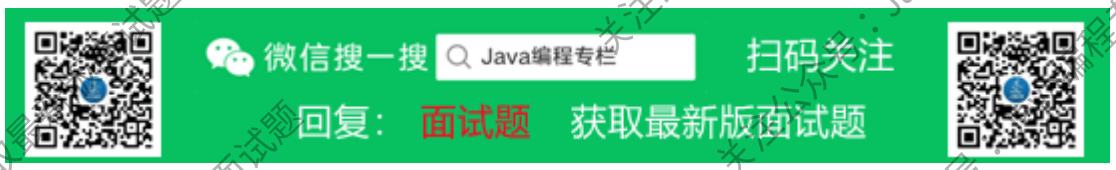


1.什么是 spring?

Spring 是个 java 企业级应用的开源开发框架。Spring 主要用来开发 Java 应用，但是有些扩展是针对构建 J2EE 平台的 web 应用。Spring 框架目标是简化 Java 企业级应用开发，并通过 POJO 为基础的编程模型促进良好的编程习惯。

2.为什么要使用 spring?

• spring 提供 ioc 技术，容器会帮你管理依赖的对象，从而不需要自己创建和管理依赖对象了，更轻松的实现了程序的解耦。

• spring 提供了事务支持，使得事务操作变的更加方便。

• spring 提供了面向切片编程，这样可以更方便的处理某一类的问题。

• 更方便的框架集成，spring 可以很方便的集成其他框架，比如 MyBatis、hibernate 等。

3.使用 Spring 框架的好处是什么？

轻量：Spring 是轻量的，基本的版本大约 2MB

控制反转：Spring 通过控制反转实现了松散耦合，对象们给出它们的依赖， 而不是创建或查找依赖的对象们

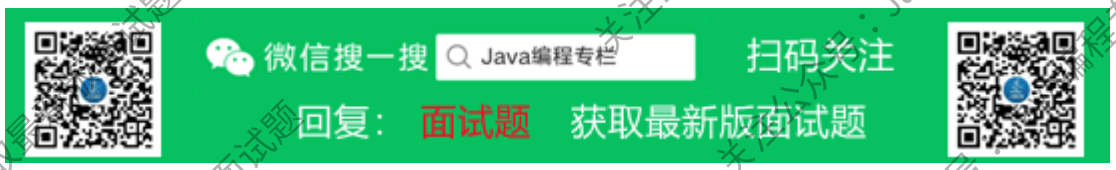
面向切面的编程(AOP)：Spring 支持面向切面的编程，并且把应用业务逻辑和系统服务分开

容器：Spring 包含并管理应用中对象的生命周期和配置

MVC 框架： Spring 的 WEB 框架是个精心设计的框架， 是 Web 框架的一个很好的替代品

事务管理：Spring 提供一个持续的事务管理接口，可以扩展到上至本地事务下至全局事务（ JTA）

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



异常处理： Spring 提供方便的 API 把具体技术相关的异常（ 比如由 JDBC ，Hibernate or JDO 抛出的） 转化为一致的 unchecked 异常

4. 解释一下什么是 aop？

aop 是面向切面编程，通过预编译方式和运行期动态代理实现程序功能的统一维护的一种技术。

简单来说就是统一处理某一“切面”（类）的问题的编程思想，比如统一处理日志、异常等。

5. 解释一下什么是 ioc？

ioc：Inversionof Control（中文：控制反转）是 spring 的核心，对于 spring 框架来说，就是由 spring 来负责控制对象的生命周期和对象间的关系。

简单来说，控制指的是当前对象对内部成员的控制权；控制反转指的是，这种控制权不由当前对象管理了，由其他（类,第三方容器）来管理。

6.IOC 的优点是什么？

IOC 或 依赖注入把应用的代码量降到最低。它使应用容易测试，单元测试不再需要单例和 JNDI 查找机制。最小的代价和最小的侵入性使松散耦合得以实现。IOC 容器支持加载服务时的饿汉式初始化和懒加载。

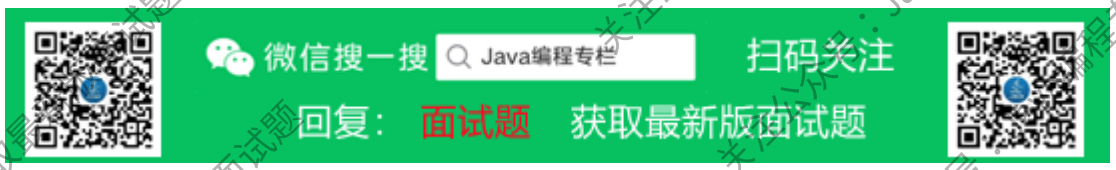
7. spring 有哪些主要模块？

• spring core：框架的最基础部分，提供 ioc 和依赖注入特性。

• spring context：构建于 core 封装包基础上的 context 封装包，提供了一种框架式的对象访问方法。

• spring dao：Data Access Object 提供了 JDBC 的抽象层。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



• spring aop：提供了面向切面的编程实现，让你可以自定义拦截器、切点等。

• spring Web：提供了针对 Web 开发的集成特性，例如文件上传，利用 servlet listeners 进行 ioc 容器初始化和针对 Web 的 ApplicationContext。

• spring Web mvc：spring 中的 mvc 封装包提供了 Web 应用的 Model-View-Controller（MVC）的实现。

8.BeanFactory – BeanFactory 实现举例

Bean 工厂是工厂模式的一个实现，提供了控制反转功能，用来把应用的配置和依赖从正真的应用代码中分离。

最常用的 BeanFactory 实现是 XmlBeanFactory 类。

9.XMLBeanFactory

最常用的就是 org.springframework.beans.factory.xml.XmlBeanFactory，它根据 XML 文件中的定义加载 beans。该容器从 XML 文件读取配置元数据并用它去创建一个完全配置的系统或应用。

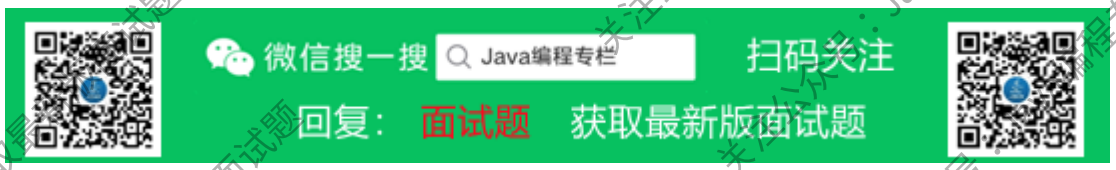
10.解释 AOP 模块

AOP 模块用于发给我们的 Spring 应用做面向切面的开发， 很多支持由 AOP 联盟提供， 这样就确保了 Spring 和其他 AOP 框架的共通性。这个模块将元数据编程引入 Spring。

11.解释 JDBC 抽象和 DAO 模块

通过使用 JDBC 抽象和 DAO 模块，保证数据库代码的简洁，并能避免数据库资源错误关闭导致的问题，它在各种不同的数据库的错误信息之上，提供了一个统一的异常访问层。它还利用 Spring 的 AOP 模块给 Spring 应用中的对象提供事务管理服务。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



12.什么是 Spring 的依赖注入？

依赖注入， 是 IOC 的一个方面， 是个通常的概念， 它有多种解释。这概念是说你不用创建对象，而只需要描述它如何被创建。你不在代码里直接组装你的组件和服务， 但是要在配置文件里描述哪些组件需要哪些服务，之后一个容器（ IOC 容器）负责把他们组装起来。

13.spring 常用的注入方式有哪些？

- setter 属性注入
- 构造方法注入
- 注解方式注入

14.有哪些不同类型的 IOC（依赖注入）方式？

构造器依赖注入：构造器依赖注入通过容器触发一个类的构造器来实现的，该类有一 系列参数，每个参数代表一个对其他类的依赖。

Setter 方法注入：Setter 方法注入是容器通过调用无参构造器或无参 static 工厂 方法实例化 bean 之后， 调用该 bean 的 setter 方法， 即实现了基于 setter 的依赖注入。

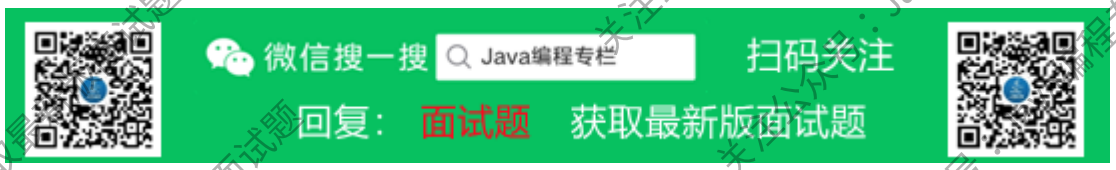
15.哪种依赖注入方式你建议使用，构造器注入，还是 Setter 方法注入？

你两种依赖方式都可以使用， 构造器注入和 Setter 方法注入。最好的解决方案是用构造器参数实现强制依赖，setter 方法实现可选依赖。

16.解释对象/关系映射集成模块

Spring 通过提供 ORM 模块， 支持我们在直接 JDBC 之上使用一个对象/关系映射映射

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



(ORM)工具，Spring 支持集成主流的 ORM 框架，如 Hiberate,JDO 和 iBATIS SQL Maps。Spring 的事务管理同样支持以上所有 ORM 框架及 JDBC。

17.解释 WEB 模块

Spring 的 WEB 模块是构建在 application context 模块基础之上， 提供一个适合 web 应用的上下文。这个模块也包括支持多种面向 web 的任务，如透明地处理多个文件上传请求和程序级请求参数的绑定到你的业务对象。它也有对 Jakarta Struts 的支持。

18.Spring 配置文件作用

Spring 配置文件是个 XML 文件， 这个文件包含了类信息， 描述了如何配置它们， 以及如何相互调用。

19.ApplicationContext 通常的实现是什么？

FileSystemXmlApplicationContext ： 此容器从一个 XML 文件中加载 beans 的定义，XML Bean 配置文件的全路径名必须提供给它的构造函数。

ClassPathXmlApplicationContext：此容器也从一个 XML 文件中加载 beans 的定义， 这里， 你需要正确设置 classpath 因为这个容器将在 classpath 里找 bean 配置。

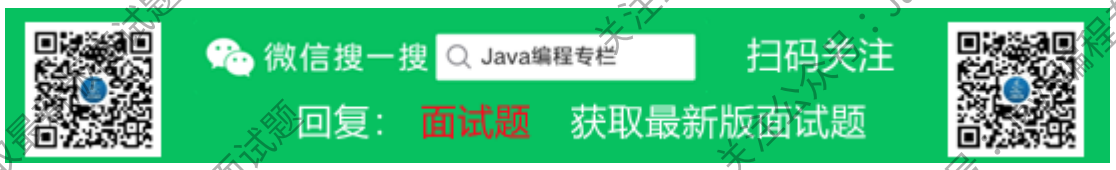
WebXmlApplicationContext：此容器加载一个 XML 文件，此文件定义了一个 WEB 应用的所有 bean。

20.Bean 工厂和 Application contexts 有什么区别？

Application contexts 提供一种方法处理文本消息，一个通常的做法是加载文件资源（比如镜像），它们可以向注册为监听器的 bean 发布事件。另外，在容器或容器内的对象上执行的那些不得不由 bean 工厂以程序化方式处理的操作， 可以在 Application contexts 中以声明的方式处理。

Application contexts 实现了 MessageSource 接口，该接口的实现以可插拔的方式提供获取本地化消息的方法。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



21.一个 Spring 的应用看起来象什么？

一个定义了一些功能的接口

这实现包括属性，它的 Setter ， getter 方法和函数等

Spring AOP

Spring 的 XML 配置文件

使用以上功能的客户端程序

22.什么是 Spring beans？

Spring beans 是那些形成 Spring 应用的主干的 java 对象。它们被 Spring IOC 容器初始化，装配，和管理。这些 beans 通过容器中配置的元数据创建。比如，以 XML 文件中 的形式定义。

Spring 框架定义的 beans 都是单件 beans。在 bean tag 中有个属性"singleton"，如果它被赋为 TRUE， bean 就是单件， 否则就是一个 prototype bean。默认是 TRUE，所以所有在 Spring 框架中的 beans 缺省都是单件。

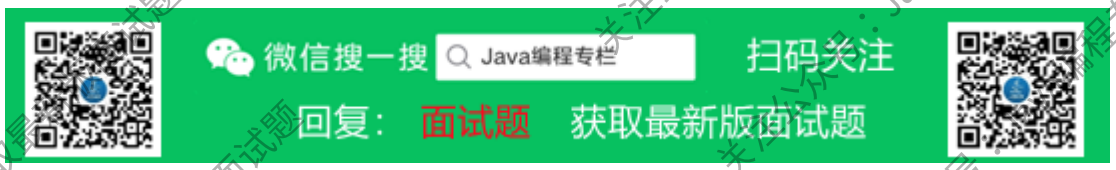
23.一个 Spring Bean 定义 包含什么？

一个 Spring Bean 的定义包含容器必知的所有配置元数据， 包括如何创建一个 bean，它的生命周期详情及它的依赖。

24. spring 中的 bean 是线程安全的吗？

spring 中的 bean 默认是单例模式，spring 框架并没有对单例 bean 进行多线程的封装处理。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



实际上大部分时候 spring bean 无状态的（比如 dao 类），所有某种程度上来说 bean 也是安全的，但如果 bean 有状态的话（比如 view model 对象），那就要开发者自己去保证线程安全了，最简单的就是改变 bean 的作用域，把“singleton”变更为“prototype”，这样请求 bean 相当于 new Bean()了，所以就可以保证线程安全了。

- 有状态就是有数据存储功能。
- 无状态就是不会保存数据。

25. spring 支持几种 bean 的作用域？

spring 支持 5 种作用域，如下：

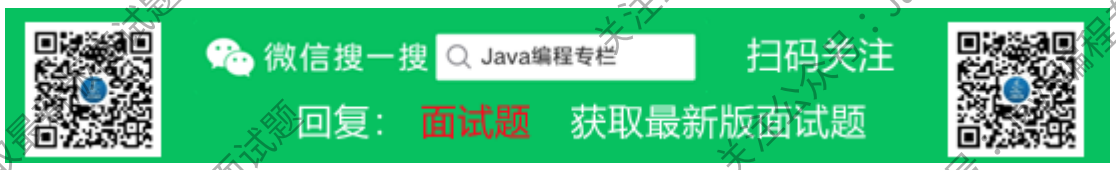
- singleton：spring ioc 容器中只存在一个 bean 实例，bean 以单例模式存在，是系统默认值；
- prototype：每次从容器调用 bean 时都会创建一个新的示例，既每次 getBean()相当于执行 new Bean()操作；
- Web 环境下的作用域：
- request：每次 http 请求都会创建一个 bean；
- session：同一个 http session 共享一个 bean 实例；
- global-session：用于 portlet 容器，因为每个 portlet 有单独的 session，globalsession 提供一个全局性的 http session。

注意： 使用 prototype 作用域需要慎重的思考，因为频繁创建和销毁 bean 会带来很大的性能开销。

26. spring 自动装配 bean 有哪些方式？

- no：默认值，表示没有自动装配，应使用显式 bean 引用进行装配。
- byName：它根据 bean 的名称注入对象依赖项。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



• byType：它根据类型注入对象依赖项。

• 构造函数：通过构造函数来注入依赖项，需要设置大量的参数。

• autodetect：容器首先通过构造函数使用 autowire 装配，如果不能，则通过 byType 自动装配。

27. spring 事务实现方式有哪些？

• 声明式事务：声明式事务也有两种实现方式，基于 xml 配置文件的方式和注解方式（在类上添加 @Transaction 注解）。

• 编码方式：提供编码的形式管理和维护事务。

28. 说一下 spring 的事务隔离？

spring 有五大隔离级别，默认值为 ISOLATION_DEFAULT（使用数据库的设置），其他四个隔离级别和数据库的隔离级别一致：

ISOLATION_DEFAULT：用底层数据库的设置隔离级别，数据库设置的是什么我就用什么；

ISOLATIONREADUNCOMMITTED：未提交读，最低隔离级别、事务未提交前，就可被其他事务读取（会出现幻读、脏读、不可重复读）；

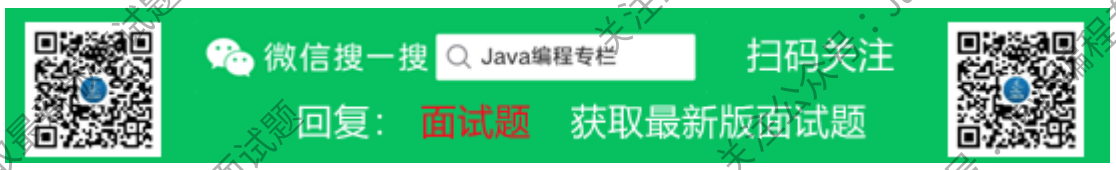
ISOLATIONREADCOMMITTED：提交读，一个事务提交后才能被其他事务读取到（会造成幻读、不可重复读），SQL server 的默认级别；

ISOLATIONREPEATABLEREAD：可重复读，保证多次读取同一个数据时，其值都和事务开始时候的内容是一致，禁止读取到别的事务未提交的数据（会造成幻读），MySQL 的默认级别；

ISOLATION_SERIALIZABLE：序列化，代价最高最可靠的隔离级别，该隔离级别能防止脏读、不可重复读、幻读。

脏读 ：表示一个事务能够读取另一个事务中还未提交的数据。比如，某个事务尝试插入记录 A，此时该事务还未提交，然后另一个事务尝试读取到了记录 A。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



不可重复读：是指在一个事务内，多次读同一数据。

幻读：指同一个事务内多次查询返回的结果集不一样。比如同一个事务 A 第一次查询时候有 n 条记录，但是第二次同等条件下查询却有 n+1 条记录，这就好像产生了幻觉。发生幻读的原因也是另外一个事务新增或者删除或者修改了第一个事务结果集里面的数据，同一个记录的数据内容被修改了，所有数据行的记录就变多或者变少了。

29.什么是基于 Java 的 Spring 注解配置? 给一些注解的例子

基于 Java 的配置，允许你在少量的 Java 注解的帮助下，进行你的大部分 Spring 配置而非通过 XML 文件。

以@Configuration 注解为例，它用来标记类可以当做一个 bean 的定义，被 Spring IOC 容器使用。另一个例子是@Bean 注解，它表示此方法将要返回一个对象，作为一个 bean 注册进 Spring 应用上下文。

30.什么是基于注解的容器配置？

相对于 XML 文件，注解型的配置依赖于通过字节码元数据装配组件，而非尖括号的声明。

开发者通过在相应的类，方法或属性上使用注解的方式，直接组件类中进行配置，而 不是使用 xml 表述 bean 的装配关系。

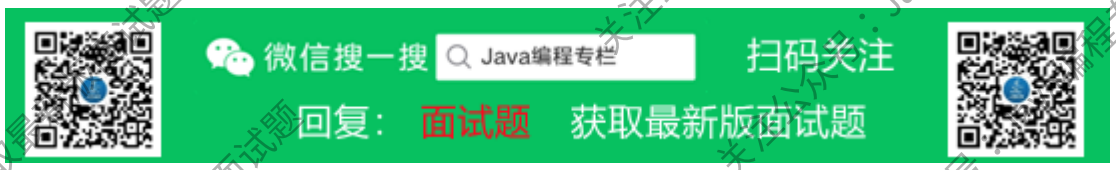
31.怎样开启注解装配？

注解装配在默认情况下是不开启的，为了使用注解装配，我们必须在 Spring 配置文件中配置 context:annotation-config/元素。

32.@Required 注解

这个注解表明 bean 的属性必须在配置的时候设置，通过一个 bean 定义的显式的属性值或通过自动装配， 若@Required 注解的 bean 属性未被设置， 容器将抛出 BeanInitializationException。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



33.@Autowired 注解

@Autowired 注解提供了更细粒度的控制，包括在何处以及如何完成自动装配。它的用法和@Required 一样，修饰 setter 方法、构造器、属性或者具有任意名称和/ 或多个参数的 PN 方法。

34.@Qualifier 注解

当有多个相同类型的 bean 却只有一个需要自动装配时，将@Qualifier 注解和@Autowire 注解结合使用以消除这种混淆，指定需要装配的确切的 bean。

35.在 Spring 框架中如何更有效地使用 JDBC？

使用 SpringJDBC 框架，资源管理和错误处理的代价都会被减轻。所以开发者只需写 statements 和 queries 从数据存取数据，JDBC 也可以在 Spring 框架提供的模板类的帮助下更有效地被使用，这个模板叫 JdbcTemplate。

36.JdbcTemplate

JdbcTemplate 类提供了很多便利的方法解决诸如把数据库数据转变成基本数据类型或对象，执行写好的或可调用的数据库操作语句，提供自定义的数据错误处理。

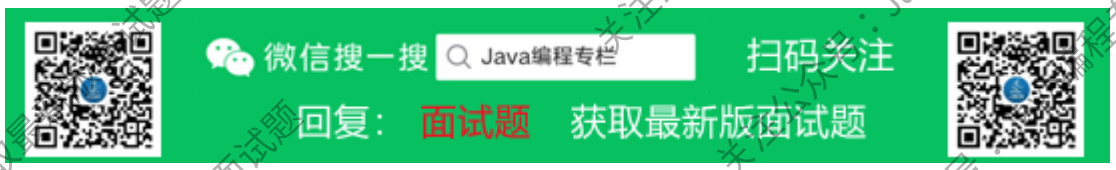
37.Spring 对 DAO 的支持

Spring 对数据访问对象（DAO）的支持旨在简化它和数据访问技术如 JDBC，Hibernate or JDO 结合使用。这使我们可以方便切换持久层。编码时也不用担心会捕获每种技术特有的异常。

38.使用 Spring 通过什么方式访问 Hibernate？

在 Spring 中有两种方式访问 Hibernate：

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



控制反转 Hibernate Template 和 Callback

继承 HibernateDAOSupport 提供一个 AOP 拦截器

39.Spring 支持的 ORM

Spring 支持以下 ORM：

Hibernate iBatis

JPA (Java Persistence API) TopLink

JDO (Java Data Objects) OJB

40.如何通过 HibernateDaoSupport 将 Spring 和 Hibernate 结合起来？

用 Spring 的 SessionFactory 调用 LocalSessionFactory。集成过程分三步： 配置 the Hibernate SessionFactory

继承 HibernateDaoSupport 实现一个 DAO 在 AOP 支持的事务中装配

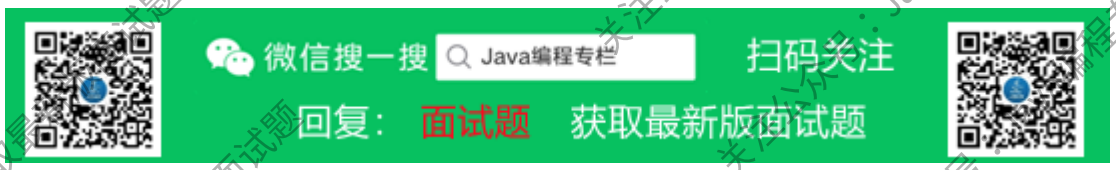
41.Spring 支持的事务管理类型

Spring 支持两种类型的事务管理： 编程式事务管理：这意味你通过编程的方式管理事务，给你带来极大的灵活性，但是 难维护。

声明式事务管理：这意味着你可以将业务代码和事务管理分离，你只需用注解和 XML 配置来管理事务。

42.Spring 框架的事务管理有哪些优点？

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



它为不同的事务 API 如 JTA， JDBC， Hibernate， JPA 和 JDO， 提供一个不变的编程模式。

它为编程式事务管理提供了一套简单的 API 而不是一些复杂的事务 API 如它支持声明式事务管理。

它和 Spring 各种数据访问抽象层很好得集成。

43.你更倾向用那种事务管理类型？

大多数 Spring 框架的用户选择声明式事务管理，因为它对应用代码的影响最小，因此更符合一个无侵入的轻量级容器的思想。声明式事务管理要优于编程式事务管理， 虽然比编程式事务管理（这种方式允许你通过代码控制事务）少了一点灵活性。

44.什么是 Spring 的 MVC 框架？

Spring 配备构建 Web 应用的全功能 MVC 框架。Spring 可以很便捷地和其他 MVC 框架集成，如 Struts，Spring 的 MVC 框架用控制反转把业务对象和控制逻辑清晰地隔离。它也允许以声明的方式把请求参数和业务对象绑定。

45. 说一下 spring mvc 运行流程？

• spring mvc 先将请求发送给 DispatcherServlet。

• DispatcherServlet 查询一个或多个 HandlerMapping，找到处理请求的 Controller。

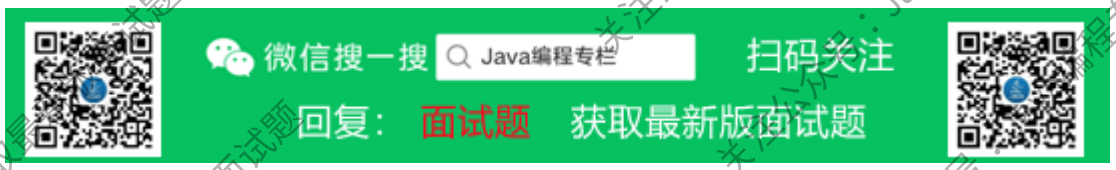
• DispatcherServlet 再把请求提交到对应的 Controller。

• Controller 进行业务逻辑处理后，会返回一个 ModelAndView。

• Dispathcher 查询一个或多个 ViewResolver 视图解析器，找到 ModelAndView 对象指定的视图对象。

• 视图对象负责渲染返回给客户端。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



46. spring mvc 有哪些组件？

- 前置控制器 DispatcherServlet。
- 映射控制器 HandlerMapping。
- 处理器 Controller。
- 模型和视图 ModelAndView。
- 视图解析器 ViewResolver。

47. @RequestMapping 的作用是什么？

将 http 请求映射到相应的类/方法上。

48. @Autowired 的作用是什么？

@Autowired 它可以对类成员变量、方法及构造函数进行标注，完成自动装配的工作，通过@Autowired 的使用来消除 set/get 方法。

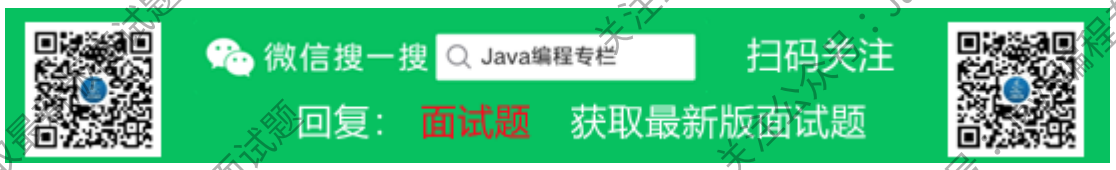
49.DispatcherServlet

Spring 的 MVC 框架是围绕 DispatcherServlet 来设计的，它用来处理所有的 HTTP 请求和响应。

50.WebApplicationContext

WebApplicationContext 继承了 ApplicationContext 并增加了一些 WEB 应用必备的特有功能，它不同于一般的 ApplicationContext ，因为它能处理主题，并找到被关联的 servlet。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



51.什么是 Spring MVC 框架的控制器？

控制器提供一个访问应用程序的行为，此行为通常通过服务接口实现。控制器解析用 户输入并将其转换为一个由视图呈现给用户的模型。Spring 用一个非常抽象的方式实现了一个控制层，允许用户创建多种用途的控制器。

52.@Controller 注解

该注解表明该类扮演控制器的角色，Spring 不需要你继承任何其他控制器基类或引用 Servlet API。

53.@RequestMapping 注解

该注解是用来映射一个 URL 到一个类或一个特定的方处理法上。

54.Aspect 切面

AOP 核心就是切面， 它将多个类的通用行为封装成可重用的模块， 该模块含有一组 API 提供横切功能。比如， 一个日志模块可以被称作日志的 AOP 切面。根据需求的不同，一个应用程序可以有若干切面。在 Spring AOP 中， 切面通过带有@Aspect 注解的类实现。

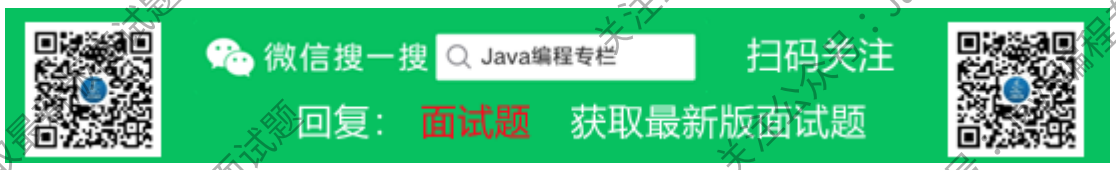
55. 在 Spring AOP 中，关注点和横切关注的区别是什么？

关注点是应用中一个模块的行为，一个关注点可能会被定义成一个我们想实现的一个 功能。 横切关注点是一个关注点，此关注点是整个应用都会使用的功能，并影响整个应用，比如日志，安全和数据传输，几乎应用的每个模块都需要的功能。因此这些都属于横切关注点。

56.连接点

连接点代表一个应用程序的某个位置，在这个位置我们可以插入一个 AOP 切面， 它实际

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



上是个应用程序执行 Spring AOP 的位置。

57.通知

通知是个在方法执行前或执行后要做的动作， 实际上是程序 执行时要通过 SpringAOP 框架触发的代码段。

Spring 切面可以应用五种类型的通知：

before：前置通知，在一个方法执行前被调用

after：在方法执行之后调用的通知，无论方法执行是否成功

after-returning：仅当方法成功完成后执行的通知

after-throwing：在方法抛出异常退出时执行的通知

around：在方法执行之前和之后调用的通知

58.切点

切入点是一个或一组连接点，通知将在这些位置执行。可以通过表达式或匹配的方式 指明切入点。

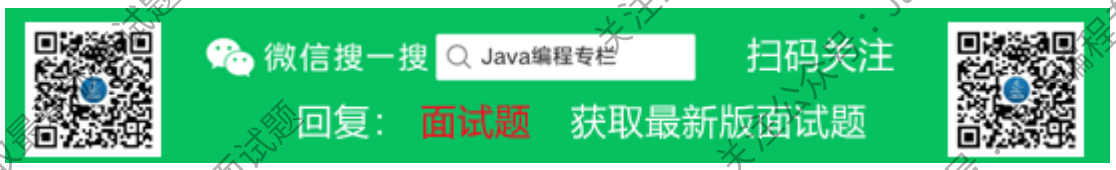
59.什么是引入？

引入允许我们在已存在的类中增加新的方法和属性。

60.什么是目标对象？

被一个或者多个切面所通知的对象。它通常是一个代理对象。也指被通知（advised）对象。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料



61.什么是代理？

代理是通知目标对象后创建的对象。从客户端的角度看，代理对象和目标对象是一样的。

62.有几种不同类型的自动代理？

BeanNameAutoProxyCreator

DefaultAdvisorAutoProxyCreator

Metadata autoproxying

63.什么是织入。什么是织入应用的不同点？

织入是将切面和到其他应用类型或对象连接或创建一个被通知对象的过程。

织入可以在编译时，加载时，或运行时完成。

64.解释基于 XML Schema 方式的切面实现

在这种情况下，切面由常规类以及基于 XML 的配置实现。

65.解释基于注解的切面实现

在这种情况下(基于@AspectJ 的实现)，涉及到的切面声明的风格与带有 java5 标注的普通 java 类一致。

关注公众号：Java 编程专栏，获取最新面试题，架构师资料