**spring ioc**

**1、本章面试题**

           什么是spring,它有什么特点

什么是spring ioc,说说在你的项目的应用

**2、知识点**

**2.1、课程回顾**

      mybatis拦截器（插件）

Executor（接口）（SimpleExecutor,ReuesExecutor,BatchExecutor） 执行器 ,SimpleExecutor->StatementHandler(接口 SimpleStatementHandler,PreparedStatementHandler,CallableStatementHandler 语句执行对象)->ParameterHandler(接口 DefaultParameterHandler 参数处理接口)

->statement.execute(sql)(对数据库进行操作，更新，查询，执行存储过程或者函数)

->ResultSetHandler(接口 DefaultResultSetHandler 一般查询结果，使用ResultSetHandler 结果集处理接口)

分页原理：

StatementHandler mybatis只是对jdbc做了封装，底层执行语句时，一定要调用Statemete ,我们想在执行查询语句执行之前改变执行语句，从StatementHandler 中可以看出，prepare方法，他会返回Statemete ，如果想对执行语句进行改变，使用拦截器调用prepare，对执行行为作出改变。

**2.2、本章重点**

         spring 简介和特点

spring ioc 的概念

spring 3种注入方式

**3、具体内容**

**3.1 spring简介**

<https://spring.io/why-spring>

Spring 使java变得简单，现代化（自动化），高效，反应性（异步，无阻塞websocket ），为云（微服务springcloud）做准备。

Spring使每个人都可以更快，更轻松和更安全地进行Java编程。Spring对速度，简单性和生产率的关注使其成为世界上最受欢迎 Java框架。

我们使用了Spring框架随附的许多工具，并获得了许多现成的解决方案的好处，而不必担心编写大量额外的代码-这样确实节省了我们一些时间和精力。

**3.2 spring 优点：**

1.方便解耦，简化开发

通过Spring提供的IoC容器，我们可以将对象之间的依赖关系交由Spring进行控制，避免硬编码所造成的过度程序耦合。有了Spring，用户不必再为单实例模式类、属性文件解析等这些很底层的需求编写代码，可以更专注于上层的应用。

2.AOP编程的支持

通过Spring提供的AOP功能，方便进行面向切面的编程，许多不容易用传统OOP实现的功能可以通过AOP轻松应付。

3.声明式事务的支持

在Spring中，我们可以从单调烦闷的事务管理代码中解脱出来，通过声明式方式灵活地进行事务的管理，提高开发效率和质量。

4.方便程序的测试

可以用非容器依赖的编程方式进行几乎所有的测试工作，在Spring里，测试不再是昂贵的操作，而是随手可做的事情。例如：Spring对Junit4支持，可以通过注解方便的测试Spring程序。

5.方便集成各种优秀框架

Spring不排斥各种优秀的开源框架，相反，Spring可以降低各种框架的使用难度，Spring提供了对各种优秀框架（如Struts,Hibernate）等的直接支持。

6.降低Java EE API的使用难度

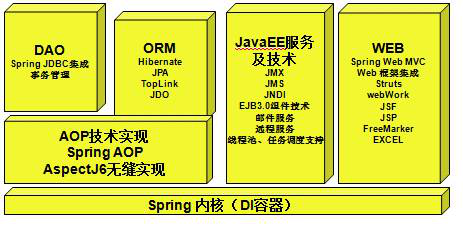
Spring对很多难用的Java EE API（如JDBC，JavaMail，远程调用等）提供了一个薄薄的封装层，通过Spring的简易封装，这些Java EE API的使用难度大为降低。

7.Java 源码是经典学习范例

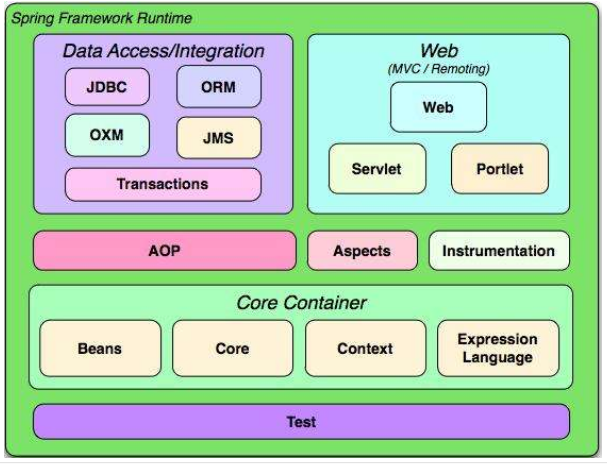
Spring的源码设计精妙、结构清晰、匠心独运，处处体现着大师对Java设计模式灵活运用以及对Java技术的高深造诣。Spring框架源码无疑是Java技术的最佳实践范例。如果想在短时间内迅速提高自己的Java技术水平和应用开发水平，学习和研究Spring源码将会使你收到意想不到的效果。

**3.2 spring 总架构图**

spring3.0及之前架构：



spring4.0后架构：



1、Core Container（Spring的核心）【重要】

2、AOP，Aspects（面向切面编程）【重要】

3、Messaging（消息发送的支持）

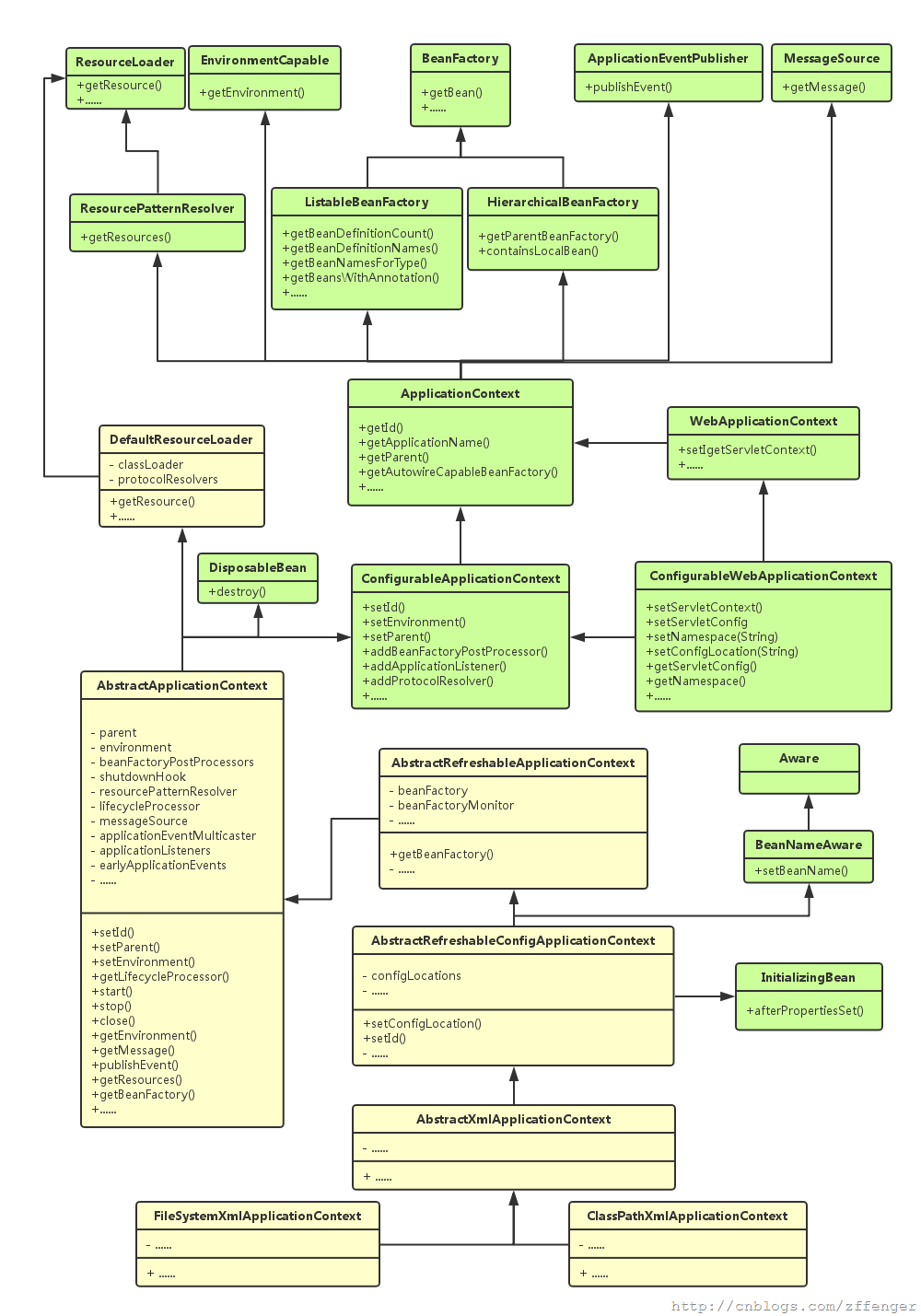
4、Data Access/Integration（数据访问和集成）

5、Web（主要是SpringWeb内容，包括MVC）【重要】

6、Test（Spring测试支持，包含JUint等测试单元的支持）

 7、Instrumentation（设备支持，比如Tomcat的支持）

**3.3 ApplicationContext和BeanFactory**



ApplicationContext应用上下文接口

实现类：

               ClassPathXmlApplicationContext：把上下文文件当成类路径资源

              FileSystemXmlApplicationContext：从文件系统中的XML文件载入上下文定义信息

BeanFactory是工厂模式（Factory pattern）的实现，负责创建和管理bean

BeanFactory接口包含以下基本方法

 containsBean(String name)

 getBean(String name)

 getBean(String name,Class class)

 getType(String name)

**3.4 什么是IOC**

<https://docs.spring.io/spring-framework/docs/current/spring-framework-reference/core.html#spring-core>

（1）IOC就是控制反转，是指创建对象的控制权的转移，以前创建对象的主动权和时机是由自己把控

的，而现在这种权力转移到Spring容器中，并由容器根据配置文件去创建实例和管理各个实例之间的依

赖关系，对象与对象之间松散耦合，也利于功能的复用。DI依赖注入，和控制反转是同一个概念的不同

角度的描述，即 应用程序在运行时依赖IoC容器来动态注入对象需要的外部资源。

（2）最直观的表达就是，IOC让对象的创建不用去new了，可以由spring自动生产，使用java的反射机

制，根据配置文件在运行时动态的去创建对象以及管理对象，并调用对象的方法的。

（3）Spring的IOC有三种注入方式 ：构造器注入、setter方法注入、根据注解注入。

IoC让相互协作的组件保持松散的耦合，而AOP编程允许你把遍布于应用各层的功能分离出来形成

可重用的功能组件。

**3.4 依赖注入方式**

创建项目，添加依赖：

常量配置：<spring.version>4.2.4.RELEASE</spring.version>

依赖jar:

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

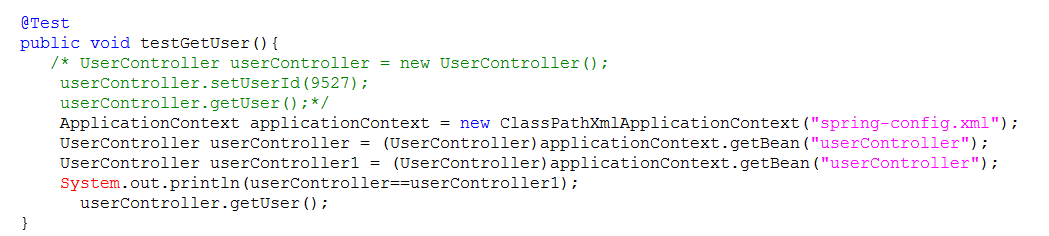
<artifactId>spring-beans</artifactId>

<version>${spring.version}</version>

</dependency>

3.4.1 设值注入





**知识点1：**

<bean> scope="prototype" 设置spring对bean管理是是否使用单例模式，默认是单例

**知识点2：**

自动装配

set注入和构造注入有时在做配置时比较麻烦。所以框架为了提高开发效率，提供自动装配功能，简化配置。Spring框架式默认不支持自动装配的，要想使用自动装配需要修改spring配置文件中<bean>标签的autowire属性

no

不支持自动装配功能

byName

从Spring环境中获取目标对象时，目标对象中的属性会根据名称在整个Spring环境中查找<bean>标签的id或者name属性值。如果有相同的，那么获取这个对象，实现关联。

byType

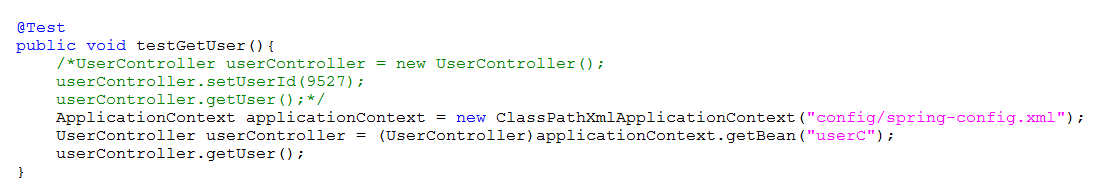
从Spring环境中获取目标对象时，目标对象中的属性会根据类型在整个spring环境中查找<bean>标签的class属性值。如果有相同的，那么获取这个对象，实现关联。

constructor

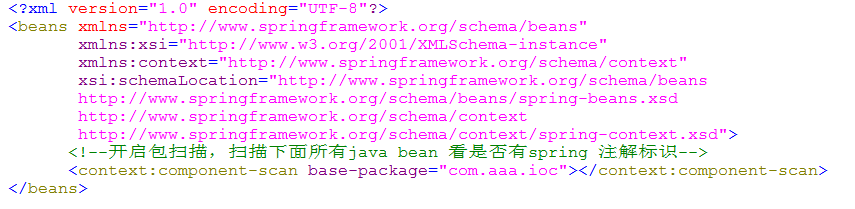
使用构造方法完成对象注入，其实也是根据构造方法的参数类型进行对象查找，相当于采用byType的方式。

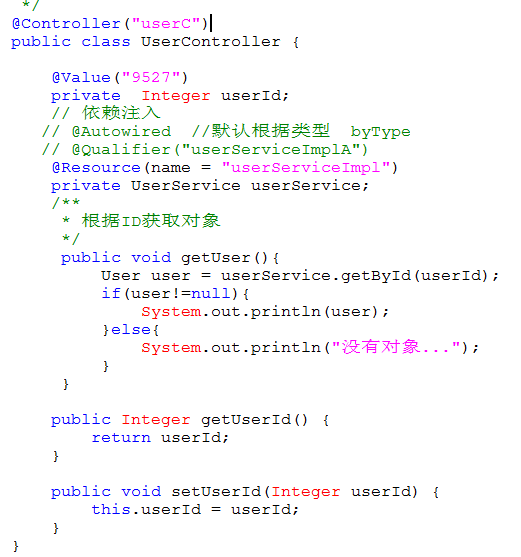
3.4.2 构造注入

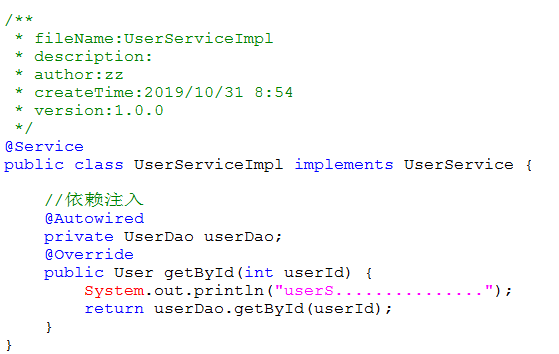




3.4.3 注解注入









@Component[分层不明确的情况下]

@Repository() [Dao层]

 @Service() [service层]

@Controller() [Action类，controller类]

@scope：指定bean作用的范围

@Autowired：指定自动装配依赖

@Qualifier：用@Autowired自动匹配时，如果有多个同类型对象则需要用该注解指定注入Bean的名称

@Resource 默认安照名称进行装配，名称可以通过name属性进行指定， 如果没有指定name属性，当注解写在字段上时，默认取字段名进行按照名称查找，如果注解写在setter方法上默认取属性名进行装配。 当找不到与名称匹配的bean时才按照类型进行装配。但是需要注意的是，如果name属性一旦指定，就只会按照名称进行装配。

**4、本章总结**

**4.1 总结本章知识点**

1， 设值注入(setter注入)

2，spring管理bean是否是单例（scope）

3，自动装配(autowire 1,byType 2,byName 3,no 4,constructor 5,default 自动选择)

4，构造注入

5，注解注入

**4.2 面试题答案**

**4.3 预习下一章重点**