|  |
| --- |
| **技术问题(至少50个)** |
| **问题1：**创建多线程有哪些方式？  回答： 1、使用继承Therad类方式 继承Thread类重写run方法 2、使用实现runable接口方式3、使用匿名内部类方式 4、callable 5、使用线程池创建线程。  **问题2：**sql优化、数据库优化  回答：数据库优化：  数据库优化吧我觉应该从硬盘、内存和网络带宽考虑，提高硬盘的读写速度，增大带宽提高吞吐量，增大服务器内存，可以采用读写分离，降低单台数据库的访问压力，查询的时候控制数据量的大小，返回更少数据，减少交互次数，减少cpu及内存的开销。  sql优化：1.选择最有效率的表名顺序;   1. WHERE子句中的连接顺序; 2. SELECT子句中避免使用‘ \* ‘; 3. 减少访问数据库的次数; 4. 使用DECODE函数来减少处理时间; 5. 整合简单,无关联的数据库访问; 6. 删除重复记录。   **问题3：**Eureka如何实现高可用？  回答：启动多台Eureka服务器，然后作为SpringCloud服务互相注册，客户端从Eureka获取信息时，按照注册的Eureka顺序对第一个Eureka进行访问。  **问题4：**Zookeeper实现分布式锁的原因？  回答：多个jvm在同一个zookeeper上创建同一个节点，如果那个jvm能创建成功，就表示它拿到了锁，剩下的jvm保持对这个节点的监听，一旦发现这个节点被删除了，那么剩下的jvm就重新再创建这个节点，谁能创建成功谁能拿到锁，依次循环下去。  **问题5：https与http区别？**  　1、https 协议需要到 ca 申请证书，一般免费证书较少，因而需要一定费用。  　2、http 是超文本传输协议，信息是明文传输，https 则是具有安全性的 ssl 加密传输协议。  　3、http 和 https 使用的是完全不同的连接方式，用的端口也不一样，前者是 80，后者是 443。  　4、http 的连接很简单，是无状态的；HTTPS 协议是由 SSL+HTTP 协议构建的可进行加密传输、身份认证的网络协议，比 http 协议安全。 **问题6：**Redis 的好处 (1) 速度快，因为数据存在内存中，类似于HashMap，HashMap的优势就是查找和操作的时间复杂度都是O(1)  (2) 支持丰富数据类型，支持string，list，set，zset，hash  (3) 支持事务，操作都是原子性，所谓的原子性就是对数据的更改要么全部执行，要么全部不执行  (4) 丰富的特性：可用于缓存消息，按key设置过期时间，过期后将会自动删除。 **问题7：**redis在项目中遇到的问题？ 1).缓存穿透的问题:  一般出现这样的问题，是因为当我们查询一条肯定不存在的数据的时候，缓存中没有，就会透过缓存来查询数据库，数据库也不存在，这样就会将值保存在缓存中最后还是缓存和数据库中都没有，如果一直访问这条数据。我们就对数据库进行频繁的查询给数据库带来压力  解决办法：当查询的时候，如果缓存和数据库中都没有，我们就将这个数据以空的形式存放在缓存中，（或者是给一个false的标示）这样就不用去数据库就可以知道不存在，减少对数据库查询的次数。当我们这个值发生改变的时候，我们在重新赋值  2.并发情况:  当我们大量访问都是查询一个缓存中没有的数据时，这样就会去数据库中进行查询，可能会造成数据库宕机，  解决办法：在查询的时候，我给它添加一个同步锁，只有第一条数据去，数据库中查询并返回到redis中后才能查询，这是数据库中已近存在了值，这样可以避免  3.雪崩:  大量数据的访问缓存超时，这样用户就会访问到数据库，第一台数据库崩溃了，访问就会到第二台数据库进行查询，这样就会导致额第二台崩溃。  解决办法：就是设置失效时间，不要一起失效，或者是设置在访问少的时候，或者设置永远不失效。  **问题8：**哨兵的详细解释  哨兵当时害怕的一点，就是说如果我要是这个一redis不够用，如果这个连接数满了，我们这个可能就是不能及时获取数据，所以我配了多个的一个集群，集群一主多从，万一主库一挂就不能更新数据了，所以我们配了个哨兵，Sentinel(三 te nao)能够监控整个这个集群。  同时向所有的这个里边的节点发生心跳，谁要死了就踢出去，主要是死了，直接从里边呢进行一个选举选举的时候，有一个选举算法选出来一个新的一个主，然后整个集群不会崩的，有哨兵的情况下什么时候集群就崩了，当这所有的节点达到一半以上全死的时候，这个集群就崩了。Sentinel有一个redis-Sentinel这么一个服务，你把它启动的时候，就类似于启动了一个redis-server一样，你启动它就起来了。它也有配置文件，它配置的时候需要监控的是主节点，主节点一旦被监控到，这个主节点下的所有从节点都会被他监控到。 **问题9：Redis 和Mongodb 的 区别？** 1.Mogodb 是非关系数据，基于bson格式的数据库  ，有关键字可以进行分页，姓名模糊匹配，大于小于等，mogondb是最向关系型数据库的非关系数据库。  2.Redis 是一个缓存的技术，mongo的数据都存在硬盘当中，而redis数据是存放在内存当中的，只是它支持 持久化到硬盘的操作，所以可以理解为 非关系数据库，但需要注意的是，redis是通过键值对的存储方式，不支持模糊，条件查询。  **问题10：**什么是守护线程？  回答：Java中有两种线程，一种是用户线程，另一种是守护线程(gc线程)。用户线程是指用户自定义创建的线程，主线程停止，用户线程不会停止，守护线程当进程不存在或主线程停止，守护线程也会被停止。使用setDaemon(true)方法设置为守护线程。 **问题11：**什么是事务，并解释四大特性 事务(Transaction)是访问并可能更新数据库中各种数据项的一个程序执行单元(unit)。  特性：事务是恢复和并发控制的基本单位。  事务应该具有4个属性：原子性、一致性、隔离性、持久性。这四个属性通常称为ACID特性。  原子性（atomicity）。一个事务是一个不可分割的工作单位，事务中包括的诸操作要么都做，要么都不做。  一致性（consistency）。事务必须是使数据库从一个一致性状态变到另一个一致性状态。一致性与原子性是密切相关的。  隔离性（isolation）。一个事务的执行不能被其他事务干扰。即一个事务内部的操作及使用的数据对并发的其他事务是隔离的，并发执行的各个事务之间不能互相干扰。  持久性（durability）。指一个事务一旦提交，它对数据库中数据的改变就应该是永久性的。接下来的其他操作或故障不应该对其有任何影响。 **问题12：**什么是分布式事务？ 回答:简单的说，就是一次大的操作由不同的小操作组成，这些小的操作分布在不同的服务器上，且属于不同的应用，分布式事务需要保证这些小操作要么全部成功，要么全部失败。本质上来说，分布式事务就是为了保证不同数据库的数据一致性。  **问题13：**spring 事务实现方式有哪些？  回答: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。 **问题14：**接口和抽象类的区别 1. 抽象类可以有构造方法，接口中不能有构造方法。  2. 抽象类中可以有普通成员变量，接口中没有普通成员变量  3. 抽象类中可以包含非抽象的普通方法，接口中的所有方法必须都是抽象的，不能有非抽象的普通方法。  4. 抽象类中的抽象方法的访问类型可以是public，受保护的和默认类型，但接口中的抽象方法只能是public类型的，并且默认即为public abstract(啊不si zhua ke de )类型。  5. 抽象类中可以包含静态方法，接口中不能包含静态方法  6. 抽象类和接口中都可以包含静态成员变量，抽象类中的静态成员变量的访问类型可以任意，但接口中定义的变量只能是public static final类型，并且默认即为public static final类型。  7. 一个类可以实现多个接口，但只能继承一个抽象类。 **问题15：**SpringMVC的注解 @RequestMapping声明方法的具体访问路径，@RequestMapping放在类上时相当于struts2的@nameSpace，就是在具体访问路径之前再加一层目录。  @RequestParam相当于request.getAttribute()方法，从request作用域对象中获取对应的参数值。  @ResponseBody 把返回的参数转换成xml或者json格式，主要与ajax请求结合使用。 **问题16：**什么是对象的序列化？什么是对象的反序列化 当两个进程在进行远程通信时，彼此可以发送各种类型的数据。无论是何种类型的数据，都会以二进制序列的形式在网络上传送。发送方需要把这个Java对象转换为字节序列，才能在网络上传送；接收方则需要把字节序列再恢复为Java对象。  把Java对象转换为字节序列的过程称为对象的序列化。  把字节序列恢复为Java对象的过程称为对象的反序列化。  对象的序列化主要有两种用途：  1） 把对象的字节序列永久地保存到硬盘上，通常存放在一个文件中；  2） 在网络上传送对象的字节序列。  **问题17：**spring 自动装配 bean 有哪些方式？  回答：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。  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**。** **问题18：**hashMap底层 HashMap底层实现是一个线性数组Entry。  HashMap其实也是一个线性的数组实现的,所以可以理解为存储数据的容器就是一个线性数组。  首先HashMap里面实现一个静态内部类Entry，重要的属性有 key , value, next，从属性key,value，我们就能很明显的看出来Entry就是HashMap键值对实现的一个基础bean，所以说到HashMap的基础就是一个线性数组，Map里面的内容都保存在Entry[]里面。 **问题19：**HashMap为什么线程不安全？ HashMap内部存储使用了一个Node数组(默认大小是16)，而Node类包含一个类型为Node的next的变量，也就是相当于一个链表，所有hash值相同(即产生了冲突)的key会存储到同一个链表里，  HashMap在并发执行put操作时会引起死循环，导致CPU利用率接近100%。因为多线程会导致HashMap的Node链表形成环形数据结构，一旦形成环形数据结构，Node的next节点永远不为空，就会在获取Node时产生死循环。hashMap默认的负载因子大小为0.75，也就是说，当一个map填满了75%的bucket时候，和其它集合类(如ArrayList等)一样，将会创建原来HashMap大小的两倍的bucket数组，来重新调整map的大小，并将原来的对象放入新的bucket数组中。  **问题20：**Zookeeper实现分布式锁与Redis实现分布式锁区别  回答：Zookeeper通过创建临时节点和利用监听事件实现分布式锁，Redis使用setnx命令创建相同的key,因为redis的key保证唯一，先创建的先获取锁。  Zookeeper实现分布式锁，即使获取不到锁创建对锁的监听即可，不需要不断去尝试获取锁，性能开销小  Redis实现分布式锁，如果客户端获取到锁的时候遇到bug或挂了，还需要等到超时时间过了以后才能重新获取锁  Zookeeper实现分布式锁，创建的是临时节点，客户端挂了，节点自然删除，也就达到了自动释放锁的效果  **问题21：**throw 和 throws 的区别  throw：  throw语句用在方法体内，表示抛出异常，由方法体内的语句处理。  throw 是具体向外抛出异常的动作，所以它抛出的是一个异常实例，执行throw 一定是抛出了某种异常。  throws：  throws语句是用在方法声明后面，表示如果抛出异常，由该方法的调用者来进行异常的处理。  throws 主要是声明这个方法会抛出某种类型的异常，让它的使用者要知道需要捕获的异常的类型。  throws表示出现异常的一种可能性，并不一定会发生这种异常。 **问题22：什么是微服务？** 以前的模式是所有的代码在同一个工程中 部署在同一个服务器中 同一个项目的不同模块不同功能互相抢占资源，微服务 将工程根据不同的业务规则拆分成微服务 微服务部署在不同的机器上，服务之间进行相互调用。java微服务的框架有dubbo（只能用来做微服务），springCloud（提供了服务的发现，断路器等）。 **问题23：**什么是spring cloud？ Spring Cloud为开发人员提供了快速构建分布式系统的一些通用模式（例如配置管理，服务发现，断路器，智能路由，微代理，控制总线，一次性令牌，全局锁，领导选举，分布式会话，群集状态）。分布式系统的协调导致了锅炉板模式，并且使用 Spring Cloud开发人员可以快速地站起来来实现这些模式的服务和应用程序。它们可以在任何分布式环境中正常工作，包括开发人员自己的笔记本电脑，裸机数据中心和受管平台，Cloud Foundry。 **问题24：springcloud如何实现服务的注册和发现？** 服务在发布时 指定对应的服务名（服务名包括了IP地址和端口） 将服务注册到注册中心（eureka或者zookeeper），这一过程时springcloud自动实现只需要在main方法添加 @EnableDisscoveryClient 同一个服务修改端口就可以启动多个实例，调用方法：传递服务名称通过注册中心获取所有的可用实例，通过负载均衡策略调用（ribbon和feign）对应的服务。  **问题25：Ribbon和Feign的区别**  Ribbon和Feign都是用于调用其他服务，不过方式不同。  1、启动类使用的注解不同，Ribbon使用的时@RibbonClient，Feign用的时@EnableFeignClients；  2、服务的指定位置不同，Ribbon是在@RibbonClient注解上声明，Feign则是在定义抽象方法的接口使用@FeignClient声明。  3、调用方式不同，Ribbon需要自己构建http请求，模拟http请求然后使用RestTemplate发送给其他服务，步骤相当繁琐。Feign则是在Ribbon的基础进行了一次改进，采用接口的方式，将需要调用的其他服务的方法定义成抽象方法即可。不需要自己构建http请求。不过要注意的是抽象方法的注解、方法签名要和提供服务的方法完全一致。  **问题26：**Kafka是如何消费信息?  在Kafka中传递消息是通过使用sendfile API完成的。它支持将字节从套接口转移到磁盘，通过内核空间保存副本，并在内核用户之间调用内核。 **问题27：**Rabbitmq消息队列 当初我们做短信发送的时候用到了rabbitmq消息队列，我们当时有一个短信发送平台，平台采用监听消息队列方式进行短信的发送，我们在项目中通过配置一个RebbitConfig在上面加@Configuration注解，然后在里边new一个Queue对象创建一个消息队列，然后我们再发送短信的地方注入AmqpTemplate,最后通过调用AmqpTemplate.convertAndSend将消息发送到队列当中，convertAndSend里边有两个参数，第一个是消息队列名称，第二个是消息队列发送进去的内容。短信发送平台通过RabbitListener queues等于队列名称，监听队列，通过RabbitHandler注解获取到消息队列中的数据，然后通过httpClient调用短信发送接口，将短信发送出去并将发送的数据以及发送状态记录到数据库中。  **问题28：**多线程的原理  同一时间，CPU只能处理1条线程，只有1条线程在工作（执行）多线程并发（同时）执行，其实是CPU快速地在多条线程之间调度（切换）如果CPU调度线程的时间足够快，就造成了多线程并发执行的假象。  思考：如果线程非常非常多，会发生什么情况？  CPU会在N多线程之间调度，CPU会累死，消耗大量的CPU资源  每条线程被调度执行的频次会降低（线程的执行效率降低）  **问题29：**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。  Java提供的四种线程池的好处在于：  a. 重用存在的线程，减少对象创建、消亡的开销，性能佳。  b. 可有效控制最大并发线程数，提高系统资源的使用率，同时避免过多资源竞争，避免堵塞。  c. 提供定时执行、定期执行、单线程、并发数控制等功能 **问题30：**wait和sleep的区别？ wait时会释放锁资源但sleep不会释放锁资源，wait通常和notify以及notifyAll结合使用，需要notify或者notifyAll对其进行唤醒，sleep通常在指定的时间内自动唤醒。 **问题31：**解决线程安全问题的方案？ a、通过加synchronized同步锁  b、避免使用全局变量和静态变量  c、ThreadLocal类用来提供线程内部的局部变量 **问题32：**死锁产生的原因？ 是由于访问共享资源顺序不当造成的。简单的说，死锁就是指两个或两个以上的线程在执行过程中，因争  夺资源而造成的一种互相等待的现象，如果没有外力作用，他们都将无法继续执行下去。  **问题33：**网站跨域解决方案  回答：1.使用jsonp缺点只能发送get请求  2.使用httpclient进行转发，效率低  3.设置响应头允许跨域  4.使用Nginx搭建api网关  5.使用Zuul微服务搭建api接口网关  **问题34：**在数据制作过程中，你如何能从Kafka得到准确的信息?  在数据中，为了精确地获得Kafka的消息，你必须遵循两件事: 在数据消耗期间避免重复，在数据生产过程中避免重复。这里有两种方法，可以在数据生成时准确地获得一个语义:  1.每个分区使用一个单独的写入器，每当你发现一个网络错误，检查该分区中的最后一条消息，以查看您的最后一次写入是否成功。  2.在消息中包含一个主键(UUID或其他)，并在用户中进行反复制。 **问题35：**final,finally,finalize 三者区别？**** **Final是一个修饰符：**  当final修饰一个变量的时候，变量变成一个常量，它不能被二次赋值，当final修饰的变量为静态变量（即由static修饰）时，必须在声明这个变量的时候给它赋值。当final修饰方法时，该方法不能被重写，当final修饰类时，该类不能被继承Final不能修饰抽象类，因为抽象类中会有需要子类实现的抽象方法，（抽 象类中可以有抽象方法，也可以有普通方法，当一个抽象类中没有抽象方法时，这个抽象类也就没有了它存在的必要）Final不能修饰接口，因为接口中有需要其实现类来实现的方法。  **Finally：**  Finally只能与try/catch语句结合使用，finally语句块中的语句一定会执行， 并且会在return，continue，break关键字之前执行  **finalize：**  Finalize是一个方法，属于java.lang.Object类，finalize()方法是GC（garbage collector垃圾回收）运行机制的一部分，finalize()方法是在GC清理它所从属的对象时被调用的。  **问题36：**什么是JTA?  JTA是Java Transaction API简称 ，即Java事务管理器，主要管理本地事务。本地事务是只同一个数据库源中，多次数据库连接之间的事务，使得这多个数据表操作遵循ACID（原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）、持久性（Durability））JTA是基于XA架构上建模的，在JTA 中，事务管理器抽象为javax.transaction.TransactionManager接口，并通过底层事务服务（即JTS）实现。 **问题37：**为什么Redis需要把所有数据放到内存中？ Redis为了达到最快的读写速度将数据都读到内存中，并通过异步的方式将数据写入磁盘。所以redis具有快速和数据持久化的特征。如果不将数据放在内存中，磁盘I/O速度为严重影响redis的性能。  **问题38：**Redis集群方案应该怎么做？都有哪些方案？  1.twemproxy，大概概念是，它类似于一个代理方式，使用方法和普通redis无任何区别，设置好它下属的多个redis实例后，使用时在本需要连接redis的地方改为连接twemproxy，它会以一个代理的身份接收请求并使用一致性hash算法，将请求转接到具体redis，将结果再返回twemproxy。使用方式简便(相对redis只需修改连接端口)，对旧项目扩展的首选。 问题：twemproxy自身单端口实例的压力，使用一致性hash后，对redis节点数量改变时候的计算值的改变，数据无法自动移动到新的节点。  2.codis，目前用的最多的集群方案，基本和twemproxy一致的效果，但它支持在 节点数量改变情况下，旧节点数据可恢复到新hash节点。  3.redis cluster3.0自带45的集群，特点在于他的分布式算法不是一致性hash，而是hash槽的概念，以及自身支持节点设置从节点。具体看官方文档介绍。  4.在业务代码层实现，起几个毫无关联的redis实例，在代码层，对key 进行hash计算，然后去对应的redis实例操作数据。 这种方式对hash层代码要求比较高，考虑部分包括，节点失效后的替代算法方案，数据震荡后的自动脚本恢复，实例的监控，等等。  **问题39：**怎么理解Redis事务？  事务是一个单独的隔离操作：事务中的所有命令都会序列化、按顺序地执行。事务在执行的过程中，不会被其他客户端发送来的命令请求所打断。事务是一个原子操作：事务中的命令要么全部被执行，要么全部都不执行。 **问题40：Get和Post的区别**get是从服务器上获取数据，post是向服务器传送数据，get传送的数据量较小，不能大于2KB。post传送的数据量较大，一般被默认为不受限制。3.get安全性非常低，post安全性较高。但是执行效率却比Post方法好。 4.在进行文件上传时只能使用post而不能是get。  5.get传输的数据在地址栏可见,post的数据不可见。  **问题41：**String、StringBuffer、StringBuilder的区别？  String：可变不可变，字符串常量，在修改时不会改变自身；若修改，等于重新生成新的字符串对象。  StringBuffer：在修改时会改变对象自身，每次操作都是对StringBuffer对象本身进行修改，不是生成新的对象；使用场景：对字符串经常改变情况下，主要方法：append（），insert（）等。  String：对象定义后不可变，线程安全。  StringBuffer：是线程安全的（对调用方法加入同步锁），执行效率较慢，适用于多线程下操作字符串缓冲区大量数据。  StringBuilder：是线程不安全的，适用于单线程下操作字符串缓冲区大量数据。 **问题42：**Forword(请求转发)与Redirect(重定向) 1、从数据共享上  Forword是一个请求的延续，可以共享request的数据  Redirect开启一个新的请求，不可以共享request的数据  2、从地址栏  Forword转发地址栏不发生变化  Redirect转发地址栏发生变化 **问题43：**SpringMvc的运行原理 整个处理过程从一个HTTP请求开始：  1.Tomcat在启动时加载解析web.xml,找到spring mvc的前端总控制器DispatcherServlet,并且通过DispatcherServlet来加载相关的配置文件信息。  2.DispatcherServlet接收到客户端请求，找到对应HandlerMapping，根据映射规则，找到对应的处理器（Handler）。  3.调用相应处理器中的处理方法，处理该请求后，会返回一个ModelAndView。  4.DispatcherServlet根据得到的ModelAndView中的视图对象，找到一个合适的ViewResolver（视图解析器），根据视图解析器的配置，DispatcherServlet将要显示的数据传给对应的视图，最后显示给用户。 **问题44：**dubbo六种容错策略一．Failover Cluster 模式 失败自动切换，当出现失败，重试其它服务器。(缺省) 通常用于读操作，但重试会带来更长延迟。 可通过retries=”2”来设置重试次数(不含第一次)。  二．Failfast Cluster  快速失败，只发起一次调用，失败立即报错。通常用于非幂等性的写操作，比如新增记录。  三．Failsafe Cluster  失败安全，出现异常时，直接忽略。通常用于写入审计日志等操作。  四．Failback Cluster  失败自动恢复，后台记录失败请求，定时重发。通常用于消息通知操作。  五．Forking(fo gen) Cluster  并行调用多个服务器，只要一个成功即返回。通常用于实时性要求较高的读操作，但需要浪费更多服务资源。 可通过forks=”2”来设置最大并行数。  六．Broadcast(bu ruo ka si te) Cluster  广播调用所有提供者，逐个调用，任意一台报错则报错。(2.1.0开始支持)通常用于通知所有提供者更新缓存或日志等本地资源信息  **问题45：**类什么时候被初始化？  1.创建类的实例，也就是 new 一个对象  2.访问某个类或接口的静态变量，或者对该静态变量赋值  3.调用类的静态方法  4.反射（Class.forName("com.lyj.load")）  5.初始化一个类的子类（会首先初始化子类的父类）  6.JVM启动时标明的启动类，即文件名和类名相同的那个类  **问题46：**描述一下JVM加载class  JVM中类的装载是由类加载器（ClassLoader）和它的子类来实现的，Java 中的类加载器是一个重要的Java运行时系统组件，它负责在运行时查找和装入类文件中的类。  由于Java的跨平台性，经过编译的Java源程序并不是一个可执行程序，而是一个或多个类文件。当Java程序需要使用某个类时,JVM 会确保这个类已经被加载、连接（验证、准备和解析）和初始化。类的加载是指把类的.class文件中的数据读入到内存中，通常是创建一个字节数组读入.class 文件，然后产生与所加载类对应的 Class 对象。加 载完成后，Class 对象还不完整，所以此时的类还不可用。当类被加载后就进入连接阶段，这一阶段包括验证、准备（为静态变量分配内存并设置默认的初始值）和解析（将符号引用替换为直接引用）三个步骤。最后JVM对类进行初始化，包括：如果类存在直接的父类并且这个类还没有被初始化，那么就先初始化父类；  如果类中存在初始化语句，就依次执行这些初始化语句。  **问题47：**获得一个类对象有哪些方式？  1.类型.class  2.对象.getClass()  3.Class.forName()  **问题48：**Java 中为什么会有 GC 机制呢?    安全性考虑；  减少内存泄露；  减少程序员工作量。  **问题49：**Java的类加载器的种类都有哪些？  1.根类加载器  2.扩展类加载器  3.系统(应用)类加载器  4.自定义加载器 **问题50：**WebService和httpcllient 区别 HttpClient对认证机制、提供了全面的支持。用户认证过程需要一组用户名/密码进行认证用户身份。HttpClient附带多个AuthScheme的实现。AuthScheme接口表示一个抽象的面向挑战-应答的认证机制。解析和处理目标服务器发送的挑战并且对受保护资源的请求做出应答。  WebService一般通过iis进行匿名调用。这种方式不安全。我们可以使用CXF的拦截器通过自定义的Interceptor，可以改变请求和响应的一些消息处理，其中最基本的原理还是一个动态代理。设置拦截器。来验证用户的账号和密码。  WebService是完全基于XML的，HttpClient是基于HTTP协议的；WebService传输的是对象，HttpClient传输的是JSON格式的数据； |
| **人资问题(至少30个)** |
| 1.请做一下自我介绍？  回答：您好，我叫刘长根，我来自河南省，今天来贵公司应聘的是Java工程师职位，我本科学历，大学期间的专业是计算机科学，我有三年的工作经验，毕业以后一直从事JAVA开发工作，我在上一家公司做的是JAVA开发工程师，主要工作职责:是完成软件的设计、开发、测试、修改bug等工作，包括业务需求的沟通，功能模块详细设计，业务功能实现与单元测试，系统维护。我一共做过三个项目，这些项目涉及了广告类,金融类等，使用的工具是STS，我现在对市面上的分布式框架、springMVC框架和微服务都能够熟悉掌握。我本人性格沉稳，沟通能力较强，在工作上能够认真负责，勇于担当，喜欢不断地挑战自我，突破自我，愿意接受新鲜事物和环境。很早就已经听说贵公司的实力和技术在业界都是先进的，正好看见贵公司在招聘JAVA工程师，所以我就投了简历来进行面试了，同时也希望能面试成功，能给公司带来更好的发展。  2.你对我们公司有什么了解？  回答：首先得提前对公司进行了解，了解公司的文化，员工制度，公司的发展方向和主做的项目等等。面试官看的只是你的态度，是否提前对公司进行了解。  3.你为什么要为我们工作？  回答：因为公司的信誉在行业内是最好的，公司的技术和产品是行业领先的，公司符合我的价值观，依据我的工作经验我想为公司创造一定的价值；同时我个人希望更好地发展。  4.你的同事如何评价你？  回答：1.我的同事经常会说我是一个很有团队精神的人，会在适当的时候给出正确的建议，与同事相处的很融洽，他们还评价我有很好的调节压力的能力。  2.同事和客户评价我具有很强专业工作能力，善于团队合作，还富有幽默感，总是可以把最坏的情况转变成一个有意思的学习历程。他们说我做事认真，效率很高而且有条理。是个值得信任的人。  3.我的朋友和同事都说我是个值得信任的人，因为所有人有事情都会给我打电话。只要我的朋友遇上什么问题，我这个朋友就会给他们提供中肯的建议，或者待在他们身边力所能及的去帮助他们。  4.你为什么从上家公司离职？  回答：原来的公司在人员框架上在近几年内没有调整的可能，所以想寻求更大的发展接受更多的挑战学习更多的知识。  5.最近一年，你做了哪些事来提高自己？  回答：我最近一年在空闲时间会学习自己不足的地方和行业最新的技术来提高自己的能力。  6.你对公司的价值?  回答：我从事java开发已经有3年的工作经验，可以自己单独开发公司的项目模块，另外我在业余时间学习行业最新的技术，同时我有对工作认真负责的态度，积极的投入工作中，提高工作效率，为公司创造更大的价值。  7.谈谈你在压力下工作的能力？  回答：我喜欢挑战未知的事物，没有压力就会失去工作的动力，压力更能激发我的工作的热情，提高我的工作效率。  8.你有什么问题想问？  回答：我想问一下公司的新员工的培训是如何培训的？还有公司的员工制度是怎么样的？如果我被贵公司录取了，我需要提前准备什么和学习的？  9.你对同事哪些行为是无法容忍的？  回答：我跟我的同时平时相处的很好，跟他们在一起感到很开心，没有什么行为让我无法容忍的。  10.谈谈你与领导之间存在的问题？  回答：我跟领导之间不存在问题，虽然平时我们会在项目上有一些分歧，但是我们通过很好的沟通很快就达成了一致。  11.你希望自己在团队中处于什么位置？  回答：一个团队中每个人都有自己的位置，无论处在那个位置，只要可以发挥出自己的能力，发挥团队的最大的能量，平衡自己的内心，清楚自己想要什么就可以了。  12.我们为什么要录用你？  回答：我符合贵公司的招聘条件，凭我目前掌握的技能、高度的责任感和良好的饿适应能力及学习能力，完全能胜任这份工作。我十分希望能为贵公司服务，如果贵公司给我这个机会，我一定能成为贵公司的栋梁！  13.如果我录用你，你将怎样开展工作？  回答：首先听取领导的指示和要求，然后就有关情况进行了解和熟悉，接下来制定一份近期的工作计划并报领导批准，最后根据计划开展工作。  14.你希望与什么样的上级共事？  回答：做为刚步入社会新人，我应该多要求自己尽快熟悉环境、适应环境，而不应该对环境提出什么要求，只要能发挥我的专长就可以了。  15.请谈谈你个人的最大特色？  回答：我的坚持度很高，事情没有做到一个令人满意的结果，绝不罢手。  16.你找工作时，最重要的考虑因素是什么？  回答：.工作的性质是否能让我发挥所长，并不断成长。  17.你希望通过这份工作获得什么？  回答：对我来说，最重要的是自己所做的工作是否适合我。我的意思是说，这份工作应该能让我发挥专长——这会给我带来一种满足感。我还希望所做的工作能够对我目前的技能水平形成挑战，从而能促使我提升自己。  18.给你一个任务，你会怎么做？  回答：我首先会站在领导的角度去考虑这个问题，不管是否想的通，我都会向领导交流学习，了解领导为什么这样做，也许是自己考虑不周全等等；第二、弄清楚后，再阐述自己的想法，后续会怎么做，手段是什么样的。阐述清楚个人观点 和意见，我想这不仅是对我自己负责，也是对我的工作负责；第三、做这件事后会有什么样的风险？如果达到了预期目标，后续怎么继续维持下去，如果达不到，有应该用什么方式进行弥补？如何调整？  19.怎么应对自己的新工作？  回答：1.首先要保持认真的态度面对这份工作；  2.保持良好的符合岗位角色的外在形象；  3.能够以专业的态度和技术标准要求自己，努力做好每一件事情；  4.搞好人际关系，新工作是陌生的，新同事也是陌生的，与同事和领导友好相处；  5.以主人翁的状态来应对工作中的事情；  6.保持耐心，给自己一定的时间来适应。  20.保持耐心，给自己一定的时间来适应？  回答：贵公司是国际上有名的IT行业公司，我一直留意和关心贵公司的发展，特别是贵公司注重对员工的培训，更让我心动，另外像贵公司这样大的企业，我想是需要各种专业人才的，所以我来应聘。  21.你对加班的看法？  回答：如果是工作需要我会义不容辞加班，我现在单身，没有任何家庭负担，可以全身心的投入工作。但同时，我也会提高工作效率，减少不必要的加班。  22.在五年的时间内，你的职业规划？  回答：1.我将尽快熟悉本岗位的职责，业务流程，适应新的工作环境。积极主动熟悉领导和同事，向领导求教，向同事求帮，争取在短期内胜任自己的本职工作；  2.进一步丰富自己的专业知识，进一步熟悉工作环境，熟悉行业背景，培养自己的人际关系，同公司文化融为一体。进一步加强理论学习，以理论指导实践，实现专业水平和解决实际问题的能力更上个台阶；  3.将形成风格，回报公司。我将在技术领域有所作为，进一步拓宽视野，多领域借鉴，掌握更多技能，全方面提升自己。成为本领域内的专家，成为公司的骨干力量。同时，我将对新进员工，新同事进行传、帮、带。  24.你欣赏哪种性格的人?  回答提示：诚实、不死板而且容易相处的人、有“实际行动”的人。  25.你通常如何对待别人的批评?  回答提示：①沈默是金，不必说什么，否则情况更糟，不过我会接受建设性的批评。②我会等大家冷静下来再讨论。  26.怎样对待自己的失败?  回答提示：我会以乐观的心态去寻找失败的原因，避免下一次再犯同样的错误，积极的面对失败，努力去实现成功。  27.你为什么愿意到我们公司来工作?  回答提示：对于这个问题，你要格外小心，如果你已经对该单位作了研究，你可以回答一些详细的原因，像“公司本身的高技术开发环境很吸引我。”、“我同公司出生在同样的时代，我希望能够进入一家与我共同成长的公司。”、“你们公司一直都稳定发展，在近几年来在市场上很有竞争力。”、“我认为贵公司能够给我提供一个与众不同的发展道路。”这都显示出你已经做了一些调查，也说明你对自己的未来有了较为具体的远景规划。  28.如果通过这次面试我们单位录用了你，但工作一段时间却发现你根本不适合这个职位，你怎么办？  回答：我会不断学习，虚心向领导和同事学习业务知识和处事经验，了解这个职业的精神内涵和职业要求，力争减少差距。  29.如果你的工作出现失误，给本公司造成经济损失，你认为该怎么办？  回答：1.我本意是为公司努力工作，如果造成经济损失，我认为首要的问题是想方设法去弥补或挽回经济损失。如果我无能力负责，希望单位帮助解决；  2.是我的责任，我甘愿受罚；如果是一个我负责的团队中别人的失误，安慰同事并且帮助同事查找原因总结经验；  3.总结经验教训，一个人的一生不可能不犯错误，重要的是能从自己的或者是别人的错误中吸取经验教训，并在今后的工作中避免发生同类的错误。检讨自己的工作方法、分析问题的深度和力度是否不够，以致出现了本可以避免的错误。  30.谈谈你对跳槽的看法？  回答：正常的“跳槽”能促进人才合理流动，找到适合自己的工作，更能促进个人的发展  和能力。如果频繁的跳槽不利于自身的发展和完善自己的技能知识。 |
|  |