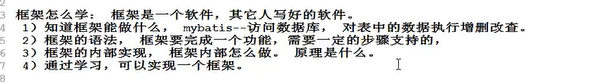
Spring全家桶:spring,springmvc, spring boot, spring clound

Spring:出现在2002年左右，解决企业开发的难度,减轻对项目模块之间的管理，

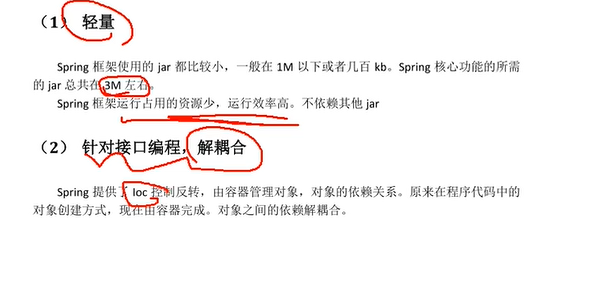
类和类之间的管理，帮助开发人员创建对象，管理对象之间的关系，

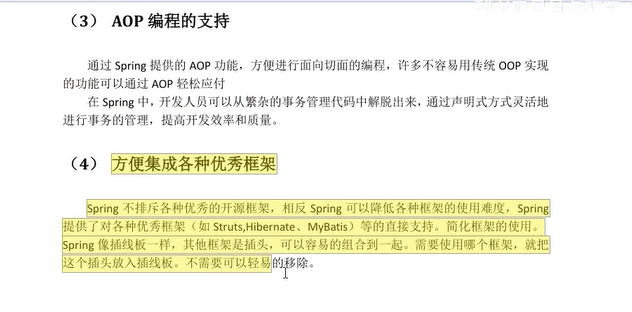
Spring核心技术ioc， aop，能实现模块之间，类之间的解耦合。

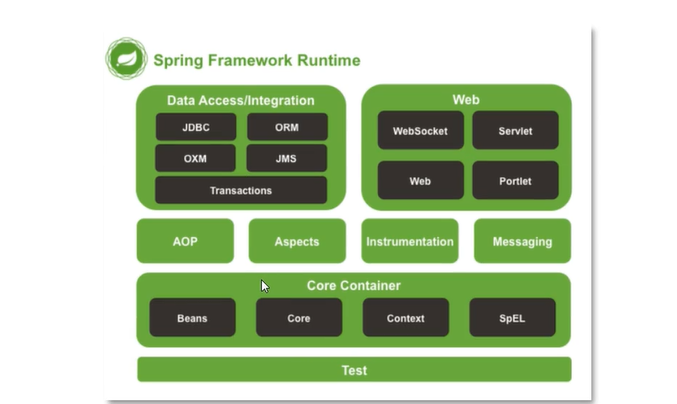
依赖:class a 中使用class b的属性或则方法，叫做class a 依赖 class b



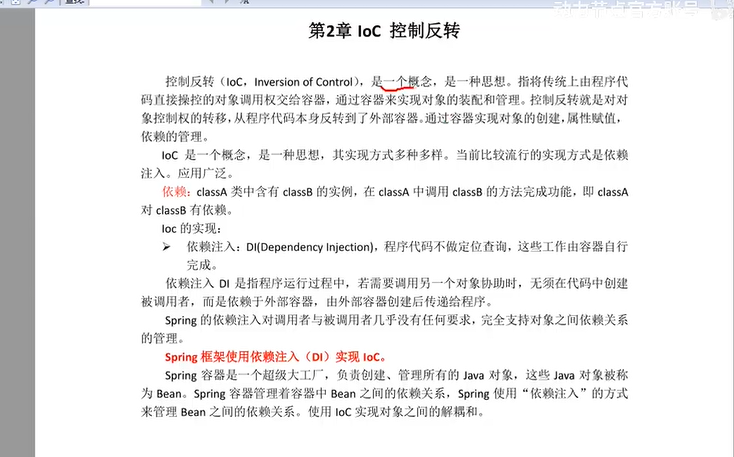
资料百度云地址https://pan.baidu.com/s/1uQV8miKR5LM3u2ZfD5Q51Q，提取码tlnb，有需要自取







Spring IOC



IOC：(Inversion of Control):控制反转，是一个理论，概念，思想。

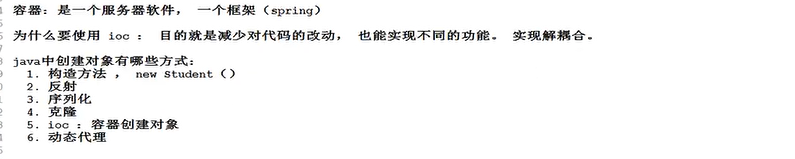
描述：把对象创建，赋值，管理工作都交给代码之外的容器实现，也就是对象的创建是有其它外部资源完成。

控制：创建对象，对象的属性赋值，对象之间的关系管理

反转:把原来的开发人员管理，创建对象的权限转移给代码之外的容器实现，由容器代替开发人员管理对象。创建对象，给对象赋值。

正转:有开发人员在代码中，使用new 构造方法创建对象，开发人员主动管理对象



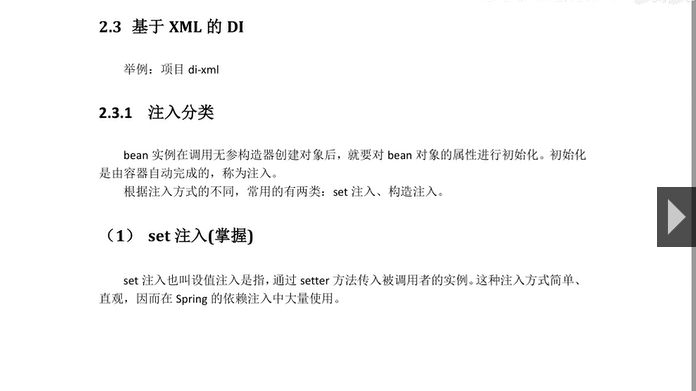


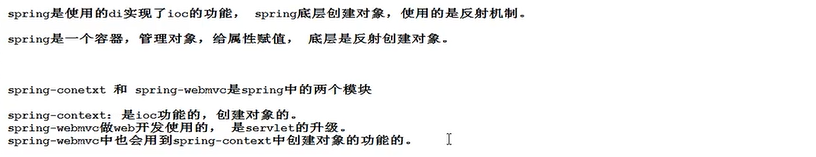


DI 是IOC的计数实现，

DI(Dependency Injection)：依赖注入,只需要在程序员中提供要是用的对象名称就可以，治愈对象如何在容器中创建，赋值，查找都由容器内部实现。

Spring是是用DI实现IOC的功能，spring 底层创建对象，是用反射机制。







<!--  
 声明student对象  
 注入:就是赋值的意思  
 di:给属性赋值  
 简单类型:spring中规定基础类型和字符串都是简单类型  
 1.set注入（设置注入）:spring调用类的set方法，你可以在set方法中完成属性赋值  
 1）简单类型的set注入  
 <bean id="xx" class="yyy">  
 <property name="属性的名字" value="此属性的值"/>  
 一个property只能给一个属性赋值  
 <property>  
 </bean>  
  
 2).引用类型的set注入:spring调用类的set方法  
 <bean id="xxx" class="yyy">  
 <property name="属性名称" ref="bean的id（对象的名称）">  
 <bean/>  
 2.构造注入：spring调用类有参构造方法，在构造对象的同时，在构造方法中给属性赋值  
 构造注入使用<constructor-arg>  
 <constructor-arg>标签：一个<constructor-arg>表示构造方法一个参数  
 <constructor-arg>标签属性：  
 name:表示构造方法的形参名  
 index:表示构造犯法的参数位置，参数从左到右位置是0 1 2 的顺序  
 value:构造方法的形参类型是简单类型的，使用value  
 ref:构造方法的形参类型是引用类型的使用ref  
  
 引用类型的自动注入：spring框架根据某些规则可以给引用类型赋值，不用你在给引用类型赋值了  
 使用规则常用的是byName，byType  
 1.byName(按名称注入):java类中引用类型的属性名和spring容器中（配置文件）<bean>的id名称一样  
 且数据类型是一致单位,这样的容器中的bean,spring能够赋值给引用类型。  
  
 2.byType(按类型注入):java类中引用类型的数据类型和spring 容器中（配置文件）<bean>的class属性  
 是同源关系，这样的bean能够赋值给引用类型。  
 同源就是一类的意思：  
 1.java类中引用类型的数据类型和bean的class的值是一样的  
 2.java类中引用类型的数据类型和bean的class是值父子关系的。  
 3.java类中引用类型的数据类型和bean的class是值接口和实现类关系的  
 语法：  
 <bean id="xx" class="yyyy" autowire="byType">  
 简单类型属性赋值  
 </bean>  
  
-->  
<!--使用name属性来构造-->  
<!-- <bean id="myStudent" class="com.AlbertChan.service.bao03.Student">-->  
<!-- <constructor-arg name="myname" value="陈俊雄"></constructor-arg>-->  
<!-- <constructor-arg name="myschool" ref="myschool"></constructor-arg>-->  
<!-- <constructor-arg name="myage" value="22"></constructor-arg>-->  
<!-- </bean>-->  
<!-- 使用index属性-->  
<!-- <bean id="myStudent2" class="com.AlbertChan.service.bao03.Student">-->  
<!-- <constructor-arg index="1" ref="school"></constructor-arg>-->  
<!-- <constructor-arg index="0" value="刘曼珍"></constructor-arg>-->  
  
<!-- <constructor-arg index="2" value="22"></constructor-arg>-->  
  
<!-- </bean>-->  
  
<!-- 使用引用类型的自动注入-->  
  
<!-- <bean id="myStudent04" class="com.AlbertChan.service.bao04.Student" autowire="byName">-->  
<!-- <property name="name" value="刘永超"></property>-->  
<!-- <property name="age" value="30"></property>-->  
<!-- </bean>-->  
 <bean id="myStudent05" class="com.AlbertChan.service.bao05.Student" autowire="byType">  
 <property name="name" value="陈俊超"></property>  
 <property name="age" value="30"></property>  
 </bean>  
<!-- <bean id="myschool" class="com.AlbertChan.service.bao03.School">-->  
<!-- <property name="name" value="东华理工大学"></property>-->  
<!-- <property name="address" value="南昌"></property>-->  
<!-- </bean>-->  
<!-- <bean id="school" class="com.AlbertChan.service.bao04.School">-->  
<!-- <property name="name" value="萍乡学院"></property>-->  
<!-- <property name="address" value="萍乡"></property>-->  
<!-- </bean>-->  
 <bean id="myschool" class="com.AlbertChan.service.bao05.School">  
 <property name="name" value="萍乡学院"></property>  
 <property name="address" value="萍乡"></property>  
 </bean>

<!-- 表示主配置文件  
 他一般不会用来定义对象  
 语法:  
 <import resource="其他配置文件的路径"/>  
 关键字:"classpath:"表示路径（class文件的目录）  
 在spring的配置文件中要指定其他文件的位置，需要使用calsspath,告诉spring到哪里去加载  
  
 在包含关系的配置文件中，可以通配符(\*:表示任意字符)  
 注意:主配置文件名称不能包含在通配符的范围内(不能叫做spring-tatol.xml)  
 -->  
  
 <import resource="classpath:bao06/spring-school.xml"></import>  
 <import resource="classpath:bao06/spring-student.xml"></import>

基于注解的di:通过注解完成java对象的创家，和属性的赋值

使用步骤:

1. 加入maven的依赖 spring-context，在你加入spring-context的同时，简接加入spring-aop的依赖。

使用注解必须使用spring-aop的依赖

1. 在类中加入spring的注解（多个不同功能的注解）
2. 在spring的配置文件中加入一个组件扫描器的标签，说明注解在你的相聚中的位置

学习的注解:

1. @component
2. @Respotory
3. @Service
4. @Controller
5. @Value
6. @Autowired
7. @Resource

/\*  
@Component:创建对象的，等同于<bean>的功能  
 属性:value 就是对象的名称，也就是bean的id值  
 value的值是唯一的,创建的对象在整个spring容器中就一个  
 位置:在类的上面  
  
 @Component（value = "myStudent")等同于  
 <bean id="myStudent" class="com.AlbertChan.bao01.Student">  
  
 spring中和@Component(value = "myStudent")功能一直，创建对象的注解还有：  
 1.@Repository(用在持久层类上面)：放在dao的实现类上面，  
 表示创建dao对象，dao 对象是能访问数据库的。  
 2.@Service（用在业务类上面）:放在service的实现类上面，  
 创建service对象，service对象是做业务处理，可以有事务等功能的。  
 3.@Controller(用在控制器上面的):放在控制器（处理器）类上面的，创建控制器对象的，  
 控制器对象，能够接受用户提交的参数，显示请求的处理结果  
 以上三个注解的使用语法和@Component一样的。都是创建对象，但是这三个注解还有额外的功能  
   
 \*/  
//使用value属性，指定对象名称  
//在不指定对象的名称，会默认用类的首字母小写  
@Component(value = "myStudent")

/\*  
@Value，简单类型的属性赋值  
 属性：value 是String类型的，表示简单类型的属性值  
 位置：1.属性定义上面，无序set方法，推荐使用  
 2.在set 方法上面  
 \*/  
@Value(value = "AlbertChan")  
private String name;  
@Value(value = "22")  
private Integer age;  
  
/\*  
引用类型  
@Autowired: spring 框架提供注解实现引用类型的赋值。  
spring中通过注解给引用类型赋值，使用的是自动注入原理，支持byName,byType  
@Autowired:默认使用的是byType自动注入。

属性：required，是一个boolean 类型的，默认true  
 required=true表示引用类型赋值失败，程序报错，并且终止执行。  
 required=false：引用类型如果赋值失败，程序正常执行，引用类型是null

位置：  
 1）在属性定义的上面，无序Set方法，推荐使用  
 2）在set方法上面  
  
如果使用byName自动注入的方式，需要做的是：  
1.在属性上面加@Autowired  
2.在属性上面加入@Qualifier（value="bean的id"）:表示使用指定名称的bean完成赋值  
  
 \*/  
@Autowired  
@Qualifier(value = "mySchool")//表示通过自动注入，在寻找mySchool对象进行赋值

假设@Resource只使用byName的方式，需要增加一个属性name  
name 的值是bean的id(名称)

//只使用byName的方式  
@Resource(name = "mySchool")  
private School school;

