1、spring原理

内部最核心的就是IOC了，动态注入，让一个对象的创建不用new了，可以自动的生产，这其实就是利用java里的反射，反射其实就是在运行时动态的去创建、调用对象，Spring就是在运行时，跟xml Spring的配置文件来动态的创建对象，和调用对象里的方法的 。

Spring还有一个核心就是AOP这个就是面向切面编程，可以为某一类对象 进行监督和控制（也就是 在调用这类对象的具体方法的前后去调用你指定的 模块）从而达到对一个模块扩充的功能。这些都是通过 配置类达到的。

Spring目的：就是让对象与对象（模块与模块）之间的关系没有通过代码来关联，都是通过配置类说明管理的（Spring根据这些配置 内部通过反射去动态的组装对象）

要记住：Spring是一个容器，凡是在容器里的对象才会有Spring所提供的这些服务和功能。

Spring里用的最经典的一个设计模式就是：模板方法模式。(这里我都不介绍了，是一个很常用的设计模式)， Spring里的配置是很多的，很难都记住，但是Spring里的精华也无非就是以上的两点，把以上两点跟理解了 也就基本上掌握了Spring.

Spring AOP与IOC

一、 IoC(Inversion of control): 控制反转

1、IoC：

概念：控制权由对象本身转向容器；由容器根据配置文件去创建实例并创建各个实例之间的依赖关系

核心：bean工厂；在Spring中，bean工厂创建的各个实例称作bean

二、AOP(Aspect-Oriented Programming): 面向方面编程

1、 代理的两种方式：

静态代理：

☐ 针对每个具体类分别编写代理类；

☐ 针对一个接口编写一个代理类；

动态代理：

针对一个方面编写一个InvocationHandler，然后借用JDK反射包中的Proxy类为各种接口动态生成相应的代理类

2、动态代理:

不用写代理类，虚拟机根据真实对象实现的接口产生一个类，通过类实例化一个动态代理，在实例化动态代理时将真实对象及装备注入到动态代理中，向客户端公开的是动态代理，当客户端调用动态代理方法时，动态代理根据类的反射得到真实对象的Method,调用装备的invoke方法，将动态代理、 Method、方法参数传与装备的invoke方法，invoke方法在唤起method方法前或后做一些处理。

1、产生动态代理的类:

java.lang.refect.Proxy

2、装备必须实现InvocationHandler接口实现invoke方法

3、反射

什么是类的返射?

通过类说明可以得到类的父类、实现的接口、内部类、构造函数、方法、属性并可以根据构造器实例化一个对象，唤起一个方法，取属性值，改属性值。如何得到一个类说明:

Class cls=类.class;

Class cls=对象.getClass();

Class.forName("类路径");

如何得到一个方法并唤起它?

Class cls=类.class;

Constructor cons=cls.getConstructor(new Class[]{String.class});

Object obj=cons.newInstance(new Object[]{"aaa"});

Method method=cls.getMethod("方法名",new Class[]{String.class,Integer.class});

method.invoke(obj,new Object[]{"aa",new Integer(1)});

4、spring的三种注入方式是什么?

setter

interface

constructor

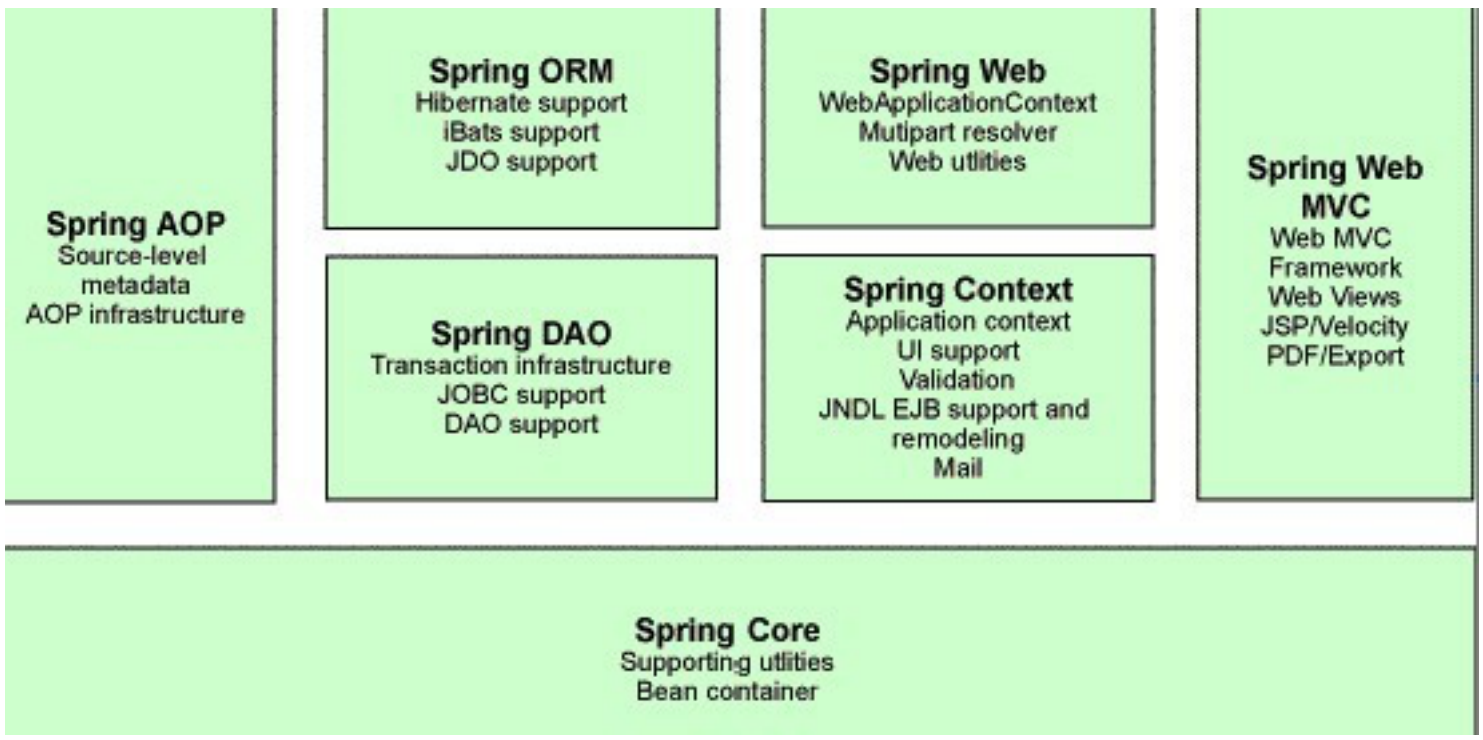
5、spring的核心接口及核类配置文件是什么?

FactoryBean:工厂bean主要实现ioc/di

ApplicationContext ac=new FileXmlApplicationContext("applicationContext.xml");

Object obj=ac.getBean("id值");

6、Spring框架的7个模块



Spring 框架是一个分层架构，由 7 个定义良好的模块组成。Spring 模块构建在核心容器之上，核心容器定义了创建、配置和管理 bean 的方式，组成 Spring 框架的每个模块（或组件）都可以单独存在，或者与其他一个或多个模块联合实现。每个模块的功能如下：

核心容器：核心容器提供 Spring 框架的基本功能。核心容器的主要组件

是 BeanFactory，它是工厂模式的实现。BeanFactory 使用控制反转 （IOC）模式将应用程序的配置和依赖性规范与实际的应用程序代码分开。

Spring 上下文：Spring 上下文是一个配置文件，向 Spring 框架提供上下文信息。Spring 上下文包括企业服务，例如 JNDI、EJB、电子邮件、国际化、校验和调度功能。

Spring AOP：通过配置管理特性，Spring AOP 模块直接将面向方面的编程功能集成到了 Spring 框架中。所以，可以很容易地使 Spring 框架管理的任何对象支持 AOP。Spring AOP 模块为基于 Spring 的应用程序中的对象提供了事务管理服务。通过使用 Spring AOP，不用依赖 EJB 组件，就可以将声明性事务管理集成到应用程序中。

Spring DAO：JDBC DAO 抽象层提供了有意义的异常层次结构，可用该结构来管理异常处理和不同数据库供应商抛出的错误消息。异常层次结构简化了错误处理，并且极大地降低了需要编写的异常代码数量（例如打开和关闭连接）。Spring DAO 的面向 JDBC 的异常遵从通用的 DAO 异常层次结构。

Spring ORM：Spring 框架插入了若干个 ORM 框架，从而提供了 ORM 的对象关系工具，其中包括 JDO、Hibernate 和 iBatis SQL Map。所有这些都遵从 Spring 的通用事务和 DAO 异常层次结构。

Spring Web 模块：Web 上下文模块建立在应用程序上下文模块之上，为基于 Web 的应用程序提供了上下文。所以，Spring 框架支持与 Jakarta Struts 的集成。Web 模块还简化了处理多部分请求以及将请求参数绑定到域对象的工作。

Spring MVC 框架：MVC 框架是一个全功能的构建 Web 应用程序的 MVC 实现。通过策略接口，MVC 框架变成为高度可配置的，MVC 容纳了大量视图技术，其中包括 JSP、Velocity、Tiles、iText 和 POI。

Spring 框架的功能可以用在任何 J2EE 服务器中，大多数功能也适用于不受管理的环境。Spring 的核心要点是：支持不绑定到特定 J2EE 服务的可重用业务和数据访问对象。毫无疑问，这样的对象可以在不同 J2EE 环境 （Web 或 EJB）、独立应用程序、测试环境之间重用。