注入Dao***

