**①Java基础部分(直说重点部分)**

**1、 "=="和 equals 方法究竟有什么区别？**

**==如果判断值类型的话，判断内容是否相同。如果判断引用类型则是判断内存地址是否相同**

**2、静态变量和实例变量的区别？**

静态变量使用时,通过类名.名称,实例变量必须要初始化后才能使用。实例变量是实例化后才会分配空间,而静态变量当类加载时会分配空间。

**3、是否可以从一个 static 方法内部发出对非 static 方法的调用？**

 非static方法可以访问static方法.

   static方法不能访问非static方法

**4、Integer 与 int 的区别**

Integer 是引用类型,默认值是null。而int是是值类型默认值是0

**5、请说出作用域 public， private， protected，以及不写时的区别**

这四个作用域的可见范围如下表所示。

说明：如果在修饰的元素上面没有写任何访问修饰符，则表示 friendly。

**作用域 当前类 同一包（ package） 子孙类 其他包（ package）  
public     √              √                                      √                 √  
protected √            √                                      √                 ×  
friendly     √             √                                      ×                ×  
private      √              ×                                   ×                  ×**

**6、重载与重写区别？**

重载是同一个类中，方法名称相同， 但是参数或个数不同。与返回值没有关系。

重写是在多个类中， 产生继承关系。父类与子类的方法必须相同。

**7、接口与抽象类的区别？**

区别:定义接口的关键字是：interface 而定义抽象类的关键字是：abstract。  
接口中成员不能有私有， 抽象类可以。  
接口中定义的成员， 是finl public static 类型， 抽象类没有。  
接口中的不能有普通方法， 抽象类中可以。  
相同：   
两个都不new

但是 接口与抽象类是面向对象必备知识，设计模式、重构代码有必然作用

**8、 final, finally, finalize 的区别。**

  final 用于声明属性，方法和类，分别表示属性不可变，方法不可覆盖，类不可继承。  
内部类要访问局部变量，局部变量必须定义成 final 类型，例如，一段代码……  
finally 是异常处理语句结构的一部分，表示总是执行。  
finalize 是 Object 类的一个方法，在垃圾收集器执行的时候会调用被回收对象的此方法，可  
以覆盖此方法提供垃圾收集时的其他资源回收，例如关闭文件等。 JVM 不保证此方法总被  
调用

**9、Java的垃圾回收机制**

  Java的垃圾回收机制是Java虚拟机提供的能力，用于在空闲时间以不定时的方式动态回收无任何引用的对象占据的内存空间。

    System.gc()

   Runtime.getRuntime().gc()

**10、 反射的优缺点？**

**反射：就是正在运行动态读取这个类的完整信息。**

**优点：java的反射机制就是增加程序的灵活性、**

**缺点：缺点：（1）性能问题：使用反射基本上是一种解释操作，**

**用于字段和方法接入时要远慢于直接代码。因此反射机制主要应用在对灵活性和扩展性要求很高的系统框架上，普通程序不建议使用。**

**（2）使用反射会模糊程序内内部逻辑：程序员希望在源代码中看到程序的逻辑，反射等绕过了源代码的技术，因而会带来维护问题。反射代码比相应的直接代码更复杂。**

**4、那些地方用到了反射？**

**例如: jdbc、Java常用框架、jdk的动态代理、android的加载布局文件**

**12、java  中有几种类型的流？JDK 为每种类型的流提供了一些抽象类以供继承， 为每种类型的流提供了一些抽象类以供继承，**

**请说出他们分别是哪些类？**

   字节流，字符流。字节流继承于 InputStream OutputStream，字符流继承于

InputStreamReaderOutputStreamWriter。在 java.io 包中还有许多其他的流，主要是为了提

高性能和使用方便。

**13、字节流与字符流的区别**

       字节流什么都存,字符流只能存放中文

**14、多态(编译时不指定方法，运行时指定对象方法的引用)**

**②Java集合部分**

**1、** **ArrayList (线性表结构、数组)和 Vector 的区别**  
    a、这两个类都实现了List接口（List接口继承了Collection接口），

    b、Vector是线程安全的，也就是说是它的方法之间是线程同步的，而ArrayList是线程序不安全的，它的方法之间是线程不同步的。如果只有一个线程会访问到集合，那最好是使用ArrayList，因为它不考虑线程安全，效率会高些；如果有多个线程会访问到集合，那最好是使用Vector，因为不需要我们自己再去考虑和编写线程安全的代码。

**2、HashMap 和 Hashtable 的区别**

**a、HashMap线程不安全，HashTable线程安全。**

**b、HashTable不允许null值(key和value都不可以) ,HashMap允许null值(key和value都可以)。**

**3、List 和 Map 区别?**

  List集合是单列集合，Map集合是双列集合 通过key,value。

      List集合是有序集合,Map集合是无序集合。  
      List此接口的用户可以对列表中每个元素的插入位置进行精确地控制。  
       用户可以根据元素的整数 索引（在列表中的位置）访问元素，并搜索列表中的元素  
**4、 List, Set, Map 是否继承自 Collection 接口?**

        List， Set 是， Map 不是  
**5、List、 Map、 Set 三个接口，存取元素时，各有什么特点？**  
      list：存储： 有序的 可重复的

       访问：可以for循环，foreach循环，iterator迭代器 迭代。

      set：存储：无序的 不重复的

        访问：可以foreach循环，iterator迭代器 迭代

       map：存储：存储的是一对一对的映射 ”key=value“，key值 是无序，不重复的。value值可重复

        访问：可以map中key值转为为set存储，然后迭代这个set，用map.get(key)获取value

        也可以 转换为entry对象 用迭代器迭代

**6、 Map的数据结构**

     1.散裂图HashMap

     2.链式散裂图LinkedHashMap

     3. 树形图 TreeMap

**7、Arraylist与**

**LinkedList（**双向链表类**）区别?**

    ArrayList和LinkedList两者都实现了List接口，但是它们之间有些不同。

　　（1）ArrayList是由Array所支持的基于一个索引的数据结构，所以它提供对元素的随机访问，复杂度为O(1)，但LinkedList存储一系列的节点数据，每个节点都与前一个和下一个节点相连接。所以，尽管有使用索引获取元素的方法，内部实现是从起始点开始遍历，遍历到索引的节点然后返回元素，时间复杂度为O(n)，比ArrayList要慢。

　　（2）与ArrayList相比，在LinkedList中插入、添加和删除一个元素会更快，因为在一个元素被插入到中间的时候，不会涉及改变数组的大小，或更新索引。

　　（3）LinkedList比ArrayList消耗更多的内存，因为LinkedList中的每个节点存储了前后节点的引用。

**8、java中hashset和hashmap有什么区别**

1、HashSet底层是采用HashMap实现的。HashSet 的实现比较简单，HashSet 的绝大部分方法都是通过调用 HashMap 的方法来实现的，因此 HashSet 和 HashMap 两个集合在实现本质上是相同的。  
2、HashMap的key就是放进HashSet中对象，value是Object类型的。  
3、当调用HashSet的add方法时，实际上是向HashMap中增加了一行(key-value对)，该行的key就是向HashSet增加的那个对象，该行的value就是一个Object类型的常量

**③多线程部分**

**1、什么是多线程？**

在一个应用程序中,同时，有多个不同的执行路径。

**2、什么是线程同步、异步？**

          线程同步表示，当前线程执行完后下一个线程接着执行。

          线程异步表示,  在一个应用程序中,同时，有多个不同的执行路径。例如 javaweb ajax android handler

**3、线程之间如何同步**

线程之间同步使用 synchronized、wait 与 notify

**4、什么是线程不安全？如何解决？**

就是在多个线程共享同一个数据会受到其他线程的干扰。如何解决:使用线程同步技术， 用上锁(synchronized)。 让一个线程执行完了，在让另一个线程执行。

**5、如何创建一个线程？有几种方法？**

       继承thread类， 重写run方法、实现Runnalbe接口，重新run方法 ， 启动一个线程用start();

**6、sleep()和 wait()有什么区别?**

    a、sleep是让当前线程指定休眠时间，然后继续工作  不释放锁

        b、让当前线程wait则是等待，直到有线程通知notify（）唤醒他才会重新工作。释放锁

**.7、什么是线程池?**

   线程池是指在初始化一个多线程应用程序过程中创建一个线程集合，然后在需要执行新的任务时重用这些线程而不是新建一个线程。线程池中线程的数量通常完全取决于可用内存数量和应用程序的需求。然而，增加可用线程数量是可能的。线程池中的每个线程都有被分配一个任务，一旦任务已经完成了，线程回到池子中并等待下一次分配任务。

**8、为什么要线程池**

 基于以下几个原因在多线程应用程序中使用线程是必须的：

  1. 线程池改进了一个应用程序的响应时间。由于线程池中的线程已经准备好且等待被分配任务，应用程序可以直接拿来使用而不用新建一个线程。

  2. 线程池节省了CLR 为每个短生存周期任务创建一个完整的线程的开销并可以在任务完成后回收资源。

  3. 线程池根据当前在系统中运行的进程来优化线程时间片。

  4. 线程池允许我们开启多个任务而不用为每个线程设置属性。

  5. 线程池允许我们为正在执行的任务的程序参数传递一个包含状态信息的对象引用。

  6. 线程池可以用来解决处理一个特定请求最大线程数量限制问题。

 9、什么是死锁?

 10、wait()方法和notify()方法干什么用的，wait()方法有什么使用限制，是哪儿都能用么？

**④JavaEE部分**

**1、http get与Post区别?**

 get：传参是通过url传输，有大小限制差不多2KB吧，安全级别差，服务端Request.QueryString["id"],

 post：传参是通过form的action方式提交到服务器，基本无大小限制，安全级别比get要强点，隐藏提交URL参数

**2、解释一下什么是 servlet;**

 Servlet是一种服务器端的Java应用程序，具有独立于平台和协议的特性,可以生成动态的Web页面。 它担当客户请求（Web浏览器或其 他HTTP客户程序）与服务器响应（HTTP服务器上的数据库或应用程序）的中间层。 Servlet是位于Web 服务器内部的服务器端的Java 应用程序，与传统的从命令行启动的Java应用程序不同，Servlet由Web服务器进行加载，该Web服务器必须包含支持Servlet的Java虚拟机

**3、servlet线程安全吗？为什么？**

    servlet 是不安全的，同时一个servlet实现类只会有一个实例对象,也就是一个单利的、,所以多个线程是可能会访问同一个servlet实例对象的。

**4、说一说 Servlet 的生命周期?**

      servlet 有良好的生存期的定义，包括加载和实例化、初始化、处理请求以及服务结束。  
这个生存期由 javax.servlet.Servlet 接口的 init,service 和 destroy 方法表达。

**5、SERVLET API 中 forward()与 redirect()的区别？**

  forward是转发    是不显示URL请求,会发送一次请求 。两个资源之间是同一个  request对象,可以共享   request中的数据

    redirect()是重定向  而是重定向指向的新的URL,会发送两次请求。两个资源之间不是同一个  request对象,不可以共享

**6、拦截器、过滤器、监听器的区别？**

1.过滤器：所谓过滤器顾名思义是用来过滤的，在java web中，你传入的request,response提前过滤掉一些信息，或者提前设置一些   参数，然后再传入servlet或者struts的action进行业务逻辑，比如过滤掉非法url（不是login.do的地址请求，如果用户没有登陆都过滤掉）,或者在传入servlet或者struts的action前统一设置字符集，或者去除掉一些非法字符（聊天室经常用到的，一些骂人的话）。filter  流程是线性的， url传来之后，检查之后，可保持原来的流程继续向下执行，被下一个filter, servlet接收等.

2.监听器：这个东西在c/s模式里面经常用到，他会对特定的事件产生产生一个处理。监听在很多模式下用到。比如说观察者模式，就是一个监听来的。又比如struts可以用监听来启动。Servlet监听器用于监听一些重要事件的发生，监听器对象可以在事情发生前、发生后可以做一些必要的处理。

3.java的拦截器 主要是用在插件上，扩展件上比如 hivernate spring struts2等 有点类似面向切片的技术，在用之前先要在配置文件即xml文件里声明一段的那个东西。

**7、jsp 有哪些内置对象?作用分别是什么?**

答:JSP 共有以下9个内置的对象：

request 用户端请求，此请求会包含来自 GET/POST 请求的参数

response 网页传回用户端的回应

pageContext 网页的属性是在这里管理

session 与请求有关的会话期

application servlet 正在执行的内容

out 用来传送回应的输出

config servlet 的构架部件

page JSP 网页本身

exception 针对错误网页，未捕捉的例外

**8、 jsp 有哪些动作?作用分别是什么?**

答:JSP 共有以下6种基本动作  
jsp:include：在页面被请求的时候引入一个文件。  
jsp:useBean：寻找或者实例化一个 JavaBean。  
jsp:setProperty：设置 JavaBean 的属性。  
jsp:getProperty：输出某个 JavaBean 的属性。  
jsp:forward：把请求转到一个新的页面。  
jsp:plugin：根据浏览器类型为 Java 插件生成 OBJECT 或 EMBED 标记  
**9、页面间对象传递的方法**

 request， session， application， cookie 等

**10、JSP 和 Servlet 有哪些相同点和不同点，他们之间的联系是什么？**

JSP 是 Servlet 技术的扩展，本质上是 Servlet 的简易方式，更强调应用的外表表达。 JSP

编译后是"类 servlet"。 Servlet 和 JSP 最主要的不同点在于， Servlet 的应用逻辑是在 Java  
文件中，并且完全从表示层中的 HTML 里分离开来。而 JSP 的情况是 Java 和 HTML 可以  
组合成一个扩展名为.jsp 的文件。 JSP 侧重于视图， Servlet 主要用于控制逻辑。

10、MVC 的各个部分都有那些技术来实现?如何实现?

答:MVC 是 Model－View－Controller 的简写。 Model 代表的是应用的业务逻辑（通过  
JavaBean， EJB 组件实现）， View 是应用的表示面（由 JSP 页面产生）， Controller 是提供  
应用的处理过程控制（一般是一个 Servlet），通过这种设计模型把应用逻辑，处理过程和显  
示逻辑分成不同的组件实现。这些组件可以进行交互和重用。

**11、反射的优缺点？**

反射：就是正在运行动态读取这个类的完整信息。

优点：java的反射机制就是增加程序的灵活性、

缺点：缺点：（1）性能问题：使用反射基本上是一种解释操作，

用于字段和方法接入时要远慢于直接代码。因此反射机制主要应用在对灵活性和扩展性要求很高的系统框架上，普通程序不建议使用。

（2）使用反射会模糊程序内内部逻辑：程序员希望在源代码中看到程序的逻辑，反射等绕过了源代码的技术，因而会带来维护问题。反射代码比相应的直接代码更复杂。

4、那些地方用到了反射？

jdbc连接、常用框架、spring ioc、Java动态代理、jsp:useBean  android加载布局文件等。

**12、什么是Java序列化?**

我们有时候将一个 java 对象变成字节流的形式传出去或者从一个字节流中恢复成一个 java

对象，例如，要将 java 对象存储到硬盘或者传送给网络上的其他计算机，这个过程我们可

以自己写代码去把一个 java 对象变成某个格式的字节流再传输，但是， jre 本身就提供了这

种支持，我们可以调用 OutputStream 的 writeObject 方法来做，如果要让 java 帮我们做，

要被传输的对象必须实现 serializable 接口，这样， javac 编译时就会进行特殊处理，编译

的类才可以被 writeObject 方法操作，这就是所谓的序列化。需要被序列化的类必须实现

Serializable 接口，该接口是一个 mini 接口，其中没有需要实现的方法，

implementsSerializable 只是为了标注该对象是可被序列化的。

**⑤Java框架部分**

**①spring部分**

**1、说说spring？**

spring的核心是控制反转、依赖注入,aop（面向切面）相当于把每个bean与bean之间的关系交给第 三方容器进行管理.

**2、说springioc、springvaop?**

spring IOC  相当于把每个bean与bean之间的关系交给第三方容器管理。而这个容器就是spring

  springaop  面向切面的编程，或AOP， 是一种编程技术，允许程序模块化横向切割关注点，或横切典型的责任划分，如日志和事务管理。 springaop 就是用 Java的动态代理

**3、简述spring的优缺点？**

**spring 的优点？?**

  1.降低了组件之间的耦合性 ，实现了软件各层之间的解耦?

  2.可以使用容易提供的众多服务，如事务管理，消息服务等?

  3.容器提供单例模式支持

  4.容器提供了AOP技术，利用它很容易实现如权限拦截，运行期监控等功能?

  5.容器提供了众多的辅助类，能加快应用的开发?

  6.spring对于主流的应用框架提供了集成支持，如hibernate，JPA，Struts等?

  7.spring属于低侵入式设计，代码的污染极低?

  8.独立于各种应用服务器?

  9.spring的DI机制降低了业务对象替换的复杂性?

  10.Spring的高度开放性，并不强制应用完全依赖于Spring，开发者可以自由选择spring的部分或全部?

**缺点 用到了大量的反射机制，执行效率比较低**

**4、spring的生命周期？**

 Spring容器 从XML 文件中读取bean的定义，并实例化bean。

Spring根据bean的定义填充所有的属性。

如果bean实现了BeanNameAware 接口，Spring 传递bean 的ID 到 setBeanName方法。

如果Bean 实现了 BeanFactoryAware 接口， Spring传递beanfactory 给setBeanFactory 方法。

如果有任何与bean相关联的BeanPostProcessors，Spring会在postProcesserBeforeInitialization()方法内调用它们。

如果bean实现IntializingBean了，调用它的afterPropertySet方法，如果bean声明了初始化方法，调用此初始化方法。

如果有BeanPostProcessors 和bean 关联，这些bean的postProcessAfterInitialization() 方法将被调用。

如果bean实现了 DisposableBean，它将调用destroy()方法。

**5、Spring框架中的单例bean是线程安全的吗?**

不，Spring框架中的单例bean不是线程安全的。

**6、说说**spring的作用域

|  |  |
| --- | --- |
| singleton | 在每个Spring IoC容器中一个bean定义对应一个对象实例。 |
| prototype | 一个bean定义对应多个对象实例。 |
| request | 在一次HTTP请求中，一个bean定义对应一个实例；即每次HTTP请求将会有各自的bean实例， 它们依据某个bean定义创建而成。该作用域仅在基于web的Spring ApplicationContext 情形下有效。 |
| session | 在一个HTTP Session 中，一个bean定义对应一个实例。该作用域仅在基于web的SpringApplicationContext 情形下有效。 |
| global session | 在一个全局的HTTP Session 中，一个bean定义对应一个实例。典型情况下，仅在使用portlet context的时候有效。该作用域仅在基于web的Spring ApplicationContext 情形下有效。 |

**7. @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。

**8. @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。

**9. @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。

**10、Spring面向切面编程（AOP）**

**a.  解释AOP**

面向切面的编程，或AOP， 是一种编程技术，允许程序模块化横向切割关注点，或横切典型的责任划分，如日志和事务管理。

**b. 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。

**c.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。

**d. 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，它实际上是个应用程序执行Spring AOP的位置。

**e. 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。

**f. 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。**

**g. 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。

**f. 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。

**g. 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。

**h. 有几种不同类型的自动代理？**

BeanNameAutoProxyCreator

DefaultAdvisorAutoProxyCreator

Metadata autoproxying

**①hibernate部分**

**1、说一下orm与jdbc的区别？**

jdbc只是一个java操作数据库的规范接口而已

 orm不过是一种思想，对象关系映射。

 ORM：是对象关系模型，如hibernate,让你以面向对象的方式去编程。封装了JDBC.

JDBC：是从底层访问数据库服务器。一般银行，金融行业为了安全起见，直接用JDBC访问

**2、说说hibernate？**

**hibernate是一个基于ORM持久框架,可以让程序员以面向对象的思想操作数据库,提高生产效率.**

**hibernate的懒加载？ 有几种禁用方法**

在Hibernate框架中，当我们要访问的数据量过大时，明显用缓存不太合适， 因为内存容量有限 ，为了减少并发量，减少系统资源的消耗，这时Hibernate用懒加载机制来弥补这种缺陷，但是这只是弥补而不是用了懒加载总体性能就提高了。

我们所说的懒加载也被称为延迟加载，它在查询的时候不会立刻访问数据库，而是返回代理对象，当真正去使用对象的时候才会访问数据库。

1.使用代理对象：Hibernate.initialize("代理对象");

2.在需要禁用懒加载的映射文件中显示的加入lazy = "false"

3.使用openSessionInView【需要借助于过滤器】

**3、hibernate load与get区别？**

  load ：找不到数据的话会抛出org.hibernate.ObjectNotFoundException异常。

                此时hibernate会使用延迟加载加载机制

     get找不到的话会返回null。

1. 如果查询不到数据，get 会返回 null,但是不会报错, load 如果查询不到数据，则报错**ObjectNotFoundException**
2. 使用get 去查询数据，(先到一级/二级)会立即向db发出查询请求(select ...), 如果你使用的是 load查询数据，(先到一级、二级))即使查询到对象，返回的是一个代理对象,如果后面没有使用查询结果，它不会真的向数据库发select ,当程序员使用查询结果的时候才真的发出select ,这个现象我们称为懒加载(lazy)

**4、hibernate的三种状态？**

在Hibernate中，对象有三种状态：临 时状态(Transient)、持久状态(Persistent)和游离状态(Detached)。

处于持久态的对象也称为 PO(PersistenceObject),临时对象和游离对象也称为VO(ValueObject).

**5、hibernate有几种查询方式？**

1、 属性查询2、 参数查询、命名参数查询3、 关联查询4、 分页查询5、 统计函数

**6、hibernate的缓存机制**

Hibernate缓存包括两大类：Hibernate一级缓存和Hibernate二级缓存。

1.Hibernate一级缓存又称为“Session的缓存”。

Session内置不能被卸载，Session的缓存是事务范围的缓存（Session对象的生命周期通常对应一个数据库事务或者一个应用事务）。

一级缓存中，持久化类的每个实例都具有唯一的OID。

2.Hibernate二级缓存又称为“SessionFactory的缓存”。

由于SessionFactory对象的生命周期和应用程序的整个过程对应，因此Hibernate二级缓存是进程范围或者集群范围的缓存，有可能出现并发问题，因此需要采用适当的并发访问策略，该策略为被缓存的数据提供了事务隔离级别。

第二级缓存是可选的，是一个可配置的插件，默认下SessionFactory不会启用这个插件。

Hibernate提供了org.hibernate.cache.CacheProvider接口,它充当缓存插件与Hibernate之间的适配器。

**6、简述一下 hibernate的优缺点？**

1.Hibernate的优缺点：  
优点：1、程序更加面向对象；  
2、提高了生产率；  
3、方便移植（修改配置文件）；  
4、无侵入性。  
缺点：  
1、效率比JDBC略差；  
2、不适合批量操作。 **如何优化hibernate？**

**Mybatis部分**

**1、mybatis与hibernate区别？**

Hibernate是面向对象的思想操作数据、mybatis是以纯sql操作数据

相对于mybatis容易优化.扩展性好，但是移植性差。

2、**mybatis如何动态sql？**

MyBatis的动态SQL是基于OGNL表达式的

**SpringMVC部分**

1、springmvc工原理

   1.spring mvc请所有的请求都提交给DispatcherServlet,它会委托应用系统的其他模块负责负责对请求进行真正的处理工作。

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

3.DispatcherServlet请请求提交到目标Controller

4.Controller进行业务逻辑处理后，会返回一个ModelAndView

5.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

6.视图对象负责渲染返回给客户端。

**⑥设计模式部分**

**代理、模板方法、外观设计模式、单例**

总体来说设计模式分为三大类：

创建型模式，共五种：工厂方法模式、抽象工厂模式、单例模式、建造者模式、原型模式。

结构型模式，共七种：适配器模式、装饰器模式、代理模式、外观模式、桥接模式、组合模式、享元模式。

行为型模式，共十一种：策略模式、模板方法模式、观察者模式、迭代子模式、责任链模式、命令模式、备忘录模式、状态模式、访问者模式、中介者模式、解释器模式。

**1、什么是代理设计模式?**

代理设计模式:**为另一个对象提供一个替身,以控制对这个对象的访问。**

**静态代理:**

**1）代理对象，要实现与目标对象一样的接口；**

**总结静态代理：**

**1）可以做到在不修改目标对象的功能前提下，对目标对象功能扩展。**

**2）缺点：**

**--》  因为代理对象，需要与目标对象实现一样的接口。所以会有很多代理类，类太多。**

**--》  一旦接口增加方法，目标对象与代理对象都要维护。**

**动态代理:**

**动态代理，**

**1）代理对象，不需要实现接口；**

**2）代理对象的生成，是利用JDKAPI，动态的在内存中构建代理对象(需要我们指定创建代理对象/目标对象实现的接口的类型；);**

**3)  动态代理， JDK代理，接口代理；**

**JDK中生成代理对象的API：**

**动态代理总结：**

**代理对象不需要实现接口，但是目标对象一定要实现接口；否则不能用动态代理！**

**1.4 Cglib代理**

**Cglib代理，也叫做子类代理。在内存中构建一个子类对象从而实现对目标对象功能的扩展。**

* **JDK的动态代理有一个限制，就是使用动态代理的对象必须实现一个或多个接口。如果想代理没有实现接口的类，就可以使用CGLIB实现。**
* **CGLIB是一个强大的高性能的代码生成包，它可以在运行期扩展Java类与实现Java接口。它广泛的被许多AOP的框架使用，例如Spring AOP和dynaop，为他们提供方法的interception（拦截）。**
* **CGLIB包的底层是通过使用一个小而快的字节码处理框架ASM，来转换字节码并生成新的类。不鼓励直接使用ASM，因为它要求你必须对JVM内部结构包括class文件的格式和指令集都很熟悉。**

**Cglib子类代理：**

**1) 需要引入cglib – jar文件，但是spring的核心包中已经包括了cglib功能，所以直接引入spring-core-3.2.5.jar即可。**

**2）引入功能包后，就可以在内存中动态构建子类**

**3）代理的类不能为final，否则报错。**

**4）目标对象的方法如果为final/static, 那么就不会被拦截，即不会执行目标对象额外的业务方法。**

**使用场景: spring aop、日志、权限控制、事物。**

**2、什么是模板方法设计模式?**

**模板方法就是定义一个算法的骨架,具体的算法让子类去实现**

**3、什么是外观设计模式？**

**外观设计模式就像spring一样,为了解决类与类的依赖.**

**4、什么是单例?**

单例对象（Singleton）是一种常用的设计模式。在Java应用中，单例对象能保证在一个JVM中，该对象只有一个实例存在。这样的模式有几个好处：

1、某些类创建比较频繁，对于一些大型的对象，这是一笔很大的系统开销。

2、省去了new操作符，降低了系统内存的使用频率，减轻GC压力。

3、有些类如交易所的核心交易引擎，控制着交易流程，如果该类可以创建多个的话，系统完全乱了。（比如一个军队出现了多个司令员同时指挥，肯定会乱成一团），所以只有使用单例模式，才能保证核心交易服务器独立控制整个流程。

单例分类为 懒汉式 和饿汉式

区别 懒汉式是当你需要的时候才会new,饿汉式当类加载就会创建该对象

**第一种（懒汉，线程安全）：**

**public** **class** Singleton {

1. **private** **static** Singleton instance;
2. **private** Singleton (){}
3. **public** **static** **synchronized** Singleton getInstance() {
4. **if** (instance == **null**) {
5. instance = **new** Singleton();
6. }
7. **return** instance;
8. }
9. }

**第二种（饿汉）：**

1. **public** **class** Singleton {
2. **private** **static** Singleton instance = **new** Singleton();
3. **private** Singleton (){}
4. **public** **static** Singleton getInstance() {
5. **return** instance;
6. }
7. }

**⑦网络部分**

**1、说说http**

    HTTP（Hypertext Transfer Protocol），即超文本传输协议。是WWW浏览器和WWW服务器之间的应用层通讯协议。HTTP协议是基于TCP/IP之上的协议

**2、说说socket**

**1、tcp与udp区别？**

            udp:   a、是面向无连接, 将数据及源的封装成数据包中,不需要建立建立连接

                      b、每个数据报的大小在限制64k内

                      c、因无连接,是不可靠协议

                      d、不需要建立连接,速度快

              tcp：  a、建议连接，形成传输数据的通道.

                     b、在连接中进行大数据量传输

                     c 通过三次握手完成连接,是可靠协议

                     d 必须建立连接m效率会稍低

            聊天、网络视频会议、桌面共享用的就是  udp

**2、说说三次握手?**

      1）第一次握手：建立连接时，客户端A发送SYN包（SYN=j）到服务器B，并进入SYN\_SEND状态，等待服务器B确认。   
（2）第二次握手：服务器B收到SYN包，必须确认客户A的SYN（ACK=j+1），同时自己也发送一个SYN包（SYN=k），即SYN+ACK包，此时服务器B进入SYN\_RECV状态。   
（3）第三次握手：客户端A收到服务器B的SYN＋ACK包，向服务器B发送确认包ACK（ACK=k+1），此包发送完毕，客户端A和服务器B进入ESTABLISHED状态，完成三次握手。   
完成三次握手，客户端与服务器开始传送数据。

**3、什么是webserivce?**

Webservice的一个最基本的目的就是提供在各个不同平台的不同应用系统的协同工作能力。

Web service 就是一个应用程序，它向外界暴露出一个能够通过Web进行调用的API。

SOAP是一种简单基于xml的轻量协议，用户web上交换结构化信息和类型信息。

soap请求是HTTP POST的一个专用版本，遵循一种特殊的xml消息格式Content-type设置为: text/xml任何数据都可以xml化。

**⑧数据库部分**

**1、什么是sql注入，如何防止？**

sql注入：就是通过把SQL命令插入到Web表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的SQL命令。

简单说就是sql拼接。

如何防御：a：删除用户输入内容中的所有连字符

b:对于用来执行查询的数据库帐户，限制其权限。用不同的用户帐户执行查询、插入、更新、删除操作。由于隔离了不同帐户可执行的操作

c: 用存储过程来执行所有的查询。SQL参数的传递方式将防止攻击者利用单引号和连字符实施攻击

d:查用户输入的合法性，确信输入的内容只包含合法的数据

6、sql如何优化？

a：要显示什么字段， 就查询什么字段

b: 建立索引

c：减少对表的查询

**d:分表分库**

**2、悲观锁与乐观锁的区别？**

悲观锁(Pessimistic Lock), 顾名思义，就是很悲观，每次去拿数据的时候都认为别人会修改，所以每次在拿数据的时候都会上锁，这样别人想拿这个数据就会block直到它拿到锁。传统的关系型数据库里边就用到了很多这种锁机制，比如行锁，表锁等，读锁，写锁等，都是在做操作之前先上锁。

乐观锁(Optimistic Lock), 顾名思义，就是很乐观，每次去拿数据的时候都认为别人不会修改，所以不会上锁，但是在更新的时候会判断一下在此期间别人有没有去更新这个数据，可以使用版本号等机制。乐观锁适用于多读的应用类型，这样可以提高吞吐量，像数据库如果提供类似于write\_condition机制的其实都是提供的乐观锁。

两种锁各有优缺点，不可认为一种好于另一种，像乐观锁适用于写比较少的情况下，即冲突真的很少发生的时候，这样可以省去了锁的开销，加大了系统的整个吞吐量。但如果经常产生冲突，上层应用会不断的进行retry，这样反倒是降低了性能，所以这种情况下用悲观锁就比较合适。

应用场景:相关实例:<https://yq.aliyun.com/articles/6899>

**3、数据库各种事务隔离级别**

**4、数据三大范式**

**5、什么是索引?**

**6、左连接与有连接的区别?**

**a、左连接：只要左边表中有记录，数据就能检索出来，而右边有**

的记录必要在左边表中有的记录才能被检索出来

    b、右连接：右连接是只要右边表中有记录，数据就能检索出来

**7、union 与union all区别**

union 将两个表连接后删除其重复的项；   
union all 将两个表连接都不删除其重复的项。

**⑨前端面试题**

**什么是jquery? jquery的选择器有几种？**  
1、基本选择器  
2、层次选择器  
3、过滤选择器  
4、属性选择器

**⑩linux部分**

1、显示日期的指令： date

2、显示日历的指令：cal

7、惯用的关机指令：shutdown

12、查看版本信息等

13、变换目录：cd

14、显示当前所在目录：pwd

15、建立新目录：mkdir

17、档案与目录的显示：ls

18、复制档案或目录：cp

19、移除档案或目录：rm

22、由第一行开始显示档案内容：cat

28、只看尾几行：tail

34、观察文件类型：file

grep:文本内容搜索;find:文件或者目录名以及权限属主等匹配搜索

**(11)其他部分**

**1、xml与json的区别**

**？**

区别:

xml是重量级、json是轻量级

xml比较占带宽、json占带宽小,易于压缩

json在webservice 用的比较少、xml用的较多

相同：

两者都用在项目交互下  例如  移动app接口用的就是json、在web项目中与其他项目对接用xml较多。

json常用解析方法 gson、jsonobject、jackson等 xml dom sax pull 解析

**2、redis使用场景**

**redis是 缓存数据库，能够减轻数据库访问操作,比关系数据访问效率高！  使用场景: 发送短信验证码计时、token生成。**

**3、JavaWeb安全编程**

  1、xss攻击

     什么恶意执行javascript代码或html代码 例如获取用户cookie、论坛恶意留言

**4、sql注入**

**5 尽量少用隐藏域、或不要暴露敏感数据信息。**

 例如:很多网站忘记密码, 通过手机号码找回,前面是手机号码+图像验证码。点击下一步是讲上一个页面的手机号放到隐藏域中。发送验证码取隐藏域里面的值。这样我讲隐藏域里面的值修改自己的手机。就可以破解别人的账号密码

**6、App接口设计一定要规范**

最好项目更新时最好用 网站安全扫描工具扫描一下。

**（十二）并发问题**

 定时器怎么实现并发和分布式