Spring

是一个Aop，IoC的轻量级容器框架，为了简化Java开发，是我们专注于业务的开发。

分为这几个模块：Spring Context, Spring Core, Spring AoP, Spring MVC, Spring DAO, Spring ORM

优点，通过IoC，将对象的创建的对象之间的依赖关系，交给IoC框架来统一管理。使用这些JavaBean的时候通过DI，进行依赖注入，实现组件之间的解耦。

通过面向切面技术AOP，将一些通用任务：日志，事务，安全，权限，集中管理，也可以重复多次使用。

Spring就像是一个大杂烩，能够往里面加入其他不同的框架，比如说mybatis框架，之前的hibernate框架。前台的struts框架。等等

依赖注入的三种方式：通过setter方法将属性注入，通过构造方法将属性注入，通过注解的方法注入（注解的方法分为两类）

1. @Component 放在类上面，将这个类注册为javaBean对象。Id为首字母小写的类名。但是不推荐使用。通常使用的是@Repository （用于Dao层的JavaBean），@Service（用于service层），@Controller。等等。JavaBean的id都是首字母小写的类名。@Primary 如果一个类有多个JavaBean，那么标有@Primary的类首选。
2. @Resource 放到类的属性上面，通过ByName的方法进行配对，默认的id为首字母小写的属性类。一般使用的是@Autowried(requier == true)检查是否存在相应的Javabean。同时@Autowried是通过ByType的方法进行配置。可以通过加上注解@Qualifier（“id”），通过id来查找。

Spring中Javabean的作用域：scope属性（单例，原型，request，session）

Singleton: 单例模式，创建容器的时候就会初始化Javabean，并且也只有一个实例对象。（Java中默认的就是单例模式）

Prototype：原型模式，创建Java容器的时候，并不会初始化JavaBean，而是再获取Javabean的时候，实例化，每获取一次就实例化一次，每次的实例化对象都不一样。

用于Java web中的Request范围，用于会话的session范围。

Request：每一次新的Http请求都会产生一个新的bean。session：每一次http请求都会产生一个新的bean，该bean只在http中的Session中有效。

BeanFactory和ApplicationContext：都是实现了Ioc的基本功能：定义、加载、实例化、依赖管理。都可以作为spring的容器。但是ApplicationContext是Bean Factory的子类，拥有的功能比Bean Factory多。

ApplicationContext可以加载多个上下文，还可以实现对资源文件的访问。

Spring 容器的生命周期：实例化，设置对象属性，处理Aware接口，前置处理，后置处理。

关于AoP：

面向对象中的继承，是一种纵向关系，会导致大量代码重复，

AOP是面向切面的，是一种横向关系。将与业务无关的模块比如：日志，事务等，进行抽取，并封装为一个可重用的模块。这模块被被命名为切面，减少了系统中的重复代码，降低了模块间的耦合度。

AOP实现的关键是通过Java中的动态代理。被代理的类实现接口。核心就是InvocationHandler接口，和Proxy类。

切面（aspect）被抽取出来的公共模块。

连接点（join point)：程序运行过程中，所执行的方法。

切点（Pointcut): 定义对那些连接点进行拦截。

通知（advice）：在连接点（join point)上执行的操作。

目标对象（target）：包含连接点的对象叫做target

织入（wearing）：通过动态代理，在目标对象的方法（即join point）中执行（advice）的过程

通知的类型（相对于连接点）：前置通知，后置通知，环绕通知，抛出异常后的通知，返回后的通知。

返回通知是连接点正常返回时候执行。抛出异常后通知是异常退出的时候执行的通知。后置通知是正常一场退出都会执行。

Spring 中注解@Bean和@Component的区别；

@Component是用来注解类的，表明这个类是一个JavaBean。这个JavaBean要通过扫描来自动装配到spring容器中。

@Bean是和@Configuration配合使用。类用Configuration给注解，表明这是一个配置类。配置类里面的方法常才使用@Bean注解。表明方法的返回类型是一个Java Bean，不需要通过扫描。

用到的模式：

工厂模式

单例模式

代理模式：AOP

适配器模式

事务：逻辑上就是指一组操作，这些要么成功，要么失败（我给你转10块钱，我的账户少10块，你的账户多10块要）原子性，一致性，持久性，隔离性

面向对象的特性：抽象，封装，继承，多态。

Java的三大特性：封装，继承，多态。

事务的5种隔离级别：脏读，不可重复读，幻读

Default，

未提交读，

已提交读，

可重复读，

串行化读