1.ArrayList,Vector和LinkedList区别

ArrayList和Vector底层是数组,都是顺序结构,是基于数组实现的，而数组是一块连续的内存空间，如果在数组的任意位置插入,删除元素，

必然导致在该位置后的所有元素需要重新排列，ArrayList的每一次有效的更改操作(插入,删除)后，

都要进行数组的重组。因此，其效率相对会比较低.但是基于数组结构查询效率高

ArrayList和Vector初始化数组长度为10, 当输出容量满足不了需求的时候

ArrayList按照1.5倍输出长度扩容

Vector按照2倍大小进行扩容

因此合理的数组大小有助于减少数组扩容的次数，从而提高系统性能

Vector提供的操作都使用synchronized修饰, 是线程安全的, ArrayList不是线程安全的

LinkedList底层使用双向链表数据结构,删除插入效率高,查询效率慢

ArrayList不是线程安全的,解决方法可以用collection的一个集合类Collections.synchronizedList()

如List<String> list = Collections.synchronizedList(new ArrayList<String>)

ArrayList,Vector,LinkedList都是有序可以重复的集合, 两者初始化长度都是10

如果扩容ArrayList是1.5倍 Vector是2倍

ArrayList,Vector底层是数组,ArrayList不是线程安全,Vector是线程安全的

LinkedList是双向链表

Set是无序不可重复的集合(添加顺序和存储顺序是有可能是不一样的) 底层是HashSet

HashSet

\* 1) HashSet底层是HashMap, HashSet的无参构造给hashMap进行了初始化

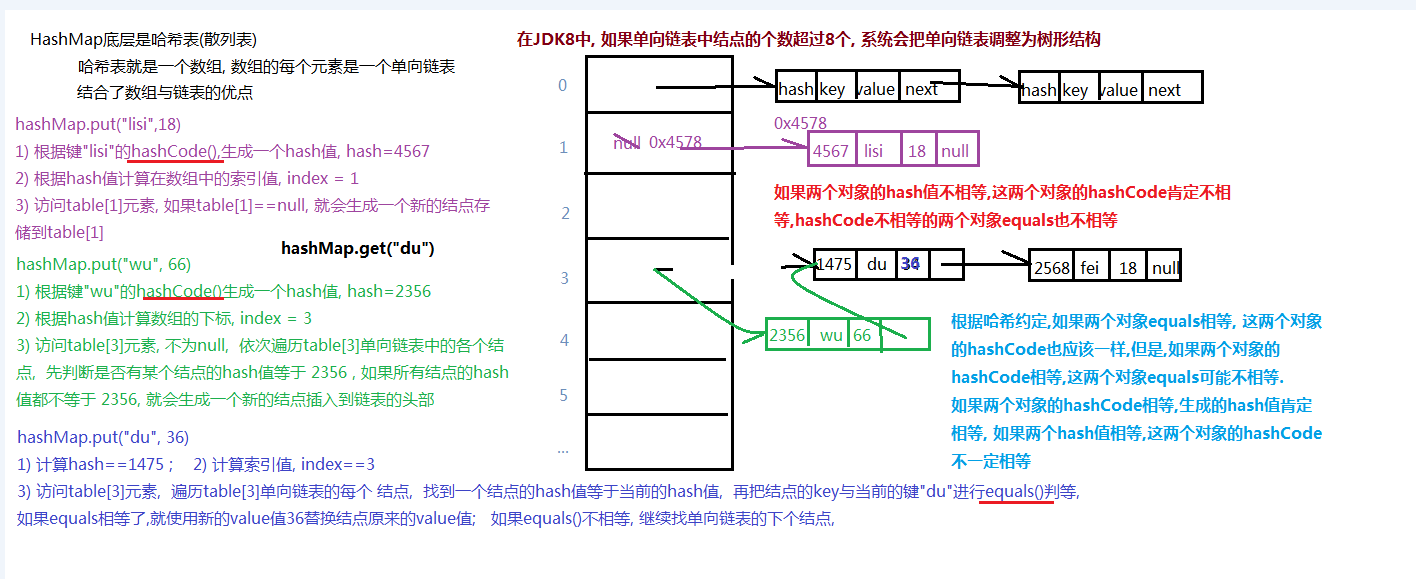
\* 2) 向hashSet中通过add(e)添加一个元素, 实际上是把这个元素作为 键,添加到了底层的hashMap中

\* map.put(e, PRESENT)

\* 3) HashSet实际上是就是底层HashMap的键 的集合

\* 4) 因为Map中的键不允许重复,所以HashSet也不能存储重复的元素

**HashMap底层实现原理**



**2.什么是redis?有什么特点?(基础知识)**

Redis是Remote Dictionary Server(远程数据服务)的缩写(这个不用说)

redis是一个key-value存储结构的存储系统,key是字符串类型的,value有多种类型,

可以是string,list,map,set和zset类型的数据,string是字符串,list是链表,set是集合,zset是有序集合,map也是集合

Zset是有序的

redis是基于内存运行的(存储数据) 是从内存中拿数据

从redis中获取数据要比直接操作数据块要快,因为redis是从缓存中获取数据,也就是从内从中获取数据,而数据库的数据是存储在磁盘上的,要先把磁盘上的数据加载到内存才能获取数据,而这个过程是非常耗时的

而数据库是磁盘 是从磁盘上拿数据 而从磁盘上拿数据 要先把磁盘上的数据加载到内存中,然后再从内存中取数据

从磁盘加载到内存中是非常耗时的,当处理海量数据时,直接操作数据库,数据库会收不了,而且效率非常低,用户体验也差(半天没有响应)

(之所以从缓存中拿数据会快，是因为缓存的数据存在于内存中，不像mysql的数据是存在磁盘上的，

即不用经过从磁盘加载到内存这个过程（这个过程是非常耗时和低效的），直接从内存获取数据。

**Redis**

为什么用redis?实际开发中是怎么用的?

每秒海量的读写请求,磁盘io无法满足,数据库中海量的数据,读写速度非常慢,

无法处理高并发请求,redis用于处理高并发情况下数据请求的

数据库比如mysql也有缓存机制?难道不行吗,非得用redis缓存?

关系型数据库（例如MySQL）使用查询缓存。这种查询缓存在更新数据后，缓存就是失效了。在频繁的数据读写交互应用中。 缓存的性能不高。

首先，对大部分分布式系统来说，由于负载均衡的关系，缓存不适合用程序实现，多台机器上的这些缓存不便于共享，所以不如Redis集中式的缓存来得方便合高效。

其次，很多大型互联网公司的经验表明，数据库缓存带来的效率提升并不明显。因为更新频繁的表，缓存命中率很低；而指定数据库对哪些表用缓存、哪些表不用缓存，难度很大。

实际开发中我们把一些热点数据(经常访问的数据) 存放到redis缓存中 加快读取速度

虽然redis是内存存储,但是redis也支持持久化,

提供两种方案

**方式1:RDB方式,**也就是快照方式,在redis.conf中配置

a.快照生成策略,

配置格式：save <seconds> <changes>对 Redis 进行设置， 让它在“ N 秒内数据集至少有 M 个key改动”这一条件被满足时， 自动保存一次数数据。

b.配置保存文件的名字

c.目录

这种方式

优点:因为存储的是快照文件,恢复方便,恢复数据速度快

缺点:会有数据丢失,无法存储保存快照后,最后一次更改的数据.因为要竟让保存文件所有会有磁盘Io操作,如果数据量非常庞大会影响redis性能

**方式2:AOF方式,**redis每接收一条改变数据的命令,读取操作是不记录的,是就把操作命令写到一个AOF文件中去,如果宕机,redis会重新执行AOF文件的命令恢复数据

优点:不会丢失数据

实际开发中我们是多台redis集群部署,一台主机两台从机,主机负责写数据,从机负责读数据,我们采用的是哨兵模式,哨兵模式的好处就是,可以自动故障转移,如果redis主机宕机,我们不用手动指定主机器,哨兵模式中会一直监控主机,当主机宕机会自动从从机器中选取一台做主机,其他从机器自动挂靠在新的主机下,()哨兵模式利用的是选取机制)

**先把主从配置配置好**

**主配置文件**

编辑Master的配置文件redis6380.conf : 在空文件加入如下内容(实际中不用这样,因为一台服务器一个redis)

include /usr/local/redis-3.2.9/redis.conf

daemonize yes(是否后台启动)

port 6380(自定义当前redis端口号)

pidfile /var/run/redis\_6380.pid(自定义的文件，表示当前程序的pid ,进程id)

logfile 6380.log(持久化的rdb文件名)

dbfilename dump6380.rdb(持久化的rdb文件名)

**从配置文件**

**编辑Slave的配置文件redis6382.conf 和redis6384.conf: 在空文件加入如下内容**

**①：redis6382.conf：**

**include /usr/local/redis-3.2.9/redis.conf**

**daemonize yes**

**port 6382**

**pidfile /var/run/redis\_6382.pid**

**logfile 6382.log**

**dbfilename dump6382.rdb**

**slaveof 127.0.0.1 6380(表明是哪个主的从)**

**手动容灾**

**①：slaveof no one，将一台slave服务器提升为Master （提升某slave为master）**

**②：slaveof 127.0.0.1 6381 （将slave挂至新的master上）**

**以上是主从配置,然后配置哨兵模式(自动容灾)**

**Redis的src目录下有 sentinel.conf文件 ,配置port改成当前redis的端口号,**

**修改 sentinel monitor mymaster 127.0.0.1 6380 2**

**解读sentinel monitor 给主机起名字 主机ip 主机端口号 投票数**

**然后先启动所有redis 再启动哨兵就可以了**

**1）Sentinel会不断检查Master和Slave是否正常**

**2）如果Sentinel挂了，就无法监控，所以需要多个哨兵，组成Sentinel网络，一个健康的Sentinel至少有3个Sentinel应用。 彼此在独立的物理机器或虚拟机。**

**3）监控同一个Master的Sentinel会自动连接，组成一个分布式的Sentinel网络，互相通信并交换彼此关于被监控服务器的信息**

**4）当一个Sentinel认为被监控的服务器已经下线时，它会向网络中的其它Sentinel进行确认，判断该服务器是否真的已经下线**

**5）如果下线的服务器为主服务器，那么Sentinel网络将对下线主服务器进行自动故障转移，通过将下线主服务器的某个从服务器提升为新的主服务器，并让其从服务器转移到新的主服务器下，以此来让系统重新回到正常状态**

**6）下线的旧主服务器重新上线，Sentinel会让它成为从，挂到新的主服务器下**

**总结**

**主从复制，解决了读请求的分担，从节点下线，会使得读请求能力有所下降，Master下线，写请求无法执行**

**Sentinel会在Master下线后自动执行故障转移操作，提升一台Slave为Master，并让其它Slave成为新Master的Slave**

1当redis缓存崩溃的时候，那么不是海量的请求都去访问数据库了？数据库能抗住吗？

**Redis面试回答:**

Redis是用来处理高并发数据请求的,缓解数据库压力.redis是基于内存的一个存储系统,是key-value形式存储数据,key是string类型,value支持五中类型,string,list,map,set,zset等,在数据库中存储数据,必须先定义好字段,而redis中想要存什么数据直接存就可以了,

为什么用redis处理高并发呢?因为如果大量请求直接到数据库上,数据库是承受不了的,数据库读取数据设计磁盘io操作,是非常耗时的,而redis是基于缓存的,读写非常快,所以我们将访问量非常大的数据提前存到redis中,应对高并发访问,虽然redis是基于内存存储的,但是redis也可以将数据持久化到磁盘上,redis提供了两种方式,一种是rdb方式也就是快照形式,我们可以配置redis多久存储一次,但是这种方式会造成数据丢失,因为如果发生意外,最后一次数据是会丢失的,还有一种方式就是aof方式,他是记录每一次的写操作,并写入日志文件,然后如果发生意外会按照文件上的每一步操作重新执行一边,这种方式比较安全,但是回复数据比较慢.

实际开发中我们redis是集群部署,redis支持一主多从,读写分离进一步提高服务能力,同时redis中的哨兵模式可以自动转移故障,应对宕机问题.让服务器处于不间断工作状态.

**数据库也有缓存为什么不用?**

1. 数据库的缓存效率不高,2.现在的项目都是分布式的,部署在不同的及其上,不同机器上的缓存不方便共享,不如redis缓存方便3.数据库中表很多,有的表更新频繁,有些表基本上不怎么更新,那些表用缓存那些表不用缓存不好指定,而且一些大公司的经验结论是数据库缓存对数据库性能提升并不大

**Redis缓存你们是什么用的?**

请求数据的时候先到redis缓存中去查数据,如果有数据就返回该数据,如果没有数据就去数据库中查询,然后将数据存放到redis中,在将数据返回

**Redis缓存穿透和缓存雪崩是什么?是怎么处理的?**

首先缓存穿透是指在某个时间点,大量请求涌入redis中获取数据,这个数据在redis中不存在,那么大量请求会到数据库中造成数据库崩溃.(这个可以不用说)

**Redis缓存穿透**

缓存穿透是指查询一个一定不存在的数据，由于缓存是不命中时需要从数据库查询，查不到数据则不写入缓存，这将导致这个不存在的数据每次请求都要到数据库去查询，造成缓存穿透。

解决办法(布隆过滤器,返回Null值到redis,加锁(效率低,基本不用))

使用布隆过滤器,判断这个可以是否存在,如果不存在就进行拦截,防止对数据库造成压力

也可以采用一个更为简单粗暴的方法，如果一个查询返回的数据为空（不管是数 据不存在，还是系统故障），我们仍然把这个空结果进行缓存，但它的过期时间会很短，最长不超过五分钟。

或者使用锁,但是这种做法效率低,对吞吐量没有什么帮助

Redis雪崩?

缓存雪崩是指在我们设置缓存时采用了相同的过期时间，导致缓存在某一时刻同时失效，请求全部转发到DB，DB瞬时压力过重雪崩。  
解决办法：在缓存失效后，通过加锁或者队列来控制读数据库写缓存的线程数量。比如对某个key只允许一个线程查询数据和写缓存，其他线程等待。 可以通过缓存reload机制，预先去更新缓存，再即将发生大并发访问前手动触发加载缓存 不同的key，设置不同的过期时间，让缓存失效的时间点尽量均匀 做二级缓存，或者双缓存策略。A1为原始缓存，A2为拷贝缓存，A1失效时，可以访问A2，A1缓存失效时间设置为短期，A2设置为长期。①用锁/分布式锁或者队列串行访问

 ②缓存失效时间均匀分布

开发中遇到了什么问题?怎么解决的?

项目中用了redis,连接池总是溢出?后来查看日志发现请求中携带了缓存中不存在的key,我们用了布隆过滤器来结局穿透问题

Sprin

**Nginx**

Nginx是什么?怎么用的?为什么用?

nginx是一个HTTP和反向代理服务器，我们用nginx实现负载均衡，对外nginx暴露自己的IP与端口，在接收到用户的请求之后通过反向代理和轮询或者权重等机制，将请求发送给服务器集群。

我们在使用集群服务的时候使用SpringSession，redis和mysql集群来解决集群之间的数据同步问题。

springsession是spring旗下的一个项目，把httpsession替换为springsession，用于解决集群下的session共享问题。

springsession在默认情况下是将session保存在第三方的容器中，比如redis。

springsession的底层实现是：使用拦截器拦截用户产生的session，通过spring-data-redis将session存入redis缓存中，然后每次请求的时候都将去redis中寻找有没有相同点session，有的话证明之前登陆过，可以拿到session的信息。

**Dubbo**

Dubbo是一个分布式服务框架,用来远程调用服务的

Dubbo包括三个核心:**远程调用,集群容错,自动发现**

Dubbo可以做什么,其实就是围绕上边三个核心来说说

Dubbo可以实现远程调用,调用远程服务端接口就像调用本地接口一样,而且dubbo可以提供软负载平衡,就是通过程序来实现负载平衡,相对于硬件负载平衡,降低了成本,可以自动注册于发现,在调用的时候不用再代码中写死地址,dubbo无缝对接spring,利用spring加载dubbod配置

面试回答:

早期是单一服务架构,所有的功能都部署在一个应用上,然后集群部署就可以满足要求,(成本低,容易开发)网站流量小没什么问题,但是随着访问量的增加,业务越来越多,功能越来越多,单一架构无法满足要求就,而且单一架构,后期维护和拓展起来都特别麻烦,所以我们把一些核心业务抽取出来,作为独立的服务,这就是分布式架构,当服务越来越多,服务之间的引用关系就会变得错综复杂,过多的URL不易管理,启动关系不清晰非常不容易管理,而且不同服务之间的访问量也是不同的,还要进行合理的资源配置,这时候就要考虑考虑应用之间的调用管理和动态资源配置了,而dubbo就可以解决这些问题,dubbo可以管理远程调用,调用远程服务就像调用本地的一样,只需要进行简单的配置,同时配合zookeeper注册中心实现服务端的自动注册于发现,实现服务的动态管理,无需写死服务端地址,而且dubbo可以实现软负载平衡,可以降低成本

Dubbo底层是通过动态代理来实现的,dubbo框架通过创建远程服务端接口的代理对象,来调用远程服务的方法

再问httpclient也可以做到调用远程服务,为什么非要使用dubbo?

Httpclient是可以,但是上边说了,不同服务之间也会调用,调用关系复杂需要管理,而且httpclient不能做负载均衡,dubbo其实就是在httpclient的基础上加了一些高级特性,比如”服务发现”,”错误重试”,而且针对可用性和效率都进行了优化,而单纯的htpclient没有这些特性

Zookeeper和dubbo是什么关系?

对于服务提供方需要发布服务,由于应用的复杂性,服务提供方有很多,而且有的应用既是服务发布者又是服务消费者,就需要一个系统来统筹管理这些调用,zookeeper就是做这个的,通过将服务统一管理起来，可以有效地优化内部应用对服务发布的流程和管理。Zookeeper就像是一个资源管理器,管理所有的资源,服务提供者在zookeeper中注册,消费者到zookeeper中调用服务,消费者需要的服务地址不用写死,而且zookeeper的心跳机制可以实时监控服务提供者的状态,不能用的服务提供者会在注册中心删除,服务消费者不会调用不能使用的提供者

SOA说的是面向服务的架构,将业务封装成接口,实现解耦合,并独立部署,提供服务,强调的是异构系统之间的网络通信

**微服务是:**(微服务就是按照业务模块进行划分,独立部署,提供服务,不同模块之间在网络中进行通信)将一个应用编程一套小型应用开发的方法,每种应用都在自己独立的进程中运行,不同应用之间相互通信,这些应用都是按照业务进行划分的,不同模块可以采用不同的技术

**微服务的好处是:我可以灵活配置资源,**因为是根据业务独立部署的,比如有的应用访问量高,有的访问量低,那么访问量高的我就可以多部署几台机器,而访问量低的我就少部署几台机器,**资源进行隔离,**每个模块都运行在自己的进程中,有自己的数据源,如果想要访问别的应用的数据库只能通过这个应用暴露的接口,调用数据库,有效避免了不同应用之间竞争数据库,CPU资源

SOA与微服务区别,SOA强调的是异构系统之间的通信,而微服务讲的是将应用按照业务进行拆分,分布式独立部署.

微服务优点:独立部署,技术多样性

微服务缺点:因为拆分成许多模块,**开发和测试复杂,不同系统之间的数据一致性问题**

**幂等性：就是用户对于同一操作发起的一次请求或者多次请求的结果是一致的，不会因为多次点击而产生了副作用。**举个最简单的例子，那就是支付，用户购买商品使用约支付，支付扣款成功，但是返回结果的时候网络异常，此时钱已经扣了，用户再次点击按钮，此时会进行第二次扣款，返回结果成功，用户查询余额返发现多扣钱了，流水记录也变成了两条．．．

**mybatis：**

mybatis是一个优秀的持久层框架，它内务封装了JDBC,开发者只需要关注SQL语句本身，而不需要花精力去处理驱动，创建链接等范泽的操作。所以mybatis，只写一个映射文件，和dao接口就可以执行，不需要实现类就是因为mybatis自己拥有每一个实体类的XML文件，在这里面引用dao类和实体类，这就相当于dao的实现类了。

mybatis是基于动态代理实现的，所以mybatis的设计模式就是代理模式。

mybatis的原理：

mybatis应用程序根据xml配置文件创建sqlsessionfactory,

sqlsessionfactory根据配置获取sqlsession,

sqlsession包含了所有执行sql的方法，可以根据sqlsession实例直接运行映射的sql语句，完成增删改查和事务的提交

使用之后关闭sqlsession

ActiveMQ

**低耦合**，不管是程序还是模块之间，使用消息中间件进行间接通信。

**消息的顺序性**，消息队列可以保证消息的先进先出。

**消息可靠传输**，持久化的存储使得消息只有在被消费之后才会删除。

**异步通信能力，**相对于RPC来说，异步通信使得生产者和消费者得以充分执行自己的逻辑而无需等待。

**缓冲能力，**消息中间件像是一个巨大的蓄水池，将高峰期大量的请求存储下来慢慢交给后台进行处理，对于秒杀业务来说尤为重要。(削峰)

为了避免单点故障,activemq采用集群配置

**String,Stringbuilder,Stringbuffer**

1. 首先说运行速度，或者说是执行速度，在这方面运行速度快慢为：StringBuilder > StringBuffer > String

如果一个StringBuffer对象在字符串缓冲区被多个线程使用时，StringBuffer中很多方法可以带有synchronized关键字，所以可以保证线程是安全的，但StringBuilder的方法则没有该关键字，所以不能保证线程安全，有可能会出现一些错误的操作。所以如果要进行的操作是多线程的，那么就要使用StringBuffer，但是在单线程的情况下，还是建议使用速度比较快的StringBuilder。

String类：

字符串：就是由多个字符组成的一串数据，也可以看成是一个字符数组，通过查看API可知： （1）字符串字面值（如：“abc”）可以看成是一个字符串对象； （2）字符串是常量，一旦被赋值就不能被改变，这也就是为什么说：String类为什么是final的原因。

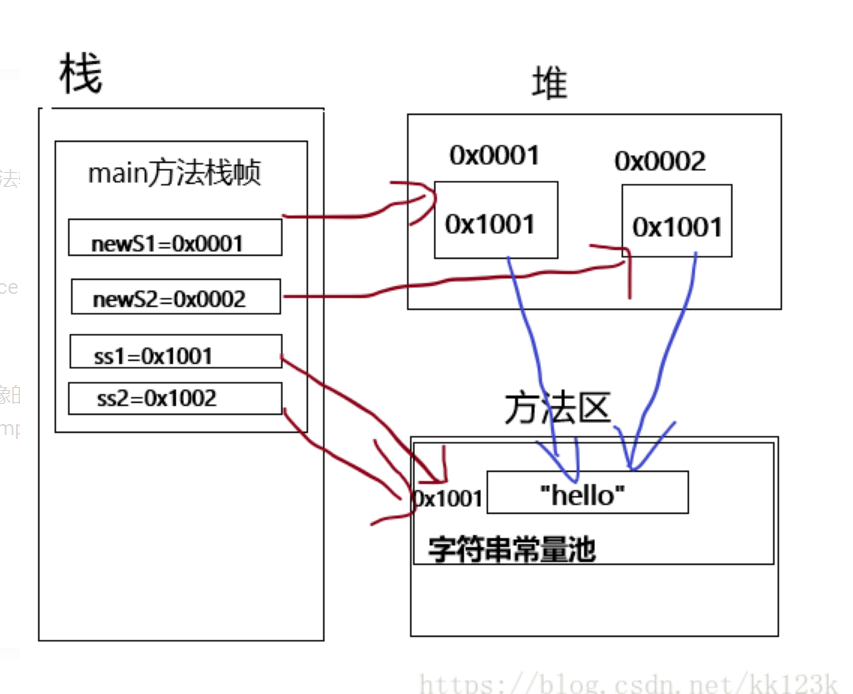
String类的特点：

A:字符串直接赋值的方式是先到字符串常量池里面去找，如果有就直接返回；如果没有，就创建并返回。

B:一旦被赋值，就不能被改变。值不能变。 StringBuffer： 线程安全的可变字节。

面试题

1:String s = new String("hello")和String s = "hello"有区别吗? 有区别。前者会创建两个对象，后者创建一个对象。（分析请看String类的特点介绍）



String newS1 = new String("hello")首先会在堆中创建一块内存， 内存地址返回给栈中newS1，然后因为"hello"是常量，JVM会去方法区中的字符串常量池查看是否有"hello"字符串的对象，没有的话就分配一个空间来存放"hello"，并将其空间地址存入堆中new出来的对象中； 在新建newS2对象时先去方法区的字符串常量池中寻找"hello"字符串对象，发现已经有了，就直接将其内存地址存入堆中new出来的对象中，同理堆中new出来的对象的内存地址返回给newS2 对于String ss1="hello"这样创建的对象，JVM会直接检查字符串常量池是否已有"hello"字符串对象，如没有，就分配一个内存存放"hello"，如有了，则直接将字符串常量池中的地址返回给栈中的ss1而不经过堆中new的对象 所以ss1和ss2指向的是同一个对象 这个时候有人就会有疑问了，既然ss1和ss2指向的是同一个对象，那我修过ss1的值岂不是也同时改变了ss2的值？ 显然，改变ss1的值并不会影响ss2以及其它对象的值，原因其实Java的String是一个不可变的类，假设我们对ss1重新赋值时其实是在字符串常量池中重新分配了一块内存然后令ss1指向新对象

2:有关此处看程序写结果的问题？

答：（1）字符串如果是变量相加，先开空间，再拼接； （2）字符串如果是常量相加，先加，之后在变量池找，如果有就直接返回，没有，就创建。

3:StringBuffer和String区别？

答：StringBuffer长度和内容可变，String内容和长度不可变。如果使用StringBuffer做字符串的拼接，不会浪费太多的资源。

4:String、StringBuffer、StringBuilder的区别？

1. String是内容不可变的，而StringBuffer、StringBuilder都是内容可变的。

（2）StringBuffer是同步的，数据安全的，但是效率低； StringBuilder是不同步的，数据不安全，效率低。

5:StringBuffer和数组的区别？

二者都是可以看成是一个容器，装其他的数据，但StringBuffer的数据最终是一个字符串数据；而数组可以放置多种数据，但必须是同一种数据类型的。

**如何解决多线程安全问题?**

多线程情况下,有共享数据,并且对这个数据进行修改的时候会出现线程

安全问题.比如实际中对数据库进行修改操作的时候就要考虑线程安全

1.尽量使用局部变量,局部变量是在当前线程中的变量

2.可以使用成员变量,此时让java对象是多例的,让每个线程是一个java对象

3.进行同步操作,优先使用同步代码块(轻量级),同步方法是重量级的(效率低)

**Servlet和Jsp的区别是什么？**

**什么叫做生命周期,表示一个类的对象,从创建到销毁的全过程.**

**servlet的生命周期,表示的就是servlet对象从创建到销毁的全过程.**

**当服务器启动,servlet对象是不创建的.**

**当浏览器发出的请求,用到了指定servlet的时候,服务器会自动的为我们创建出来一个servlet对象.**

**servlet对象创建完毕后,马上调用init方法执行初始化操作.**

**servlet对象在第一次访问创建完毕后,浏览器发出的第2~n次请求不重新创建对象了,而是使用第一次访问时创建出来的对象.所以servlet对象是以单例形式存储在服务器中的.**

**以上单例创建为懒汉式.**

**servlet执行doGet/doPost方法来处理请求.**

**当服务器关闭,servlet对象销毁.**

**servlet对象销毁前,调用destory方法.**

**加入load-on-startup,会更改servlet对象的创建时机,从以前的用到时才创建,更改为启动服务器时创建.由原来的懒汉式更改为饿汉式.**

**补充:servlet对象使用的时候是单例对象.但其实servlet对象是一个假单例.**

**假单例:表示servlet对象我们是可以手动创建对象的.**

**真单例:我们手动是不能创建对象的**

**知道单例模式吗?饿汉式和懒汉式区别?什么时候用?手写单例模式**

编译时初始化

1. // 在自己内部定义自己一个实例，是不是很奇怪？
2. // 注意这是private 只供内部调用
3. private static Singleton instance = new Singleton();
5. // 这里提供了一个供外部访问本class的静态方法，可以直接访问
6. public static Singleton getInstance() {
7. return instance;
8. }

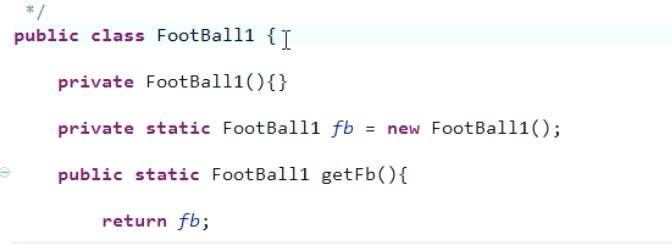
使用时实例化

1. private static Singleton instance = null;
3. public static synchronized Singleton getInstance() {
5. // 这个方法比上面有所改进，不用每次都进行生成对象，只是第一次
6. // 使用时生成实例，提高了效率！
7. if (instance == null)
8. instance = new Singleton();
9. return instance;
10. }

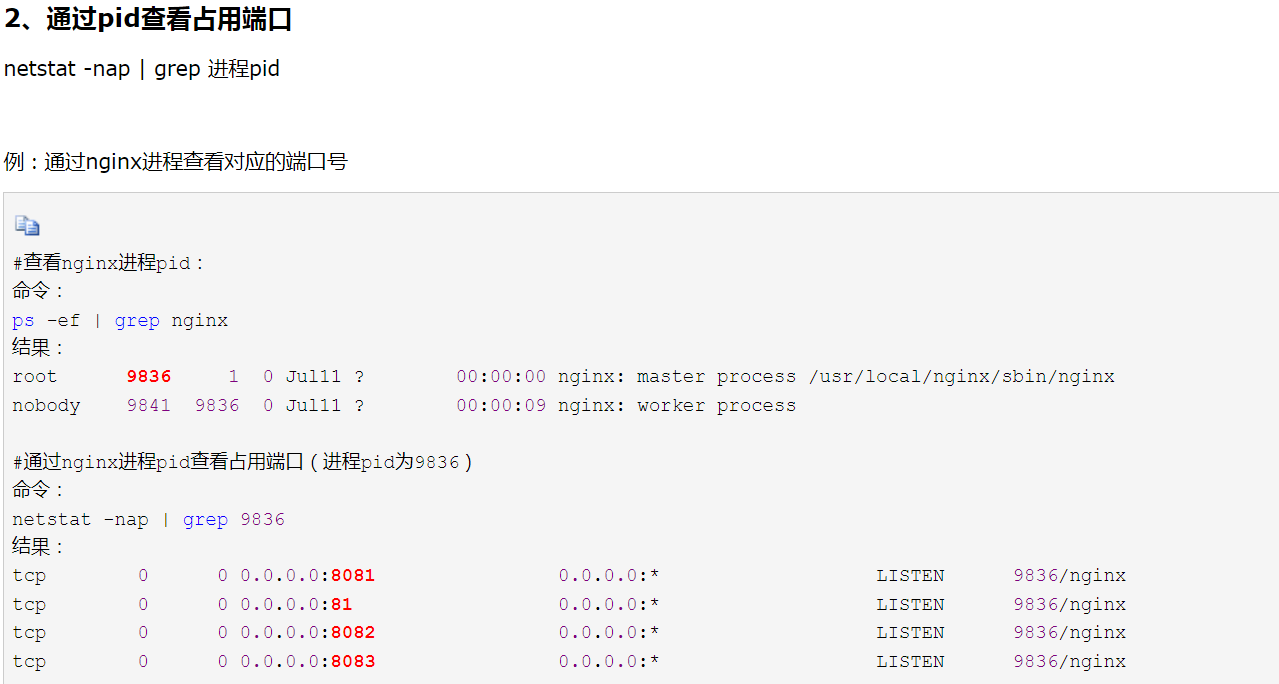
饿汉式是在编译时候进行初始化对象,而饿汉式是在用的时候才进行初始化

饿汉式会出现线程安全问题,实际开发中我们的工具类会使用单例模式

**真单例模式与假单例模式?**真单例模式是我们手动无法创建的,而假单例模式是我们可以手动创建也可以获取到单例模式



这种情况我们想获得football是没法手动创建的,因为构造方法是私有化的,而对于servlet就是假单例的,第一次接受请求的时候tomcat会通过反射创建一个servlet,但是我们我们还可以自己创建servlet所以对于servlet来说,他是一个假单例的



# 非技术面试

## 为什么选择我们公司

你们公司叫我来了X

看了下你们公司的官网，做的是XX，与我之前的有相同的业务，同样，公司的规模，和发展前景很吸引我，符合我的职业规划，让我放心。

## 朋友对你的看法

“我的朋友都说我是一个可以信赖的人。因为，我一旦答应别人的事情，就一定会做到。如果我做不到，我就不会轻易许诺；或者说“我觉的我是一个比较随和的人，与不同的人都可以友好相处。在我与人相处时，我总是能站在别人的角度考虑问题。

## 在未来的五年的时间内，你的职业规划是什么？

主要是先把提升并完善自己技术框架并丰富自己工作经验，如果5年之间我达到了项目经理水平，我会考虑往项目经理转型

## 你还有什么问题要问

公司现在在做的项目，用了什麽框架，晋升机制

## 工作后发现工作不适合你

这个现象在实际工作中很常见，为什么呢？有句老话：“不当家不知柴米油盐贵”，在你向往这份工作的时候，你往往看到的是这份工作比较好的那一面，而看不到它相反的一面，当你真正投入这份工作的时候，你会感觉理想与现实的差别，会造成你心里极大的落差，从而你会有离职的思想，所以选择一份工作的同时，你一定要明确知道你要在这份工作中得到什么，是否在你的人生规划当中，如果你在面试过程中真的被问到这道题先这样回答

1.如果你确实热爱这个职业，那你就要不断学习，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；

热爱钱

## 你对跳槽的看法

## 为什么从上家公司离职

## 了解的技术栈



## 为什么选择武汉

# 怎么判断简历有没有包装

## ****前言****

|  |
| --- |
| [今天，聊点非技术的话题，不知道是否有读者是“996”呢，人生嘛，出来奋斗总是要吃点苦的，但是身体又是革命的本钱，所以二者怎么权衡，见仁见智咯。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [今天转载一下我年少有为的前同事coody写的文章，在金三银四的时候，作为面试官跟应聘者双方的角度，有哪些值得思考的东西呢？](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) ****正文**** [在互联网极速膨胀的社会背景下，各行各业涌入互联网的IT民工日益增大。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [早在2016年，我司发布了Java、Ios工程师的招聘信息，就Java工程师单个岗位而言，日收简历近200份，Ios日收简历近一千份。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [没错，这就是当年培训机构对Ios工程师这个岗位发起的市场讨伐。而随着近几年的发展，市场供大于求现象日益严重。人员摸底成为用人单位对人才考核的重大难题。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [笔者初次与求职者以面试的形式进行沟通是2015年6月。由于当时笔者从业时间短，经验不够丰富，错过了一些优秀的求职者。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [三年后的，今天，笔者再次因公司规模扩大而深入与求职者进行沟通。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [培训机构除了提供技术培训，往往还提供简历编写指导、面试指导。很多潜移默化的东西，我们很难甄别。但培训机构包装的简历，存在千遍一律的特征。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [在甄别对方水分这一块，并没有明确的标准，但是笔者可以提几个点。这也是笔者在实际面试中惯用的做法。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) |

## ****一、初选如何鉴别劣质简历****

|  |
| --- |
| [****1、年龄较小却具备高级文凭****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [年龄较小却具备高级文凭，这个或许不能作为一项标准，但是大部分的应聘者，均符合传统文凭的市场情况。个别技术爱好者可能通过自考获得文凭，这种情况需提供独有的技术亮点。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****2、年龄较大却几乎不具备技术经验****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [年龄较大却几乎不具备技术经验，相对前一点，这个问题就比较严重了。大家都知道，一个正常的人，对新事物的接受能力会随着年龄的增长而降低，互联网技术也包括其内。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [如果一个人年龄较大不具备技术经验，那么只有两种情况：1、中途转行(通过培训、自学等方式强行入行)。2、由于能力问题，已有的经验不敢写入简历中(能力与经验/薪资不符)。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****3、项目经验多为Crm系统、商城、XX管理系统、调查问卷系统、课堂考试系统等****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [项目经验，这一项用来评估应聘者的水平太合适不过了。随着互联网的发展迭代，每一年都会出来很多创新型的互联网公司和新兴行业。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [笔者最近发布的招聘需求里面。CRM系统、商城、XX管理系统、问卷系统、课堂系统占了90%的份额。试问现在2019年，内部管理系统这么火爆么。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [言归正传，我们对于简历的评估，应当多考虑“确有其事”的项目。比如说该人员当时就职于XX公司，该公司当时的背景下确实研发了该项目（外包除外）。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****4、项目的背景不符合互联网发展背景****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [项目背景，每年的市场走向不同，从早些年的电商、彩票风波，到后来的O2O、夺宝、直播、新零售。每个系列的产品的出现，都符合市场的定义。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [如果简历中出现18年、19年才刚立项做彩票(15年政府禁止互联网彩票)、O2O、商城、夺宝(17年初禁止夺宝类产品)、直播等产品。显然是非常不符合市场需求的。这种情况下需考虑具体情况是否存在理解空间。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****5、不同工作经验下多个项目技术架构或项目结构一致，缺乏新意。****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [一般情况而言，不同的公司技术栈不同，甚至产品的走向和模式完全不同。故此，当一个应聘者多家公司的多个项目中写到的技术千遍一律，业务流程异曲同工。看似整洁，实则更加缺乏说服力。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****6、技术过于新颖，对旧技术却只字不提。****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [技术过于新颖，根据互联网技术发展的走向来看，我们在不断向新型技术靠拢。但是任何企业作为资历深厚的CTO、架构师来说。往往会选择更稳定、更成熟、学习成本更低的已有技术。对新技术的追求不会过于明显。而培训机构则是“哪项技术火我们就教哪项”。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [故此，出现了很多走入互联网行业的新人对旧技术一窍不通。甚至很多技术都没听过。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****7、工作经验较丰富，但从事的工作较低级。****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [工作经验比较丰富，单从事的工作比较低级，这里存在很大的问题，要么就是原公司没法提供合理的舞台给该人员更好的发展空间，要么就是该人员能力不够，没法完成更高级的工作。当然，还有一种情况就是该人员包装过多的经验导致简历中不和谐。这种情况需要评估公司规模和背景。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****8、公司背景跨省跨市****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [可能很多用人单位和鄙人一样，最近接受到的简历，90%为跨市跳槽的人员。其中武汉占了60%以上。均为武汉XX网络科技有限公司。公司规模均小于50人。也有厦门、宁波、南京等等。这个问题笔者就不提了，大家都懂的。跨地区跳槽不好查证。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****9、缺少业余热情于技术的证明。****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [有些眼高手低的技术员，做了几个管理系统。用到的技术确是各种分布式、集群、高并发、大数据、消息队列、搜索引擎、镜像容器、多数据库、数据中心等等。期望的薪资也高于行业标准。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [一个对技术很热情的人，业余时间肯定在技术方面花费过不少时间。那么可以从该人员的博客、git地址入手。甚至可以通过手机号、邮箱、昵称、马甲。去搜索引擎进行搜集，核实该人员是否在论坛、贴吧、开源组织有过技术背景。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) |

## [****二、进入面试阶段，如何甄别对方的水分****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)

|  |
| --- |
| [****1、通过公司规模、团队规模、人员分配是否合理、人员合作方式来判断对方是否具备工作经验。****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [当招聘初级、初中级IT人员的时候，可以询问一些问题，比如公司有多少人、产品团队多少人、产品、技术、后端、前端、客户端、UI、测试各多少人。工作中如何合作的、产品做了多少时间、何时上线的、上线后多长时间迭代一个版本、多长时间迭代一个活动、发展至今多少用户(后端)、多大并发等等(后端)。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  根据笔者的经验，如果一个人没有任何从业周期，面对这些问题的时候，或多或少答非所问或者给出的答案非常不合理。 [****2、背景公司入职时间、项目立项实现、完工时间、产品技术栈、迭代流程的核实。****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [很多应聘者对于简历过于包装，只为了追求更高的薪资。当我们问起：你是xx年xx月入职的该公司？你们项目是xx年xx月上线的？你们项目使用到xx技术？你们每次上线前夕是如何评审的。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [面对这些问题，应聘者给出的答案经常与简历不符合。这样问题就来了。关于项目使用到的技术，很多项目我们可以通过搜索该项目的地址、APP。通过http协议、技术特征、抛出异常特征来大致判别对方使用到的技术。如果应聘者给出的答案明显与之不匹配，嘿嘿。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [****3、通过技术深度，甄别对方的技术水平****](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) [笔者最近接待的面试者，很多面试者的简历上，写着层出不穷的各种技术，为了不跨越求职者的技术栈，笔者专门挑应聘者简历写到或用到的技术来进行询问。笔者举几个例子。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [某求职者简历上写着熟练使用redis。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [1、介绍一下你使用过redis的哪些数据结构，并描述一下使用的业务场景；2、介绍一下你操作redis用到的是什么插件；3、介绍一下你们使用的序列化方式；4、介绍一下你们使用redis遇到过给你印象较深的问题；](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [**某求职者声称熟练http协议并编写过爬虫。**](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [1、介绍一下你所了解的几个http head头并描述其用途；2、如果前端提交成功，后端无法接受数据，这时候你将如何排查问题；3、描述一下http基本报文结构; 4、如果服务器返回cookie，存储在响应内容里面head头的字段叫做什么; 5、当服务端返回Transer-Encoding：chunked 代表什么含义 6、是否了解分段加载并描述下其技术流程。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [当然，面向不同的技术，对应的技术深度自然也不一样。但是笔者碰到的问题是：使用git两年却不知道github、使用redis一年却不知道数据结构也不知道序列化、专业做爬虫却不懂content-type含义、使用搜索引擎技术却说不出两个分词插件、使用数据库读写分离却不知道同步延时等等。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect)  [写在最后，笔者认为在招聘途中，并不是不允许求职者包装，但是尽可能满足能筹平衡。虽然这篇文章没有完美的结尾，但是笔者提供了面试失败的各种经验。笔者最终招到了如意的小伙伴。也希望所有技术面试官早日找到符合自己产品发展的IT伙伴。](https://mp.weixin.qq.com/s?__biz=MzA3MTUzOTcxOQ==&mid=2452968813&idx=2&sn=5cbb5eb84d81779e8674c48f3784b309&scene=21#wechat_redirect) |

# 思维逻辑题

## 经过努力，甲、乙、丙、丁等4位品学兼优的中学生考上了北京大学、清华大学、南京大学、东南大学等4所名校。9月初，甲、丙与乙、丁分别奔赴两座不同的城市去上学了。已知： （1）甲没有上北京大学 （2）乙没有上清华大学 （3）丙没有上南京大学 （4）丁没有上东南大学 根据以上陈述，可以得出以下哪项？（　D　）

|  |  |
| --- | --- |
| A | 甲没有上清华大学 |
| B | 乙没有上北京大学 |
| C | 丙没有上东南大学 |
| D | 丁没有上北京大学 |

# Java基础知识梳理

## 第一次作业day01

public class day1{

public static void main(String[] args){

System.out.println("\t\t\t购物清单\n商品名称\t购买数量\t商品单价\t金额\n鳄鱼夹克\t 1 \t\t1900 \t1900\nipodMP4 \t 1 \t\t400 \t800");

}

}

# Java基础

## java之父

|  |
| --- |
|  |

## java中length和length()方法的区别

|  |
| --- |
| Java中，length方法用于获取数组的长度。  int[] arr = new int[10];System.out.println(arr.length);而length()用于获取String字符串中字符的个数。  String str = "HelloJava";System.out.println(str.length());  ————————————————  版权声明：本文为CSDN博主「键盘侠001」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。原文链接：https://blog.csdn.net/qq\_43126186/article/details/98990041 |

## 运算符优先级

|  |
| --- |
| **运算符优先级**  **口诀：淡云一笔安洛三福 单目>算数运算符>移位>比较>按位>逻辑>三目>赋值**  单目运算符：+，-，++，--  算数运算符：+，-，\*，/，%  移位运算符：<<,>>  关系运算符：>,<,>=,<=,==,!=  位运算符：&，|，~，^,  逻辑运算符：&&，||  三目运算符：表达式1？表达式2：表达式3;  赋值运算符：=等 |

## 如何不使用第三个变量来交换两个数的值

 a=a^b  
b=b^a  
a=a^b

|  |  |
| --- | --- |
| 1  2  3 | a = a + b;  b = a - b;  a = a - b; |

这样就行了，前提是a和b不能是太大或太小的数，怕a+b越界

## For循环的执行顺序

## i++和++i的异同之处

**共同点：**

1、i++和++i都是变量自增1，都等价于i=i+1

2、如果i++,++i是一条单独的语句，两者没有任何区别，

如：for（int i=1;i<10;i++）和for（int i=1;i<10;++i）

3、i++和++i的使用仅仅针对变量。 5++和++5会报错，因为5不是变量。

**不同点：**

如果i++,++i不是一条单独的语句，他们就有区别i++ ：先运算后增1。如：

|  |  |
| --- | --- |
| 1  2  3  4 | int x=5;  int y=x++;  System.out.println("x="+x+", y="+y);      //以上代码运行后输出结果为：x=6, y=5 |

++i ： 先增1后运算。如：

|  |  |
| --- | --- |
| 1  2  3  4 | int x=5;  int y=++x;  System.out.println("x="+x+", y="+y);      //以上代码运行后输出结果为：x=6, y=6 |

## Java型本数据类型

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  | **Java基本数据类型** |  | | 数据类型 | 字节 | 表示范围 | | Long(长整型) | 8 | -2^63~2^63-1 | | Int（整数型） | 4 | -2^31~2^31-1 | | Short（短整型） | 2 | -2^15~2^15-1 | | Byte（位） | 1 | -128~127 | | Char（字符型） | 2 | 0~65536的数所代表的Unicode表 | | Boolen（布尔型） | 1 | 布尔值只能使用true或false | | Float（浮点型单精度） | 4 | -3.4X10^38 | | Double（浮点型双精度） | 8 | -1.7X10^308 |   关于boolean占几个字节，众说纷纭，虽然boolean表现出非0即1的“位”特性，但是存储空间的基本计量单位是字节，不是位。所以boolean至少占1个字节。  JVM规范中，boolean变量当作int处理，也就是4字节；而boolean数组当做byte数组处理，即boolean类型的数组里面的每一个元素占1个字节。 |

## 封装、继承、多态？

|  |
| --- |
| 封装：通过对象的set、get方法设置和获取类的属性，隐藏内部实现，提高数据的安全性  继承：把这个已经存在的类所定义的内容作为自己的内容，并可以加入若干新的内容，或修改原来的方法使之更适合需要  多态：指程序中定义的引用变量所指向的具体类型和通过该引用变量发出的方法调用在编程时并不确定，而是在程序运行期间才确定，即一个引用变量倒底会指向哪个类的实例对象，该引用变量发出的方法调用到底是哪个类中实现的方法，必须在由程序运行期间才能决定。  Object obj = new xxx();  User Dao user Dao = new UserDaoJdbcImpl();  User Dao user Dao = new UserDaoHibernateImpl();  靠的是父类或接口定义的引用变量可以指向子类或具体实现类的实例对象，而程序调用的方法在运行期才动态绑定，就是引用变量所指向的具体实例对象的方法，也就是内存里正在运行的那个对象的方法，而不是引用变量的类型中定义的方法。 |

## 为什么还要包装类?

|  |
| --- |
| 首先8种基本类型每种类型都提供了包装类只是int变成了Integer，char变成了Character其他都是首字母大写  自动装箱：把基本类型转变成包装类型 例如：Integer I = 1；实际上编译时调用Integer.valueOf(i)  自动拆箱：把包装类转变成基本类型 例如：int s = I 编译时候调用l.intValue();  Java是面向对象语言，而基本的数据类型，不具备面向对象特性  在集合存的是对象，所以为了方便直接写基本类型会转变成包装类 |

## 基本类和包装类的区别

|  |
| --- |
| 包装类是对象,拥有方法和字段.而基本类型不是  包装类是引用传递放堆， 而基本类型是值传递放栈  包装类默认值是null，基本类型各不相同 |

## 说一下“==”和equals的区别

|  |
| --- |
| ==用来判断两个变量之间的的值是否相等。变量就可以分为基本数据类型变量，引用类型。  如果是基本数据类型的变量直接比较值而引用类型要比较对应的引用的内存的首地址。  \*\*\*其中int和integer比较的时候，Integer会向下转型成基本类型和int比较值。  equals 用来比较两个对象长得是否一样。判断两个对象的某些特征是否一样。实际上就是调用对象的equals方法进行比较。Equals是object类里面的方法，没有重写时实际是==的比较，在包装类中重写，比较的是内容，例如：String |

## String，StringBuild，StringBuffer区别

|  |
| --- |
| 1．说区别先说共同点，首先都是操作字符串的都能够做字符串拼接  String是final修饰的类，所以是内容不可变得字符串，底层使用了不可变的字符数组final char[]  StringBuild和stringbuffer是内容可变的字符数组，但stringbuild是非线程安全速度快stringbuffer线程安全速度稍慢  2.最经典就是拼接字符串。   1. String进行拼接.String c = “a”+”b”; 2. StringBuilder或者StringBuffer   StringBuilder sb = new StringBuilder(); sb.apend(“a”).apend(“b”)  -\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_-\_  这里研究一下string类创建对象个数，在笔试常有  String str=new String("aaa");  //2个·常量池一个，new在堆中创建一个，所以共两  那什么又是常量池：常量池(constant pool)指的是在编译期被确定，并被保存在已编译的.class文件中的一些数据。它包括了关于类、方法、接口等中的常量，也包括字符串常量。  1. String str1="aaa";  2. String str2="aaa";  //上面这行代码被执行的时候，JVM先到字符串池中查找，看是否已经存在值为"aaa"的对象，如果存在，则不再创新的对象，直接返回已存在对象的引用；如果不存在，则先创建这个对象，然后把它加入到字符串池中，再将的引用返回。 也只创建1个对象。能过上面的解释这个就更清楚了，在执行第二行代码时，aaa字符串对象在池中已存在，所以直接返回池中已存在的那个字符串对象。  String s = “a”+”b”+”c”只产生一个对象，右边都是字面量，编译时在string pool中生成“abc”  <https://blog.csdn.net/s1491695565/article/details/51833579> |

## Final、fially、finalize的区别

|  |
| --- |
| 1. final修饰符（关键字）   如果一个类被声明为final，意味着它不能再派生出新的子类，不能作为父类被继承。因此一个类不能既被声明为 abstract的，又被声明为final的。  被声明为final的变量必须在声明时给定初值，而在以后的引用中只能读取，不可修改。被声明为final的方法也同样只能使用，不能重载。   1. finally   在异常处理时提供 finally 块来执行任何清除操作。如果抛出一个异常，那么相匹配的 catch 子句就会执行，然后控制就会进入 finally 块（如果有的话）。如果finally中有return，直接返回finally中的   1. finalize   方法名。Java 技术允许使用 finalize() 方法在垃圾收集器将对象从内存中清除出去之前做必要的清理工作。这个方法是由垃圾收集器在确定这个对象没有被引用时对这个对象调用的。它是在 Object 类中定义的，因此所有的类都继承了它。子类覆盖 finalize() 方法以整理系统资源或者执行其他清理工作。finalize() 方法是在垃圾收集器删除对象之前对这个对象调用的。 |

## try{}catch(){}finally{}执行顺序

|  |
| --- |
| 异常处理中，try、catch、finally的执行顺序，大家都知道是按顺序执行的。即，如果try中没有异常，则顺序为try→finally，如果try中有异常，则顺序为try→catch→finally。但是当try、catch、finally中加入return之后，就会有几种不同的情况出现，下面分别来说明一下。也可以跳到最后直接看总结。    **一、try中带有return**  1 private int testReturn1() {  2 int i = 1;  3 try {  4 i++;  5 System.out.println("try:" + i);  6 return i;  7 } catch (Exception e) {  8 i++;  9 System.out.println("catch:" + i);  10 } finally {  11 i++;  12 System.out.println("finally:" + i);  13 }  14 return i;  15 }  输出：  try:2 finally:3 2  　　因为当try中带有return时，会先执行return前的代码，然后暂时保存需要return的信息，再执行finally中的代码，最后再通过return返回之前保存的信息。所以，这里方法返回的值是try中计算后的2，而非finally中计算后的3。但有一点需要注意，再看另外一个例子：  1 private List<Integer> testReturn2() {  2 List<Integer> list = new ArrayList<>();  3 try {  4 list.add(1);  5 System.out.println("try:" + list);  6 return list;  7 } catch (Exception e) {  8 list.add(2);  9 System.out.println("catch:" + list);  10 } finally {  11 list.add(3);  12 System.out.println("finally:" + list);  13 }  14 return list;  15 }  输出：  try:[1] finally:[1, 3] [1, 3]  　　看完这个例子，可能会发现问题，刚提到return时会临时保存需要返回的信息，不受finally中的影响，为什么这里会有变化？其实问题出在参数类型上，上一个例子用的是基本类型，这里用的引用类型。list里存的不是变量本身，而是变量的地址，所以当finally通过地址改变了变量，还是会影响方法返回值的。  **二、catch中带有return**  1 private int testReturn3() {  2 int i = 1;  3 try {  4 i++;  5 System.out.println("try:" + i);  6 int x = i / 0 ;  7 } catch (Exception e) {  8 i++;  9 System.out.println("catch:" + i);  10 return i;  11 } finally {  12 i++;  13 System.out.println("finally:" + i);  14 }  15 return i;  16 }  输出：  try:2 catch:3 finally:4 3  　　catch中return与try中一样，会先执行return前的代码，然后暂时保存需要return的信息，再执行finally中的代码，最后再通过return返回之前保存的信息。所以，这里方法返回的值是try、catch中累积计算后的3，而非finally中计算后的4。  **三、finally中带有return**  1 private int testReturn4() {  2 int i = 1;  3 try {  4 i++;  5 System.out.println("try:" + i);  6 return i;  7 } catch (Exception e) { 8 i++;  9 System.out.println("catch:" + i);  10 return i;  11 } finally {  12 i++;  13 System.out.println("finally:" + i);  14 return i;  15 }  16 }  输出：  try:2 finally:3 3  当finally中有return的时候，try中的return会失效，在执行完finally的return之后，就不会再执行try中的return。这种写法，编译是可以编译通过的，但是编译器会给予警告，所以不推荐在finally中写return，这会破坏程序的完整性，而且一旦finally里出现异常，会导致catch中的异常被覆盖。    **总结：**  1、finally中的代码总会被执行。  2、当try、catch中有return时，也会执行finally。return的时候，要注意返回值的类型，是否受到finally中代码的影响。  3、finally中有return时，会直接在finally中退出，导致try、catch中的return失效。 |

## HashMap原理

|  |
| --- |
| 面试可以这样说：Hashmap底层是哈希表也叫散列表，底层是数组，数组的每个元素是一个单向链表，在jdk1.8中当一个单向链表元素超过8个节点的时候，会调整成树形结构  Put存key、value数据，通过key的hashCode方法计算hash值，再根据Hash值计算key在数组中的索引值，根据索引值访问table[index]这个单向链表，如果为null，就说明这个位置没有人来过，就会生成一个新的节点存储到该位置table[index].  第二种情况：通过key的hashCode方法计算hash值，再根据Hash值计算key在数组中的索引值，根据索引值访问table[index]这个元素，如果不为null，就依次访问table[index]这条链条的各个结点，先判断是否有某个结点hash值与刚才计算key的hsah相等，如果所有结点的hsah值都不等于它，就会生成一个新的节点插到链表的头部  第三种情况: 通过key的HashCode方法计算hash值，再根据Hash值计算key在数组中的索引值，根据索引值访问table[index]这个元素，遍历单向链表中的各个节点，找到一个节点的hash值等于当前key的hash值，再把当前key与节点的key进行equals判等，如果equals相等了就用新的value替换旧的value值，如果equals不相等再比较下一个节点 |

## HashMap和HashTable的区别

|  |
| --- |
| \* 1) HashTable的父类是Dictionary, HashMap的父类是AbstractMap  \* 2)HashTable的初始化容量: 11, HashMap默认初始化容量: 16  \* 3) HashTable的加载因子: 0.75, HashMap的加载因子: 0.75  \* 4) HashTable扩容: (oldCapacity << 1) + 1 , HashMap按2倍大小扩容  \* 5) HashTable也可以指定初始化容量不会自动调整, HashMap会把第一个大于该数值的2次幂作为初始化容量\* 6) HashTable的键与值都不能为 null, HashMap的键与值可以为null  \* 7) HashTable是线程安全的, HashMap不是线程安全的 |

## list中删除指定元素

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1、从list中删除指定元素可以使用remove方法来实现:   |  |  | | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | ArrayList al = new ArrayList();    al.add("a");    al.add("b");    //al.add("b");    //al.add("c");    //al.add("d");    for (int i = 0; i < al.size(); i++) {    if (al.get(i) == "b") {    al.remove(i);    i--;    }    } | |  |  |   在代码中，删除元素后，需要把下标减一。这是因为在每次删除元素后，ArrayList会将后面部分的元素依次往上挪一个位置(就是copy)，所以，下一个需要访问的下标还是当前下标，所以必须得减一才能把所有元素都遍历完。  2还可以这样做  while(list.contains("0")) list.remove("0"); System.out.println(list.toString());//到此为止，所有的0都被删除了   1. 使用removeAll（） |

## 重载和重写要注意哪些

|  |
| --- |
| 判断方法重载的依据：  1、 必须是在同一个类中  2、 方法名相同  3、 方法参数的个数、顺序或类型不同  4、 与方法的修饰符或返回值没有关系  请看以上依据，重载能且只能在同一个类中 |

## 线程的三种实现方式

|  |
| --- |
| ①实现方式   1. 通过继承Thread类实现一个线程 2. 通过实现Runnable接口实现一个线程   继承扩展性不强，java总只支持单继承，如果一个类继承Thread就不能继承其他的类了。  3、实现callable接口，重写call方法,可提供返回值  ②怎么启动？  Thread = new Thread(继承了Thread的对象/实现了Runnable的对象)  thread.setName(“设置一个线程名称”);  thread.start();  启动线程使用start方法，而启动了以后执行的是run方法。  ③怎么区分线程？在一个系统中有很多线程，每个线程都会打印日志，我想区分是哪个线程打印的怎么办？  thread.setName(“设置一个线程名称”); 这是一种规范，在创建线程完成后，都需要设置名称。 |

# Web基础

## 静态代理和动态代理

## Post和get有什么区别

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| GET和POST的区别：   |  |  |  | | --- | --- | --- | |  | **GET** | **POST** | | **点击返回/刷新按钮** | 没有影响 | 数据会重新发送（浏览器将会提示用户“数据被从新提交”） | | **添加书签** | 可以 | 不可以 | | **缓存** | 可以 | 不可以 | | **编码类型(Encoding type)** | application/x-www-form-urlencoded | application/x-www-form-urlencoded or multipart/form-data. 请为二进制数据使用multipart编码 | | **历史记录** | 有 | 没有 | | **长度限制** | 有IE62083服务器8000 | 没有 | | **数据类型限制** | 只允许ASCII字符类型 | 没有限制。允许二进制数据 | | **安全性** | 查询字符串会显示在地址栏的URL中，不安全，请不要使用GET请求提交敏感数据 | 因为数据不会显示在地址栏中，也不会缓存下来或保存在浏览记录中，所以看POST求情比GET请求安全，但也不是最安全的方式。如需要传送敏感数据，请使用加密方式传输 | | **可见性** | 查询字符串显示在地址栏的URL中，可见 | 查询字符串不会显示在地址栏中，不可见 |   其他HTTP请求方式   |  |  | | --- | --- | | **方式** | **描述** | | HEAD | 与GET请求类似，不同在与服务器只返回HTTP头部信息，没有页面内容 | | PUT | 上传指定URL的描述 | | DELETE | 删除指定资源 | | OPTIONS | 返回服务器支持的HTTP方法 | | CONNECT | 转换为透明TCP/IP隧道的连接请求 | |

## Session和cookie区别

|  |
| --- |
|  |

## jsp九大内置对象分别是

|  |
| --- |
| request              请求对象　                类型 javax.servlet.ServletRequest         作用域    Request  response           响应对象                   类型 javax.servlet.SrvletResponse         作用域    Page  pageContext     页面上下文对象        类型 javax.servlet.jsp.PageContext       作用域    Page  session             会话对象                    类型 javax.servlet.http.HttpSession       作用域    Session  application        应用程序对象            类型 javax.servlet.ServletContext           作用域    Application  out                      输出对象                    类型 javax.servlet.jsp.JspWriter           作用域    Page  config                 配置对象                    类型 javax.servlet.ServletConfig            作用域    Page  page                   页面对象                    类型 javax.lang.Object                       作用域    Page  exception           例外对象                    类型 javax.lang.Throwable                     作用域   page |

## 什么是跨域？怎么将解决

|  |
| --- |
| <https://www.jianshu.com/p/8fa2acd103ea> 1.什么是跨域？ 跨域：指的是浏览器不能执行其他网站的脚本。它是由浏览器的同源策略造成的，是浏览器对javascript施加的安全限制。  例如：a页面想获取b页面资源，如果a、b页面的协议、域名、端口、子域名不同，所进行的访问行动都是跨域的，而浏览器为了安全问题一般都限制了跨域访问，也就是不允许跨域请求资源。注意：跨域限制访问，其实是**浏览器的限制**。理解这一点很重要！！！  同源策略：是指协议，域名，端口都要相同，其中有一个不同都会产生跨域； 2.跨域访问示例 假设有两个网站，A网站部署在：http://localhost:81 即本地ip端口81上；B网站部署在：http://localhost:82 即本地ip端口82上。现在A网站的页面想去访问B网站的信息，A网站页面的代码如下（这里使用jquery的异步请求）  $(function (){      $.get("http://localhost:82/api/values", {},function (result) {            $("#show").html(result);    })})；  从错误信息可以看出以上出现了跨域问题！ 3.如何解决跨域问题？ 由之前的介绍我们已经知道错误的原因，既然跨域会产生问题，那么我们就不跨域不就完了嘛！！！  先上图：  首先我们用nginx作为代理服务器和用户交互，这样用户就只需要在80端口上进行交互就可以了，这样就避免了跨域问题，因为我们都是在80端口上进行交互的；  下面我们看一下利用nginx作为反向代理的具体配置：  server {          listen      80; #监听80端口，可以改成其他端口          server\_name  localhost; # 当前服务的域名          #charset koi8-r;          #access\_log  logs/host.access.log  main;          location / {              proxy\_pass http://localhost:81;              proxy\_redirect default;          }  location /apis { #添加访问目录为/apis的代理配置  rewrite  ^/apis/(.\*)$ /$1 break;  proxy\_pass  http://localhost:82;        }  #以下配置省略  1.当用户发送localhost:80/时会被nginx转发到http://localhost:81服务；  2.当界面请求接口数据时，只要以/apis 为开头，就会被nginx转发到后端接口服务器上；  总结：nginx实现跨域的原理，实际就是把web项目和后端接口项目放到一个域中，这样就不存在跨域问题，然后根据请求地址去请求不同服务器（真正干活的服务器）；  作者：仰望星空得少年 链接：https://www.jianshu.com/p/8fa2acd103ea 来源：简书 简书著作权归作者所有，任何形式的转载都请联系作者获得授权并注明出处。 |

## HTTP请求报文和响应报文

|  |
| --- |
| 请求报文 一个HTTP请求报文由请求行（request line）、请求头部（header）、空行和请求数据4个部分组成，下图给出了请求报文的一般格式。2019年8月12日 |

## Forword和redirect的区别

|  |
| --- |
| 请求转发是一个请求，而重定向是两个请求；  请求转发后浏览器地址栏不会有变化，而重定向会有变化，因为重定向是两个请求；  请求转发的目标只能是本应用中的资源，重定向的目标可以是其他应用；  请求转发对AServlet和BServlet的请求方法是相同的，即要么都是GET，要么都是POST，因为请求转发是一个请求；  重定向的第二个请求一定是GET； |

# Dubbo

## 为什么用dubbo

dubbo，某种程度上而言，阿里在这里推动了行业技术的前进。

正是因为有阿里的dubbo，很多中小型公司才可以基于dubbo，来把系统拆分成很多的服务，每个人负责一个服务，大家的代码都没有冲突，服务可以自治，自己选用什么技术都可以，每次发布如果就改动一个服务那就上线一个服务好了，不用所有人一起联调，每次发布都是几十万行代码，甚至几百万行代码了。

# Mybatis

## Mybatis怎么写插入多条数据？

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" ?>  <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">  <mapper namespace="com.example.mapper.UserMapper">  <insert id="insertIntoUsers" parameterType="java.util.ArrayList">  insert into `user`(uPassword,uName,uIDCard,uPhone,uEmail,uStatus) values  <foreach collection="list" index="index" separator="," item="item">  (#{item.uPassword},#{item.uName},#{item.uIDCard},#{item.uPhone},#{item.uEmail},#{item.uStatus})  </foreach>  </insert>  </mapper>  提示：一般不要写open,close，不然容易造成嵌套错误。  nested exception is java.sql.SQLException: Column count doesn't match value count at row 1  (嵌套异常：sql语句的列计数与第1行的值计数不相匹配)  特别注意：mysql默认接受sql的大小是1048576(1M)，List插入数据量超过1M会报如下异常：（可通过调整MySQL安装目录下的my.ini文件中[mysqld]段的＂max\_allowed\_packet = 1M＂）  nested exception is com.mysql.jdbc.PacketTooBigException: Packet for query is too large (5677854 > 1048576).  You can change this value on the server by setting the max\_allowed\_packet' variable.  ————————————————  版权声明：本文为CSDN博主「CCIEJohn\_zhou」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。  原文链接：https://blog.csdn.net/CCIEJohn\_zhou/article/details/90523769 |

## 什么是框架？

|  |
| --- |
| C:\Users\Administrator\Desktop\面试  把程序代码的公共部分提取出来，最终打成一个jar或多个jar  使用框架可以提高开发效率  程序员只要写变化的代码  Mybatis是封装了jdbc的持久层框架，属于Dao层  属于javaee设计模式  底层用了大量的反射，用 |

## {}与#{}区别

|  |
| --- |
| #{}是preparedStatement预编译，#{} 在预处理时，会把参数部分用一个占位符 ? 传入的是字符串  ${}是statement字符串注入，一般注入表名和排序字段 |

## 常用标签

|  |
| --- |
| Select  Insert  Update  Delete |
|  |

## 动态SQL

|  |
| --- |
| 1、Foreach删除多个id对应的数据  <delete>  Delete from xxxx where id in  <foreach元素item=”id” separator以什么分隔=“，” open以什么开始=“（” close以什么结束=“）”>  #{id}  </foreach>  </delete>  <delete>  delete from xxxx where id in (  <foreach元素item=”id” separator以什么分隔=“，”>  #{id}  </foreach>  )  </delete>   1. 插入多条数据 public int dosaves(List<User> userList){return doword};   <insert id=”save3 parameterType=”User”>  Insert into tbl\_user  (id,username,password,email)  Values  <foreach collection=”list” item=”o” separator=”,”>  (#{o.id},#{o.username},#{o.password},#{o.email})  </foreach>  </insert> |
| 4、<where>和<if>主要使用在分页查询当中  分析1：\*分页查询的时候，浏览器向服务器提交哪些数据  \*提交查询条件（condition）  \*提交页码（pageNo）  \*提交每页显示记录条数（pageSize）  分析2：返回json数组字符串  \*返回符合查询条件的总记录条数（total）  \*返回符合查询条件的当前页数据  Json格式  {“total”:3,  “dataList”:[  {“id”:’1’,”username”:”lisi”,”password”:123,“email”：”123@qq.com”},  {“id”:’2’,”username”:”heqi”,”password”:456,“email”：”4563@qq.com”},  {“id”:’3’,”username”:”haha”,”password”:666,“email”：”666@qq.com”}  ]  }  分析3：Cotroller怎么写，service怎么写，dao怎么写  分析4：sql语句怎么写？  \*几条SQL语句？  \*第一条  Select  Count(\*)  from  tal\_user  <where>  <if test=“username ！= null and username ！= ‘’”>  and username like ‘%’ #{username} ‘%’  </if>  <if test=“email ！= null and email ！= ‘’”>  and email like ‘%’ #{email} ‘%’  </if>  </where>  \*第二条  Select  \*  From  tal\_user  <where> -- 这里面写都都是条件  <if test=“username ！= null and username ！= ‘’”>  and username like ‘%’ #{username} ‘%’< -- 注意mybatis中没有空格，是无法识别占位符的mybatis要求#{}这种占位符俩边必须要空格>  </if>  <if test=“email ！= null and email ！= ‘’”>  and email like ‘%’ #{email} ‘%’  </if>  order by  id asc 按id升序排列  limit  开始下标（pageNo-1）\*pageSize，每页显示数据条数#{pageSize}  PS：为什么是pageNo-1，因为我要看到当前页的需要把之前所有量计算出来所以-1，又因为下标是从0开始的，所以乘以pageSize不用再加一  </where>  注意：1、mybatis的SQL语句不支持数学运算，提前算好，传递过来就行  2、and不能换成&& |

## Jackion插件

## 数据库连接池

|  |
| --- |
| 1、数据库连接数，也就是一个数据库,最多能够同时 接受 多少个 客户的连接. 2、在没有数据库连接池 的情况下, 一个客户,每次访问, 就要创建一个 数据库连接, 执行 SQL, 获取结果， 然后关闭、释放掉数据库连接，问题就在于创建一个数据库连接， 是一个很消耗资源，花费很多时间的操作，于是数据库连接池产生了。  3、数据库连接池 预先打开一定数量的数据库连接， 并维持着连接。 4、当客户要执行[SQL语句](https://www.baidu.com/s?wd=SQL%E8%AF%AD%E5%8F%A5&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)的时候， 从数据库连接池 里面，获取一个连接，执行SQL, 获取结果， 然后把数据库连接，交还给数据库连接池。 5、假如一个 session , 执行10次独立的操作，那么不使用 数据库连接池的，需要创建数据库连接10次，并关闭10次。  6、使用 数据库连接池的，直接使用 数据库连接池中已经打开好的，直接使用。 |

## 配置文件数据源

|  |
| --- |
| 驱动、mysql地址、用户名、密码 |

## MyBatis的事务管理

|  |
| --- |
| MyBatis的事务管理分为两种形式：  一、使用JDBC的事务管理机制：即利用java.sql.Connection对象完成对事务的提交（commit()）、回滚（rollback()）、关闭（close()）等  二、使用MANAGED的事务管理机制：这种机制MyBatis自身不会去实现事务管理，而是让程序的容器如（JBOSS，Weblogic）来实现对事务的管理 |

## Hibernate与mybatis区别

|  |
| --- |
| Hibernate与mybatis相比较，mybatis更为轻便、灵活，容易掌握。mybatis可以把sql语句从java代码中分离了出来，放在了配置文件中书写，大大降低里java代码与SQL语句的耦合度，更容易对sql语句操作，重要的是mybatis还可以书写动态的sql语句，但mybatis也存在一些缺陷，比如mybatis本身的缓存机制没有hibernate那么完善，hibernate除了本身有良好的缓存机制，还可以使用第三方缓存。Hibernate较完整的封装了JDBC，但学起来要比mybatis更困难一些。Hibernate的DAO层开发比MyBatis简单，对对象的维护和缓存要比MyBatis好。  ————————————————  版权声明：本文为CSDN博主「x\_minde」的原创文章，遵循CC 4.0 by-sa版权协议，转载请附上原文出处链接及本声明。  原文链接：https://blog.csdn.net/xue\_mind/article/details/52959107 |

# Git

## Git分支你们是怎么管理的❤？

## Git和svn有什么区别？

|  |
| --- |
|  |

## Git工作原理？

## Git常见的命令？

# SVN

|  |
| --- |
| 1、什么是SVN？ F:\JavaProjects\13-SVN完整的资料藏在电脑这个路径下面，需要时候直接访问  SubVersioN：软件版本控制工具  只是一个工具而已。  专门完成软件版本的管理。  和SVN工具类似的还有：Git    目前在企业中SVN和Git工具都有在使用。各占50%左右。  Git也是最近几年才兴起的。  SVN之前的比重很高。 2、SVN工具有什么作用呢？没有SVN工具之前团队是如何协作的呢？ 团队开发，代码一定是共享的，不是各干各的。  没有SVN工具之前，那么代码是怎么共享的呢？  使用FTP的方式，FTP服务器上存放共享的代码，FTP客户端进行下载，  每天写完代码之后将自己写的代码传到服务器上。  这种方式，使用一不小心就会出现覆盖的问题。    在这种背景下，在这种情况下，出现了很多版本控制工具，例如：CVS、Clear Case等。  CVS的升级版是SVN。  当然，目前使用Git的也比较多。  SVN最主要就是对代码的版本进行控制，来达到保证代码不会丢失或覆盖。 3、在SVN工具当中涉及到这么几个术语？ \* 仓库 (Repository)：用来存储共享的代码，资料等。  \* 从仓库上往客户端下载代码称为：checkout（检出）  \* 从客户端向仓库传代码称为：checkin（检入）  \* 更新(update)、提交(commit)  4、在实际开发中，公司当中的某个角落应该有这样的一台计算机，该  计算机上安装了SVN服务器端，在这台计算机上创建了一个仓库，每一个  程序员电脑上都安装了SVN客户端工具，从仓库上下载代码，或者传代码到  仓库。  SVN工具包括：  服务器端（程序员是不需要安装的）  客户端（程序员需要安装的）  5、开始工作的时候，项目经理会给程序员一个SVN的仓库地址，以及用户名和密码。  6、仓库地址：  https://192.168.151.2/svn/crm/  7、步骤：模拟两个用户（admin:123,zhangsan:123）  - 新建两个目录：admin、zhangsan  - 分别在admin和zhangsan目录下与仓库同步：  右键 --> SVN Checkout  - checkin:  选中要提交的文件，右键-->TortoiseSVN --> Add..（先加入SVN客户端软件当中）  右键 --> SVN Commit...  - checkout:  空白出：右键 --> svn update...    - 非常友好的情况下：  admin提交，zhangsan下载  zhangsan提交，admin下载  同步的情况下不会出现任何问题。  异步的时候可能会出现问题：两者同时并发，谁也不管谁。  SVN主要解决的就是这个问题。（出现冲突后怎么解决。）  8、在实际开发中，代码是放到eclipse当中，所以eclipse应该安装一个SVN客户端插件。  9、使用SVN工具的注意事项：  - 程序中存在编译错误的代码不能提交。  - 有些公司要求更严格，没有经过单元测试的代码也不能提交。（提交到仓库上的代码一定是合法有效的代码。）  - 项目当中与本地环境有关系的文件不允许提交到仓库上。（因为每个程序员本地的环境都不同）  .classpath  .project  - 提交到SVN仓库上的程序，别人一定知道是谁哪一天、哪个时间点提交上去的。  - 提交的单位越小越好。  \* 文件尽可能少，不要锁文件  \* 上一次提交和下一次提交的时间间隔尽可能小。  \* 最好以天为单位。（晚上下班提交，早晨来了第一件事更新。）    - 先更新后提交。  - 提交文件的时候一定要注意：不要锁文件。  - 提交之前不确定的时候，可以先“与资源库同步”。  如果远程代码变更与你本地的修改没有冲突，就不会有什么问题，  如果远程代码与你的修改有冲突是不会直接覆盖你本地的修改的，  它会问你怎么去处理冲突，是采用本地的还是远程的。所以一般不用担心会被覆盖。 |

# MySQL

## select查询语句得执行顺序

|  |
| --- |
| [MySQL SELECT执行顺序](https://www.cnblogs.com/linjiqin/p/5509547.html) SELECT语句的完整语法为：  (7) SELECT  (8) DISTINCT <select\_list>  (1) FROM <left\_table>  (3) <join\_type> JOIN <right\_table>  (2) ON <join\_condition>  (4) WHERE <where\_condition>  (5) GROUP BY <group\_by\_list>  (6) HAVING <having\_condition>  (9) ORDER BY <order\_by\_condition>  (10) LIMIT <limit\_number>  说明：语法前面的序号为SELECT执行顺序  MySQL的SELECT执行顺序一共分为10步，如上所标注的那样，最先执行的是FROM操作，最后执行的是LIMIT操作。其中每一次操作都会产生一张虚拟的表，这个虚拟的表作为一个处理的输入，只是这些虚拟的表对用户来说是透明的，但是只有最后一个虚拟的表才会被作为结果返回。如果没有在语句中指定某一个子句，那么将会跳过相应的步骤。  下面我们来具体分析一下查询处理的每一个阶段：  SELECT a.customer\_id, COUNT(b.order\_id) as total\_orders  FROM table1 AS a  LEFT JOIN table2 AS b  ON a.customer\_id = b.customer\_id  WHERE a.city = 'hangzhou'  GROUP BY a.customer\_id  HAVING count(b.order\_id) < 2  ORDER BY total\_orders DESC;  1、FORM: 对FROM左边的表和右边的表计算笛卡尔积，产生虚表VT1。 2、ON: 对虚表VT1进行ON过滤，只有那些符合<join-condition>的行才会被记录在虚表VT2中。 3、JOIN： 如果指定了OUTER JOIN（比如left join、 right join），那么保留表中未匹配的行就会作为外部行添加 到虚拟表VT2中，产生虚拟表VT3。 4、WHERE： 对虚拟表VT3进行WHERE条件过滤。只有符合<where-condition>的记录才会被插入到虚拟表VT4中。 5、GROUP BY: 根据group by子句中的列，对VT4中的记录进行分组操作，产生VT5。 6、HAVING： 对虚拟表VT5应用having过滤，只有符合<having-condition>的记录才会被 插入到虚拟表VT6中。 7、SELECT： 执行select操作，选择指定的列，插入到虚拟表VT7中。 8、DISTINCT： 对VT7中的记录进行去重。产生虚拟表VT8. 9、ORDER BY: 将虚拟表VT8中的记录按照<order\_by\_list>进行排序操作，产生虚拟表VT9. 10、LIMIT：取出指定行的记录，产生虚拟表VT10, 并将结果返回。 |

## MySQL和Oracle的区别

|  |
| --- |
| 同时二者都支持serializable串行化事务隔离级别，可以实现最高级别的     读一致性。每个session提交后其他session才能看到提交的更改。Oracle通过在undo表空间中构造多版本数据块来实现读一致性，每个session     查询时，如果对应的数据块发生变化，Oracle会在undo表空间中为这个session构造它查询时的旧的数据块     MySQL没有类似Oracle的构造多版本数据块的机制，只支持read commited的隔离级别。一个session读取数据时，其他session不能更改数据，但     可以在表最后插入数据。session更新数据时，要加上排它锁，其他session无法访问数据  (5) 保存数据的持久性     MySQL是在数据库更新或者重启，则会丢失数据，Oracle把提交的sql操作线写入了在线联机日志文件中，保持到了磁盘上，可以随时恢复 (6) 并发性     MySQL以表级锁为主，对资源锁定的粒度很大，如果一个session对一个表加锁时间过长，会让其他session无法更新此表中的数据。   虽然InnoDB引擎的表可以用行级锁，但这个行级锁的机制依赖于表的索引，如果表没有索引，或者sql语句没有使用索引，那么仍然使用表级锁。   Oracle使用行级锁，对资源锁定的粒度要小很多，只是锁定sql需要的资源，并且加锁是在数据库中的数据行上，不依赖与索引。所以Oracle对并   发性的支持要好很多。 (8) 复制     MySQL:复制服务器配置简单，但主库出问题时，丛库有可能丢失一定的数据。且需要手工切换丛库到主库。     Oracle:既有推或拉式的传统数据复制，也有dataguard的双机或多机容灾机制，主库出现问题是，可以自动切换备库到主库，但配置管理较复杂。 (9) 性能诊断     MySQL的诊断调优方法较少，主要有慢查询日志。     Oracle有各种成熟的性能诊断调优工具，能实现很多自动分析、诊断功能。比如awr、addm、sqltrace、tkproof等     (11)分区表和分区索引     MySQL的分区表还不太成熟稳定。     Oracle的分区表和分区索引功能很成熟，可以提高用户访问db的体验。  相同点：  1.都是关系型数据库管理系统  2.都是目前比较流行的数据库  不同点：   * 1. **大小**   MySQL是轻量型数据库152M，并且免费开源的，没有服务恢复数据。 Oracle是重量型数据库3G空间大小，特别吃内存,而且收费，Oracle公司对Oracle数据库提供服务   * 1. **逻辑备份**     MySQL逻辑备份时要锁定数据，才能保证备份的数据是一致的，影响业务正常的DML语句使 用,Oracle逻辑备份时不锁定数据，且备份的数据是一致   2. **单引号的处理**Mysql里可以用双引号包起字符串，Oracle里只可以用单引号包起字符串。在插入和修改字符串前必须做单引号的替换：把所有出现的一个单引号替换成两个单引号。   3. **自动增长**   Mysql是一个自动增长的数据类型，插入数据的时候，不需要管理，它自己会自动增长，Oracle不支持自动增长的数据类型，通过建立一个自动增长的序列号来完成自动增长。   * 1. **事务隔离级别**   MYSQL 默认是''可重读 '' ---- repeatable read  ORCALE 默认是的"读提交” ---- read commited   * 1. **对事务的支持** MySQL在innodb存储引擎的行级锁的情况下才可支持事务，而Oracle则完全支持事务   2. **事物提交方式**   oracle默认不自动提交，需要用户手动提交。  Mysql默认是自动提交。不支持事物。  Mysql默认自动提交，也就是你提交一个query，他就直接执行，我们可以通过  set autocommit=0 禁止自动提交  set autocommit=1 开启自动提交   * 1. **翻页的SQL语句的处理**   MySQL是直接在SQL语句中写"select... from ...where...limit  x, y",有limit就可以实现分页;而Oracle则是需要用到伪列ROWNUM和嵌套查询，并且只能用ROWNUM<100，不能用ROWNUM>80。  以下是经过分析后较好的两种Oracle翻页SQL语句(ID是唯一关键字的字段名)：  语句一：  SELECT ID, [FIELD\_NAME,...] FROM TABLE\_NAME WHERE ID IN ( SELECT ID FROM (SELECT ROWNUM AS NUMROW, ID FROM TABLE\_NAME WHERE 条件1 ORDER BY 条件2) WHERE NUMROW > 80 AND NUMROW < 100 ) ORDER BY 条件3;  语句二：SELECT \* FROM (( SELECT ROWNUM AS NUMROW, c.\* from (select [FIELD\_NAME,...] FROM TABLE\_NAME WHERE 条件1 ORDER BY 条件2) c) WHERE NUMROW > 80 AND NUMROW < 100 ) ORDER BY 条件3;   * 1. **sql语句的扩展和灵活性**   Mysql对sql语句有很多非常实用而方便的扩展，比如limit功能，insert可以一次插入多行数据，select某 些管理数据可以不加from。Oracle在这方面感觉更加稳重传统一些。   * 1. **长字符串的处理**   长字符串的处理ORACLE也有它特殊的地方。INSERT和UPDATE时最大可操作的字符串长度小于等于4000个单字节, 如果要插入更长的字符串, 请考虑字段用CLOB类型，方法借用ORACLE里自带的DBMS\_LOB程序包。插入修改记录前一定要做进行非空和长度判断，不能为空的字段值和超出长度字段值都应该提出警告,返回上次操作。   * 1. **空字符的处理**   Mysql的非空字段也有空的内容，ORACLE里定义了非空字段就不容许有空的内容。按Mysql的NOT NULL来定义Oracle表结构, 导数据的时候会产生错误。因此导数据时要对空字符进行判断，如果为NULL或空字符，需要把它改成一个空格的字符串。   1. **字符串的模糊比较量**   Mysql里用 字段名 like ‘%字符串%’;  Oracle里也可以用 字段名 like ‘%字符串%’ 但这种方法不能使用索引, 速度不快。  原文链接：https://blog.csdn.net/qq\_35835118/article/details/83961788 |
|  |
|  |

## MySQL进行行转列的示范

|  |
| --- |
| SELECT userid,  SUM(IF(`subject`='语文',score,0)) as '语文',  SUM(IF(`subject`='数学',score,0)) as '数学',  SUM(IF(`subject`='英语',score,0)) as '英语',  SUM(IF(`subject`='政治',score,0)) as '政治'  FROM tb\_score  GROUP BY userid  ——————————————————— ——————————————————————————————  注意点：  （1）SUM() 是为了能够使用GROUP BY根据userid进行分组，因为每一个userid对应的subject="语文"的记录只有一条，所以SUM() 的值就等于对应那一条记录的score的值。  假如userid ='001' and subject='语文' 的记录有两条，则此时SUM() 的值将会是这两条记录的和，同理，使用Max()的值将会是这两条记录里面值最大的一个。但是正常情况下，一个user对应一个subject只有一个分数，因此可以使用SUM()、MAX()、MIN()、AVG()等聚合函数都可以达到行转列的效果。  （2）IF(subject='语文',score,0) 作为条件，即对所有subject='语文'的记录的score字段进行SUM()、MAX()、MIN()、AVG()操作，如果score没有值则默认为0  ————————————————  版权声明：本文为CSDN博主「天问运」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。  原文链接：https://blog.csdn.net/lilong329329/article/details/81664451 |

## 列级约束和表级约束的区别

|  |
| --- |
|  |

## 主键和唯一约束的区别

|  |
| --- |
|  |

## 查看msyql表中所有索引，包括主键，外键，唯一

|  |
| --- |
|  |

## 表的约束与给表格创建约束

|  |
| --- |
|  |

## 表的创建语法

|  |
| --- |
|  |

## 数据库的数据类型

|  |
| --- |
|  |

## 复制表的语法

|  |
| --- |
|  |

## 删除表的语法

|  |
| --- |
|  |

## 修改表的语法

|  |
| --- |
|  |

## truncate和delete删除的区别

|  |
| --- |
|  |

## 四种隔离级分别是？

|  |
| --- |
| 一、读未提交  二、读已提交  三、可重复读  四、可串行化 |

## 数据库设计的经验

|  |
| --- |
| **1: 表的数目不要太多，一般20-30张就够了。如果表的数目太多，则可以考虑采用同化操作，即将大体相同的实体放入到一张表中。**  **2：当数据库中的信息非常庞大时，不要使用外键(逆规范化)，因为由此可能带来非常大的性能损失。**  **3：一般以消耗存储空间来换取效率。** |

## 数据库三范式

|  |
| --- |
| 第一范式（ 1NF）： 字段具有原子性,不可再分。 所有关系型数据库系统都满足第一范式） 数据库表中的字段都是单一属性的， 不可再分。 要求数据库表中的每个实例或行必须可以被惟一地区分。 通常需要为表加上一个列， 以存储 各个实例的惟一标识。 这个惟一属性列被称为主关键字或主键。 第二范式（ 2NF） 要求实体的属性完全依赖于主关键字。 所谓完全依赖是指不能存在仅依赖 主关键字一部分的属性， 如果存在， 那么这个属性和主关键字的这一部分应该分离出来形成 一个新的实体， 新实体与原实体之间是一对多的关系。为实现区分通常需要为表加上一个列， 以存储各个实例的惟一标识。 简而言之， 第二范式就是非主属性非部分依赖于主关键字。 第三范式的要求如下： 满足第三范式（ 3NF） 必须先满足第二范式（ 2NF）。 简而言之， 第三范式（ 3NF） 要求一个 数据库表中不包含已在其它表中已包含的非主关键字信息。 |

## Mysql存储引擎有哪些，有什么区别

|  |
| --- |
|  |

## 数据库去重的几种方法及删除表中重复的方法20191105

|  |
| --- |
| 数据库中的去重操作（删除数据库中重复记录的SQL语句）主要有三种方法  （1）rowid方法  （2）group by 方法  （3）distinct方法  1、rowid方法  根据Oracle带的rowid属性，可以进行判断是否存在重复语句；  --对a，b字段都重复的记录，只保留最新的一条  select a,b,max(rowid) from test  group by a,b;  --删除操作  delete from test t where rowid not in  (  select max(rowid) from test  group by a,b;  )  2、用group by方法  主要用于分组统计,一般都是使用在聚合函数中使用；  select t.name,count(name) from STUDENT t group by t.name having count(name) > 1;  --删除重复数据，重复数据只保留id最大一条  delete from user  where username in (select username from (select \* from user group by username having count(username)>1) t1) and id not in(  select t.id from(  select max(id) as id from user group by username having count(username)>1  ) t  )  3、用distinct方法  一般用于比较小的表进行去重,会过滤掉多余的重复记录，返回不重复的记录或字段；  select distinct t.name from STUDENT t;  ————————————————  版权声明：本文为CSDN博主「visant」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。  原文链接：https://blog.csdn.net/visant/article/details/79720085 |

## SQL中 select count(1) count中的1 到底是什么意思呢?和count(\*)的区别？

|  |
| --- |
|  |

## Count什么情况下会忽略null值

|  |
| --- |
|  |

## 数据库常用函数有哪些

|  |
| --- |
| 1. 数学函数   ceiling(x) 返回大于x的最小整数值，取顶  floor(x) 返回小于x的最大整数值，取底  pi() 返回圆周率  rand() 返回０到１内的随机值,可以通过提供一个参数(种子)使rand()随机数生成器生成一个指定的值。  round(x,y) 返回参数x的四舍五入的有y位小数的值  二、聚合函数 (常用于group by从句的select查询中)  avg(col) 返回指定列的平均值，排除为null的值  count(col) 返回指定列中非null值的个数，排除为null的值  min(col) 返回指定列的最小值，排除为null的值  max(col) 返回指定列的最大值，排除为null的值  sum(col) 返回指定列的所有值之和，排除为null的值  group\_concat(col) 返回属于一组的列值连接组合而成的结果，排除为null的值  针对group\_concat的使用，在网上找了案例  <https://blog.csdn.net/yz357823669/article/details/78794050>还有去重、排序和设置分隔符  **SELECT** **id**,**GROUP\_CONCAT**(score) **FROM** testgroup **GROUP** **BY** **id** |

|  |
| --- |
| SQL查询~ 存在一个表而不在另一个表中的数据 转载[mutu\_du](https://me.csdn.net/xiaohanshenchu) 发布于2013-02-20 11:15:26 阅读数 6270  收藏  转)A、B两表，找出ID字段中，存在A表，但是不存在B表的数据。A表总共13w数据，去重后大约3W条数据，B表有2W条数据，且B表的ID字段有索引。 方法一 　　使用 not in ,容易理解,效率低 ~执行时间为：1.395秒~  1 select distinct A.ID from A where A.ID not in (select ID from B) 方法二 　　使用 left join...on... , "B.ID isnull" 表示左连接之后在B.ID 字段为 null的记录 ~执行时间：0.739秒~  1 select A.ID from A left join B on A.ID=B.ID where B.ID is null  　　图解 方法三  　　逻辑相对复杂,但是速度最快 ~执行时间: 0.570秒~  1 select \* from B  2 where (select count(**1**) as num from A where A.ID = B.ID) = **0** |

## Mysql优化

|  |
| --- |
| <https://blog.csdn.net/a745233700/article/details/85265665> |

# Spring

## 什么是spring

|  |
| --- |
| Spring是一个2M大小的轻量级框架；他是一个容器管理对象，管理着对象和对象之间的依赖关系。  IOC:即控制反转，这不是一种技术，而是一种思想，被调用者的实例不是由调用者创建，而是交给Spring容器来管理。也就是说，所有的类都会在spring中进行登记，告诉spring自己（你）是什么，需要什么，然后spring会在系统运行到适合的时候把想要的东西给你，把你交给需要你的东西，所有的类的创建销毁都由spring完成。  AOP:面向切面编程，日志功能，事务管理，拦截器等都实现了AOP。AOP的底层是由动态代理实现完成的，在spring中AOP的代理是由IOC容器负责生成与管理。那么spring创建代理的规则为：  a) 默认情况下使用java的动态代理实现AOP的管理，这样就可以为任何接口实现创建代理了；  b)当需要代理的类不是接口的时候，spring就会切换为CGLIB代理。当然也可以spring强制使用CGLIB代理。 |

## Spring有什么好处？

|  |
| --- |
| Spring是一个框架，是一个半成品软件，有20个模块组成。它是一个容器管理对象，管理对象之间的依赖关系  1、Spring提供了对各种优秀框架（如Struts,Hibernate、MyBatis）等的直接支持。简化框架的使用。  2、Spring框架的API不会在业务逻辑上出现，由于业务逻辑中没有Spring的API，所以业务逻辑可以从Spring框架快速的移植到其他框架，即与环境无关。  3、通过声明式方式灵活地进行事务的管理  4、Spring作为一个容器，可以管理对象的生命周期、对象与对象之间的依赖关系。可以通过配置文件，来定义对象，以及设置与其他对象的依赖关系  5、Spring是轻量级框架,基本的版本大约2M  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 |

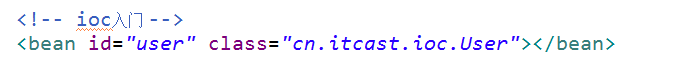
## IOC和DI的区别？

依赖注入和控制反转是对同一件事情的不同描述，从某个方面讲，就是它们描述的角度不同。依赖注入是从应用程序的角度在描述，可以把依赖注入描述完整点：应用程序依赖容器创建并注入它所需要的外部资源；

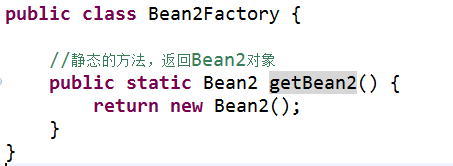
而控制反转是从容器的角度在描述，描述完整点：容器控制应用程序，由容器反向的向应用程序注入应用程序所需要的外部资源。

## spring Bean实例化的方式

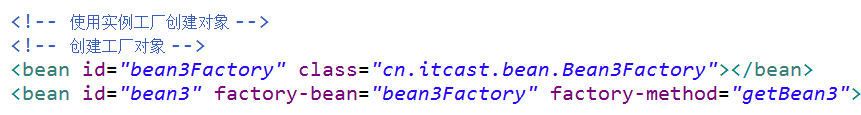
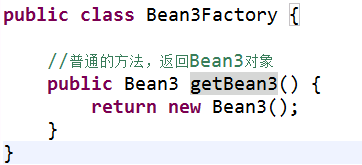
1. 使用类的无参构造函数



1. 使用静态工厂创建



1. 使用实例工厂创建



## Spring的事物管理

（1）编程式事务管理（不用）

（2） 声明式事务管理

1.基于xml配置文件实现

2.基于注解实现

## spring容器加载方式有几种?

三种

//方法1：在类路径下寻找配置文件来实例化容器

ApplicationContext ctx=new ClassPathXmlApplicationContext("classpath:applicationContext.xml");

//如果需要加载多个spring容器,可以定义成数组

ApplicationContext ctx=new ClassPathXmlApplicationContext(new String[]{"classpath:applicationContext.xml"});

//方法2：在文件系统路径下寻找配置文件来实例化容器

ApplicationContext ctx=newFileSystemXmlApplicationContext("G:\\workPlace2\\project2\\Spring\_DI\\config\\applicationContext.xml");

//方法3：使用BeanFactory

BeanFactory ctx=new XmlBeanFactory(new FileSystemResource("G:\\workPlace2\\project2\\Spring\_DI\\config\\applicationContext.xml"));

BeanFactory和ApplicationContext的区别：

ApplicationContext是对BeanFactory的扩展,ApplicationContext提供了更多的功能

如：国际化处理,Bean的自动装配和各种不同应用层的context实现

BeanFactory采取延迟加载,第一次getBean时才会初始化Bean

## BeanFactory和ApplicationContext有什么区别？

**BeanFactory：**

是Spring里面最低层的接口，提供了最简单的容器的功能，只提供了实例化对象和拿对象的功能；

**ApplicationContext：**

应用上下文，继承BeanFactory接口，它是Spring的一各更高级的容器，提供了更多的有用的功能；

1) 国际化（MessageSource）

2) 访问资源，如URL和文件（ResourceLoader）

3) 载入多个（有继承关系）上下文 ，使得每一个上下文都专注于一个特定的层次，比如应用的web层

4) 消息发送、响应机制（ApplicationEventPublisher）

5) AOP（拦截器

**两者装载bean的区别**

**BeanFactory：**

BeanFactory在启动的时候不会去实例化Bean，只有从容器中拿Bean的时候才会去实例化；

**ApplicationContext：**

ApplicationContext在启动的时候就把所有的Bean全部实例化了。它还可以为Bean配置lazy-init=true来让Bean延迟实例化；

## Spring Bean的作用域之间有什么区别？

Spring容器中的bean可以分为5个范围。所有范围的名称都是自说明的，但是为了避免混淆，还是让我们来解释一下：

（1）singleton：这种bean范围是默认的，这种范围确保不管接受到多少个请求，每个容器中只有一个bean的实例，单例的模式由bean factory自身来维护。

（2）prototype：原形范围与单例范围相反，为每一个bean请求提供一个实例。

（3）request：在请求bean范围内会每一个来自客户端的网络请求创建一个实例，在请求完成以后，bean会失效并被垃圾回收器回收。

（4）Session：与请求范围类似，确保每个session中有一个bean的实例，在session过期后，bean会随之失效。

（5）global-session：global-session和Portlet应用相关。当你的应用部署在Portlet容器中工作时，它包含很多portlet。如果你想要声明让所有的portlet共用全局的存储变量的话，那么这全局变量需要存储在global-session中。

## spring的注解有哪些？

Spring的一个核心功能是IOC，就是将Bean初始化加载到容器中，Bean是如何加载到容器，可以使用spring注解方式或者spring XMl配置方式。

@Component:标准的spring bean类。

@Repository:标注一个Dao组件类。

@Service：标注一个业务业务逻辑组件类。

@Controller：标注一个控制器组件类。

@Component可以代替@Repository@Service@Controller因为这三个注解是被@Component标注的

@Autowired：属于Spring的org.springframework.beans.factory.annotation包下，可用于为类的属性、构造器、方法进行注值

@Resource：不属于spring的注解，而是来自于JSR-250位于java.annotation包下，使用该annotation为目标bean指定协作者bean

@PostConstruct和@PreDestroy方法实现初始化和销毁bean之前进行的操作

总结

（1）相同点

@Resource作用相当于@Autowired,都可以标注在字段或属性的setter上。

1. 不同点

A:提供Autowired是spring的注解，@Resource是javax.annotation注解，而是来自于JSR-250，J2EE提供，需要JDK

B:注入方式@Autowired只按照Type注入；@Resource默认按Name自动注入，

也提供按照Type注入；

C:属性

@Autowired注解可用于为类的属性，构造器，方法进行注解。默认情况下，其依赖的对象必须存在（bean可用），如果需要改变这种默认方式，可以设置其required属性为false。还有一个比较重要的点就是，@Autowired注解默认按照类型装配，如果容器中包含多个同一类型的Bean，那么启动容器时会报找不到指定类型bean的异常，解决办法是结合@Qualified注解进行限定，指定注入的bean名称。

@Resource有两个中重要的属性：name和type。name属性指定byName，如果没有指定name属性，当注解标注在字段上，即默认取字段的名称作为bean名称寻找依赖对象，当注解标注在属性的setter方法上，即默认取属性名作为bean名称寻找依赖对象。

需要注意的是，@Resource如果没有指定name属性，并且按照默认的名称仍然找不到依赖对象时， @Resource注解会回退到按类型装配。但一旦指定了name属性，就只能按名称装配了。

d：@Resource注解的使用性更为灵活，可指定名称，也可以指定类型 ；@Autowired注解进行装配容易抛出异常，特别是装配的bean类型有多个的时候，而解决的办法是需要在增加@Qualitied进行限定。

@Bean注解主要用于方法上，有点类似于工厂方法，当使用了@Bean注解，我们可以连续使用多种定义bean时用到的注解，譬如用@Qualifier注解定义工厂方法的名称，用@Scope注解定义该bean的作用域范围，譬如是singleton还是prototype等。

使用@Configuration 来注解类表示类可以被 Spring 的 IoC 容器所使用，作为 bean 定义的资源。

2、举例说明@Component 和 @Configuration

@Configuration

public static class Config {

@Bean

public SimpleBean simpleBean() {

return new SimpleBean();

}

@Bean

public SimpleBeanConsumer simpleBeanConsumer() {

return new SimpleBeanConsumer(simpleBean());

}

}

@Component

public static class Config {

@Bean

public SimpleBean simpleBean() {

return new SimpleBean();

}

@Bean

public SimpleBeanConsumer simpleBeanConsumer() {

return new SimpleBeanConsumer(simpleBean());

}

}

第一个代码正常工作，正如预期的那样，SimpleBeanConsumer将会得到一个单例SimpleBean的链接。第二个配置是完全错误的，因为Spring会创建一个SimpleBean的单例bean，但是SimpleBeanConsumer将获得另一个SimpleBean实例（也就是相当于直接调用new SimpleBean() ，这个bean是不归Spring管理的），既new SimpleBean() 实例是Spring上下文控件之外的。

3、原因总结

使用@ configuration，所有标记为@ bean的方法将被包装成一个CGLIB包装器，它的工作方式就好像是这个方法的第一个调用，那么原始方法的主体将被执行，最终的对象将在spring上下文中注册。所有进一步的调用只返回从上下文检索的bean。

在上面的第二个代码块中，新的SimpleBeanConsumer(simpleBean())只调用一个纯java方法。为了纠正第二个代码块，我们可以这样做

@Component

public static class Config {

@Autowired

SimpleBean;

@Bean

public SimpleBean simpleBean() {

return new SimpleBean();

}

@Bean

public SimpleBeanConsumer simpleBeanConsumer() {

return new SimpleBeanConsumer(simpleBean);

}

}

## @Autowired和@resource有什么区别

@Resource的作用相当于@Autowired

只不过@Autowired按byType自动注入，

而@Resource默认按 byName自动注入罢了。

@Resource有两个属性是比较重要的，分是name和type，Spring将@Resource注解的name属性解析为bean的名字，而type属性则解析为bean的类型。所以如果使用name属性，则使用byName的自动注入策略，而使用type属性时则使用byType自动注入策略。如果既不指定name也不指定type属性，这时将通过反射机制使用byName自动注入策略。

**@Resource装配顺序:**

1、如果同时指定了name和type，则从Spring上下文中找到唯一匹配的bean进行装配，找不到则抛出异常

2、如果指定了name，则从上下文中查找名称（id）匹配的bean进行装配，找不到则抛出异常

3、如果指定了type，则从上下文中找到类型匹配的唯一bean进行装配，找不到或者找到多个，都会抛出异常

4、如果既没有指定name，又没有指定type，则自动按照byName方式进行装配；如果没有匹配，则回退为一个原始类型进行匹配，如果匹配则自动装配；

## Spring怎么集成mybatis

1. Maven导入MyBatis相关依赖，编辑pom.xml
2. 数据库相关bean对象配置，创建resources/spring/applicationContext\_database.xml
3. 配置数据源，<!-- dataSource配置 -->，jdbc四要素

连接数据库的四个要素,驱动类,连接url（统一资源定位符）,用户名,密码

②<!-- mybatis文件配置,mapper.xml文件扫描 -->，配置SqlSessionFactoryBean

③ <!-- mapper注解扫描器配置,扫描@MapperScan注解,自动生成代码对象 -->，配置MapperScannerConfigurer

<!-- 自动扫描com/hys/app/\*\*/dao下的所有dao接口，并实现这些接口，可直接在程序中使用dao接口，不用再获取sqlsession对象 -->

1. SpringMVC

JavaEE体系结构包括四层，从上到下分别是应用层、Web层、业务层、持久层。Struts和SpringMVC是Web层的框架，Spring是业务层的框架，Hibernate和MyBatis是持久层的框架。

## 三层架构与MVC的关系

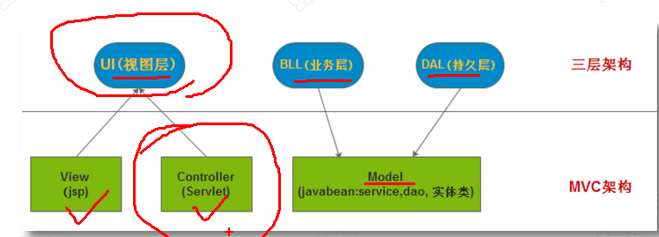
三层架构：

UI:界面层（User Interface layer）包括jsp+controller

BLL: 业务逻辑层（Business Logic Layer）

DAL:数据访问层(Data Access Layer)

BLL+DAL=Model



MVC架构：

M 即Model(模型层)，主要负责出来业务逻辑以及数据库的交互

V 即View(视图层)，主要用于显示数据和提交表单

C 即Controller(控制器)，主要是接收请求，调用Model处理请求，并控制请求转发

他们之间的区别与联系：

联系是：

都是三层，分层的目的是解耦合，按职责划分层次。各层功能独立单一。易于扩展应用，提高开发效率。

区别是：

三层架构适用所有的应用架构，MVC侧重是web应用的使用（jsp和servlet的解耦）。

三层架构中没有控制器Controller. MVC中有控制器层Controller。

# **SpringMVC**

## springMVC的执行原理☆？

## 什么是springMVC？

Spring-Mvc是spring的一个模块，基于MVC的一个框架 无需中间整合层来整合 ；

SpringMVC 框架是以请求为驱动，围绕 Servlet 设计，将请求发给控制器，然后通过模型对象，分派器来展示请求结果视图。其中核心类是 DispatcherServlet，它是一个 Servlet，顶层是实现的Servlet接口。

## SpringMVC怎么使用？

需要在 web.xml 中配置 DispatcherServlet 。并且需要配置 Spring 监听器ContextLoaderListener

## 当初为什么选择springMVC

1，基于MVC架构；功能分工明确。解耦合，

2，容易理解，上手快；使用简单。

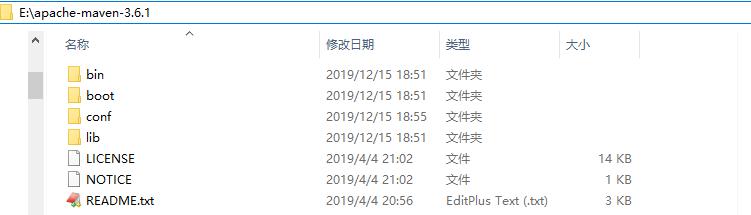
3，作为Spring框架的一部分，能够使用Spring的IoC,Aop等功能。方便的整合MyBatis,Hiberate,JPA等。

4，SpringMVC强化注解的使用，在控制器，Service，Dao都可以使用注解注入对象。方便灵活。

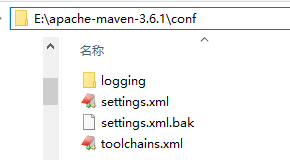
# Maven

## Maven环境配置——从安装修改settings、path、idea

Maven的bin目录本地安装地址E:\apache-maven-3.6.1（ps:我的是3.6.1，因为它新的里面稳定版本，3.6.2坑死我了）



找到Settings文件的地址E:\apache-maven-3.6.1\conf\settings.xml



默认情况，jar会从远程仓库下到/user/name/.m2下，所以我们需要去更改一下本地仓库地址

<localRepository>E:\repository</localRepository>

添加国内依赖仓库（推荐）阿里云的找到mirrors标签，在它里面加上

<mirror>

<id>nexus-aliyun</id>

<name>Nexus aliyun</name>

<url>http://maven.aliyun.com/nexus/content/groups/public/</url>

<mirrorOf>central</mirrorOf>

</mirror>

添加编译默认jdk1.8,省去在idea里面配置java complier到处警告

<profile>

<id>jdk-1.8</id>

<activation>

<activeByDefault>true</activeByDefault>

<jdk>1.8</jdk>

</activation>

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

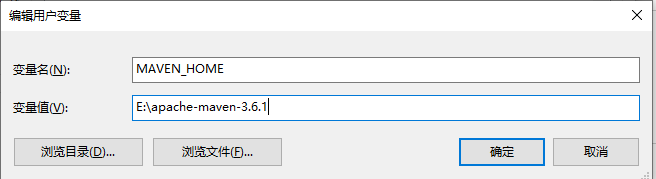
<maven.compiler.target>1.8</maven.compiler.target>

<maven.compiler.compilerVersion>1.8</maven.compiler.compilerVersion>

</properties>

</profile>

新建环境变量MAVEN\_HOME，赋值E:\apache-maven-3.6.1

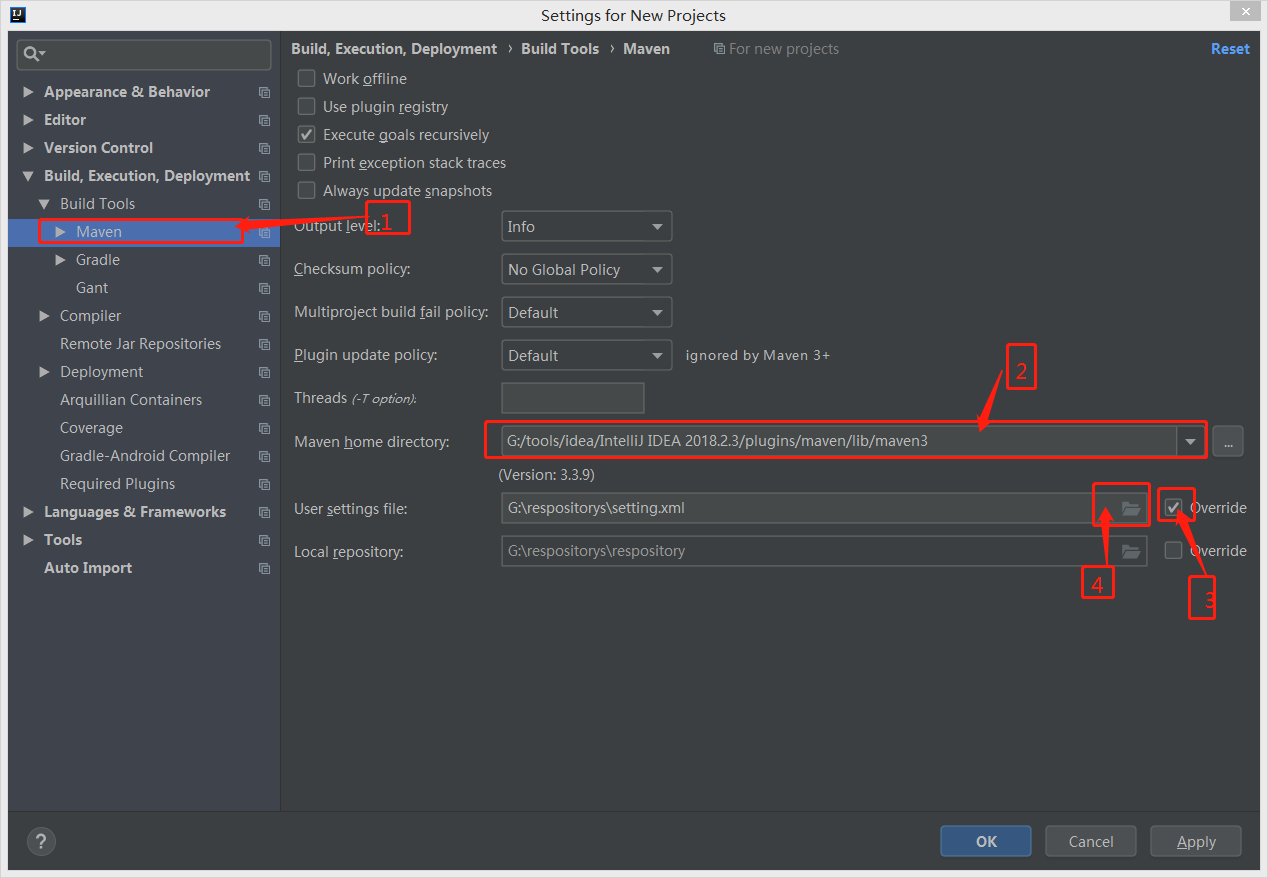
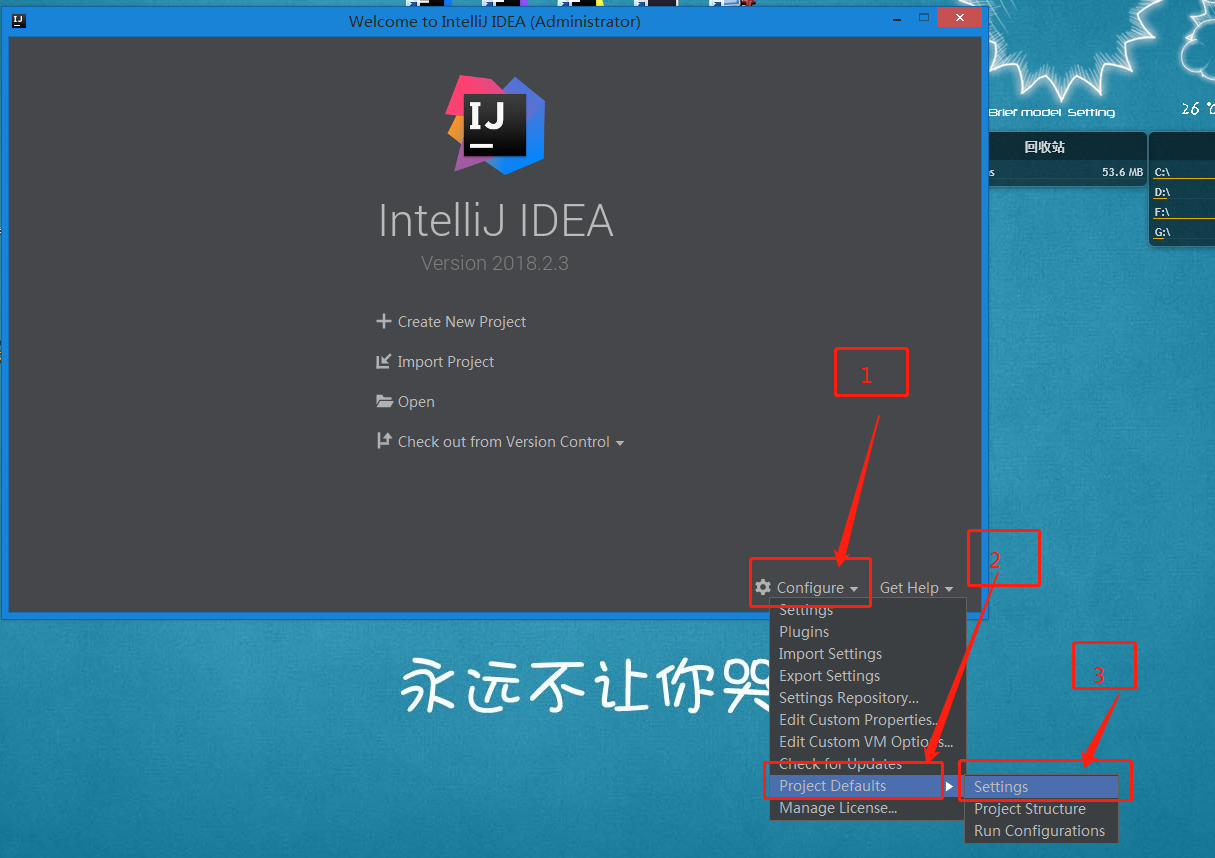


在path里面编辑新建%MAVEN\_HOME%\bin把上面引过来，好处更换maven时候只要更换MAVEN\_HOME值甚至版本号就可以

Ctrl+R输入cmd回车命令mvn -v检查环境是否成功



打开idea



## Settings文件配置详解

|  |
| --- |
| 1.声明规范 <settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 　　　　　　　　　　　http://maven.apache.org/xsd/settings-1.0.0.xsd"> 2.localRepository <!-- 本地仓库的路径。默认值为 -->  <localRepository>/opt/repository</localRepository> 3.interactiveMode <!--Maven是否需要和用户交互以获得输入。如果Maven需要和用户交互以获得输入，则设置成true，反之则应为false。默认为true。-->  <interactiveMode>true</interactiveMode> 4.usePluginRegistry <!--Maven是否需要使用plugin-registry.xml文件来管理插件版本。如果需要让Maven使用文件来管理插件版本，则设为true。默认为false。-->  <usePluginRegistry>false</usePluginRegistry> 5.offline <!--表示Maven是否需要在离线模式下运行。如果构建系统需要在离线模式下运行，则为true，默认为false。当由于网络设置原因或者安全因素，构建服务器不能连接远程仓库的时候，该配置就十分有用。 -->  <offline>false</offline> 6.pluginGroups <!--当插件的组织Id（groupId）没有显式提供时，供搜寻插件组织Id（groupId）的列表。该元素包含一个pluginGroup元素列表，每个子元素包含了一个组织Id（groupId）。当我们使用某个插件，并且没有在命令行为其提供组织Id（groupId）的时候，Maven就会使用该列表。默认情况下该列表包含了org.apache.maven.plugins和org.codehaus.mojo -->  <pluginGroups>  <!--plugin的组织Id（groupId） -->  <pluginGroup>org.codehaus.mojo</pluginGroup>  </pluginGroups> 7.proxies <!--用来配置不同的代理，多代理profiles 可以应对笔记本或移动设备的工作环境：通过简单的设置profile id就可以很容易的更换整个代理配置。 -->  <proxies>  <!--代理元素包含配置代理时需要的信息-->  <proxy>  <!--代理的唯一定义符，用来区分不同的代理元素。-->  <id>myproxy</id>  <!--该代理是否是激活的那个。true则激活代理。当我们声明了一组代理，而某个时候只需要激活一个代理的时候，该元素就可以派上用处。 -->  <active>true</active>  <!--代理的协议。 协议://主机名:端口，分隔成离散的元素以方便配置。-->  <protocol>http</protocol>  <!--代理的主机名。协议://主机名:端口，分隔成离散的元素以方便配置。 -->  <host>proxy.somewhere.com</host>  <!--代理的端口。协议://主机名:端口，分隔成离散的元素以方便配置。 -->  <port>8080</port>  <!--代理的用户名，用户名和密码表示代理服务器认证的登录名和密码。 -->  <username>proxyuser</username>  <!--代理的密码，用户名和密码表示代理服务器认证的登录名和密码。 -->  <password>somepassword</password>  <!--不该被代理的主机名列表。该列表的分隔符由代理服务器指定；例子中使用了竖线分隔符，使用逗号分隔也很常见。-->  <nonProxyHosts>\*.google.com|ibiblio.org</nonProxyHosts>  </proxy>  </proxies> 8.servers <!--配置服务端的一些设置。一些设置如安全证书不应该和pom.xml一起分发。这种类型的信息应该存在于构建服务器上的settings.xml文件中。-->  <servers>  <!--服务器元素包含配置服务器时需要的信息 -->  <server>  <!--这是server的id（注意不是用户登陆的id），该id与distributionManagement中repository元素的id相匹配。-->  <id>server001</id>  <!--鉴权用户名。鉴权用户名和鉴权密码表示服务器认证所需要的登录名和密码。 -->  <username>my\_login</username>  <!--鉴权密码 。鉴权用户名和鉴权密码表示服务器认证所需要的登录名和密码。密码加密功能已被添加到2.1.0 +。详情请访问密码加密页面-->  <password>my\_password</password>  <!--鉴权时使用的私钥位置。和前两个元素类似，私钥位置和私钥密码指定了一个私钥的路径（默认是${user.home}/.ssh/id\_dsa）以及如果需要的话，一个密语。将来passphrase和password元素可能会被提取到外部，但目前它们必须在settings.xml文件以纯文本的形式声明。 -->  <privateKey>${usr.home}/.ssh/id\_dsa</privateKey>  <!--鉴权时使用的私钥密码。-->  <passphrase>some\_passphrase</passphrase>  <!--文件被创建时的权限。如果在部署的时候会创建一个仓库文件或者目录，这时候就可以使用权限（permission）。这两个元素合法的值是一个三位数字，其对应了unix文件系统的权限，如664，或者775。 -->  <filePermissions>664</filePermissions>  <!--目录被创建时的权限。 -->  <directoryPermissions>775</directoryPermissions>  </server>  </servers> 9.mirrors <!--为仓库列表配置的下载镜像列表。高级设置请参阅镜像设置页面 -->  <mirrors>  <!--给定仓库的下载镜像。 -->  <mirror>  <!--该镜像的唯一标识符。id用来区分不同的mirror元素。 -->  <id>planetmirror.com</id>  <!--镜像名称 -->  <name>PlanetMirror Australia</name>  <!--该镜像的URL。构建系统会优先考虑使用该URL，而非使用默认的服务器URL。 -->  <url>http://downloads.planetmirror.com/pub/maven2</url>  <!--被镜像的服务器的id。例如，如果我们要设置了一个Maven中央仓库（http://repo.maven.apache.org/maven2/）的镜像，就需要将该元素设置成central。这必须和中央仓库的id central完全一致。-->  <mirrorOf>central</mirrorOf>  </mirror>  </mirrors> 10.profiles <!--根据环境参数来调整构建配置的列表。settings.xml中的profile元素是pom.xml中profile元素的裁剪版本。它包含了id，activation, repositories, pluginRepositories和 properties元素。这里的profile元素只包含这五个子元素是因为这里只关心构建系统这个整体（这正是settings.xml文件的角色定位），而非单独的项目对象模型设置。如果一个settings中的profile被激活，它的值会覆盖任何其它定义在POM中或者profile.xml中的带有相同id的profile。 -->  <profiles>  <!--根据环境参数来调整的构件的配置-->  <profile>  <!--该配置的唯一标识符。 -->  <id>test</id> 11.Activation <!--自动触发profile的条件逻辑。Activation是profile的开启钥匙。如POM中的profile一样，profile的力量来自于它能够在某些特定的环境中自动使用某些特定的值；这些环境通过activation元素指定。activation元素并不是激活profile的唯一方式。settings.xml文件中的activeProfile元素可以包含profile的id。profile也可以通过在命令行，使用-P标记和逗号分隔的列表来显式的激活（如，-P test）。-->  <activation>  <!--profile默认是否激活的标识-->  <activeByDefault>false</activeByDefault>  <!--当匹配的jdk被检测到，profile被激活。例如，1.4激活JDK1.4，1.4.0\_2，而!1.4激活所有版本不是以1.4开头的JDK。-->  <jdk>1.5</jdk>  <!--当匹配的操作系统属性被检测到，profile被激活。os元素可以定义一些操作系统相关的属性。-->  <os>  <!--激活profile的操作系统的名字 -->  <name>Windows XP</name>  <!--激活profile的操作系统所属家族(如 'windows') -->  <family>Windows</family>  <!--激活profile的操作系统体系结构 -->  <arch>x86</arch>  <!--激活profile的操作系统版本-->  <version>5.1.2600</version>  </os>  <!--如果Maven检测到某一个属性（其值可以在POM中通过${name}引用），其拥有对应的name = 值，Profile就会被激活。如果值字段是空的，那么存在属性名称字段就会激活profile，否则按区分大小写方式匹配属性值字段-->  <property>  <!--激活profile的属性的名称-->  <name>mavenVersion</name>  <!--激活profile的属性的值 -->  <value>2.0.3</value>  </property>  <!--提供一个文件名，通过检测该文件的存在或不存在来激活profile。missing检查文件是否存在，如果不存在则激活profile。另一方面，exists则会检查文件是否存在，如果存在则激活profile。-->  <file>  <!--如果指定的文件存在，则激活profile。 -->  <exists>${basedir}/file2.properties</exists>  <!--如果指定的文件不存在，则激活profile。-->  <missing>${basedir}/file1.properties</missing>  </file>  </activation> 12.Repositories <repository>  <id>nexus</id>  <name>local private nexus</name>  <url>http://localhost:8081/nexus/content/groups/public</url>  <releases><enabled>true</enabled></releases>  <snapshots><enabled>true</enabled></snapshots>  </repository>  </repositories>  <!--远程仓库列表，它是Maven用来填充构建系统本地仓库所使用的一组远程项目。 -->  <repositories>  <!--包含需要连接到远程仓库的信息 -->  <repository>  <!--远程仓库唯一标识-->  <id>codehausSnapshots</id>  <!--远程仓库名称 -->  <name>Codehaus Snapshots</name>  <!--如何处理远程仓库里发布版本的下载-->  <releases>  <!--true或者false表示该仓库是否为下载某种类型构件（发布版，快照版）开启。 -->  <enabled>false</enabled>  <!--该元素指定更新发生的频率。Maven会比较本地POM和远程POM的时间戳。这里的选项是：always（一直），daily（默认，每日），interval：X（这里X是以分钟为单位的时间间隔），或者never（从不）。 -->  <updatePolicy>always</updatePolicy>  <!--当Maven验证构件校验文件失败时该怎么做-ignore（忽略），fail（失败），或者warn（警告）。-->  <checksumPolicy>warn</checksumPolicy>  </releases>  <!--如何处理远程仓库里快照版本的下载。有了releases和snapshots这两组配置，POM就可以在每个单独的仓库中，为每种类型的构件采取不同的策略。例如，可能有人会决定只为开发目的开启对快照版本下载的支持。参见repositories/repository/releases元素-->  <snapshots>  <enabled/><updatePolicy/><checksumPolicy/>  </snapshots>  <!--远程仓库URL，按protocol://hostname/path形式 -->  <url>http://snapshots.maven.codehaus.org/maven2</url>  <!--用于定位和排序构件的仓库布局类型-可以是default（默认）或者legacy（遗留）。Maven 2为其仓库提供了一个默认的布局；然而，Maven 1.x有一种不同的布局。我们可以使用该元素指定布局是default（默认）还是legacy（遗留）。 -->  <layout>default</layout>  </repository>  </repositories>  <!--发现插件的远程仓库列表。仓库是两种主要构件的家。第一种构件被用作其它构件的依赖。这是中央仓库中存储的大部分构件类型。另外一种构件类型是插件。Maven插件是一种特殊类型的构件。由于这个原因，插件仓库独立于其它仓库。pluginRepositories元素的结构和repositories元素的结构类似。每个pluginRepository元素指定一个Maven可以用来寻找新插件的远程地址。-->  <pluginRepositories>  <!--包含需要连接到远程插件仓库的信息.参见profiles/profile/repositories/repository元素的说明-->  <pluginRepository>  <releases>  <enabled/><updatePolicy/><checksumPolicy/>  </releases>  <snapshots>  <enabled/><updatePolicy/><checksumPolicy/>  </snapshots>  <id/><name/><url/><layout/>  </pluginRepository>  </pluginRepositories>  </profile>  </profiles> 13.activeProfiles 默认配置：  <activeProfiles>  <activeProfile>dev</activeProfile>  <!-- <activeProfile>devitcast</activeProfile>-->  </activeProfiles>  配置解释：  <!--手动激活profiles的列表，按照profile被应用的顺序定义activeProfile。 该元素包含了一组activeProfile元素，每个activeProfile都含有一个profile id。任何在activeProfile中定义的profile id，不论环境设置如何，其对应的profile都会被激活。如果没有匹配的profile，则什么都不会发生。例如，env-test是一个activeProfile，则在pom.xml（或者profile.xml）中对应id的profile会被激活。如果运行过程中找不到这样一个profile，Maven则会像往常一样运行 -->  <activeProfiles>  <!-- -->  <activeProfile>env-test</activeProfile>  </activeProfiles>  </settings> |

## Pom.xml文件的详解

<https://blog.csdn.net/qq_33363618/article/details/79438044>

## Maven的生命周期？

## 什么是Maven的依赖冲突？

|  |
| --- |
| 作者 | 桔子链接 | segmentfault.com/a/1190000017542396  1、何为依赖冲突Maven是个很好用的依赖管理工具，但是再好的东西也不是完美的。Maven的依赖机制会导致Jar包的冲突。举个例子，现在你的项目中，使用了两个Jar包，分别是A和B。现在A需要依赖另一个Jar包C，B也需要依赖C。但是A依赖的C的版本是1.0，B依赖的C的版本是2.0。这时候，Maven会将这1.0的C和2.0的C都下载到你的项目中，这样你的项目中就存在了不同版本的C，这时Maven会依据依赖路径最短优先原则，来决定使用哪个版本的Jar包，而另一个无用的Jar包则未被使用，这就是所谓的依赖冲突。在大多数时候，依赖冲突可能并不会对系统造成什么异常，因为Maven始终选择了一个Jar包来使用。但是，不排除在某些特定条件下，会出现类似找不到类的异常，所以，只要存在依赖冲突，在我看来，最好还是解决掉，不要给系统留下隐患 |

## 怎么解决jar冲突

|  |
| --- |
|  |

# 消息队列

## ActiveMQ

### 大幅度

### 好地方

## RabbitMQ

## RocketMQ

## Kafka

# 服务器

## Nginx

## Tomcat

# 开发工具idea和eclipse（主要是idea）

## Idea和eclipse的区别？

## Idea开发有什么好处？

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——Ctrl+鼠标滚轮放大、缩小代码

File->Settings->keymap->配置Decrease Font Size  字体缩小 ,右键选择Add Mouse Shortcut, 鼠标向下滚动+Ctrl  就OK！->配置Increase Font Size  字体放大 ,右键选择Add Mouse Shortcut, 鼠标向上滚动+Ctrl  就OK->测试，用鼠标向上或向下滚动就放大或缩小字体哦！

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——快捷键

Ctrl + Y 删除当前行

Ctrl + X 剪切当前行

Ctrl + F 在当前页查找

Ctrl + R 在当前页替换

Ctrl + D 复制当前行

比较两个jar包中文件

选中一个或两个jar后，Ctrl + D

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——类注释模板

File -> Other Setting -> Default Setting打开默认设置

Editor -> File and Code Templates -> Files -> Class 找到类注释模板，在public class前面添加模板注释

/\*\*

\* @program: ${PROJECT\_NAME}->${NAME}

\* @description: ${description}

\* @author: qiuyu

\* @create: ${YEAR}-${MONTH}-${DAY} ${HOUR}:${MINUTE}

\*\*/

*/\*\*  
 \* 京颐 工具类  
 \*  
 \** ***@author*** *龚旺旺  
 \** ***@email*** *gww.vip@qq.com  
 \** ***@date*** *2018/12/11  
 \** ***@copyright*** *<url>http://www.chongies.com/</url>  
 \*/*

/\*\*  
\* @program: **${**PROJECT\_NAME**}**->**${**NAME**}**\* @description: **${**description**}**\* @author: 龚旺旺  
\* @create: **${**YEAR**}**-**${**MONTH**}**-**${**DAY**} ${**HOUR**}**:**${**MINUTE**}**\*\*/

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——方法注释

打开Setting

Editor -> Live Templates -> 点击右边加号为自己添加一个Templates Group

然后选中自己的Group再次点击加号添加Live Templates

/\*\*

\* @Description: $description$

\* @Param: $params$

\* @return: $returns$

\* @Author: qiuyu

\* @Date: $date$

\*/

选择适用范围，选择所有就可以了

点击edit variables，选择表达式

最后在方法上面直接输入/ + 你设置的Abbreviation快捷键再按tab键就直接生成

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——文件历史版本

右击项目或文件 -> Local History -> Show History可以查看文件增删改的历史记录。

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——连接数据库

打开Database窗口

添加数据库

配置IP端口和账号密码

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——根据表自动生成POJO

连接好数据库后选择表，右击表 -》 Generate POJOs.groovy即可.

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——宽屏模式

这个会让最左侧项目栏贯穿上下，和Myeclipse布局一样。

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——插件

Setting -> Plugins

### ****阿里巴巴编码规范****

Alibaba Java Coding Guidelines

很好用的神器，会自动扫描不规范的代码。

### ****FindBugs****

这个插件可以检测可能的bug及不规范的位置。可以避免低级bug，而且让代码更加整洁规范。

### ****翻译插件****

Translation

使用：选中单词或者段落ctrl+shift+y翻译，ctrl+shift+s切换翻译源

### ****mybatis跳转插件****

Free Mybatis plugin

Mapper接口跳转到XML文件的插件

### ****Lombok插件****

IntelliJ Lombok plugin

自动生成java模板代码，Lombok的pom.xml 文件配置，在自定义的pojo中很有用。

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.16.18</version>

<scope>provided</scope>

</dependency>

### ****GenAllSetter****

在Java方法中, 根据 new 关键词, 为Java Bean 生成所有Setter方法。

按GenAllSetter键两次, 会为Setter方法生成默认值。

### mybatisGenerator

根据generator.xml生成pojo，mapper

### ****GenDaoCode****

一键生成 service dao xml sql（建表语句）

* 根据Pojo 文件一键生成 Dao，Service，Xml，Sql文件。
* Pojo文件更新后一键更新对应的Sql和mybatis xml文件。
* 提供insert，insertList，update，select，delete五种方法。
* 能够批量生成多个Pojo的对应的文件。
* 自动将pojo的注释添加到对应的Sql文件的注释中。
* 目前支持MySQL + Java

### GsonFormat

一键根据json字符生成java类。

新建一个空的pojo的java文件，然后Code -> Generate -> GsonFormat（快捷键Alt + S）

### ****CodeGlance****

在右侧生成代码地图，类似于Ultraedit中右侧的代码地图。

### ****RestfulToolkit****

RequestMapping搜索神器，并可以直接测试

View -> Tool Windows -> RestServices 加入这个插件之后, 在做java借口搜索和测试的时候方便很多

### Jrebel

热部署神器，虽然收费但一定要装。[具体参考](https://blog.csdn.net/is_chenzh/article/details/82148157)

装好之后再View -> Tool Windows中找到Jrebel的几个功能

先把Tomcat配置好Update和frame要配置成Update classes and resources，

VM options最好配置下永久代（方法区）大小PermSize，不然Jrebel可能会内存溢出。

-Dfile.encoding=UTF-8 -server -XX:PermSize=64m -XX:MaxPermSize=256m

Deployment要部署Web exploded，不能部署war文件。不然热部署无效。

在Setting中设置Jrebel的Time Saved statistics设置短一点。

配置好后用Jrebel启动tomcat

### Grep console

让日志打印的警告和报错显示颜色

## [IDEA常用功能](https://www.cnblogs.com/aeolian/p/9465377.html)——非主要功能性插件

### Material Theme UI

这是一款主题插件，可以让你的ide的图标变漂亮，配色搭配的很到位，还可以切换不同的颜色，甚至可以自定义颜色。默认的配色就很漂亮了，如果需要修改配色，可以在工具栏中Tools->Material Theme然后修改配色等。

### Background image Plus

这是一款可以设置idea背景图片的插件，不但可以设置固体的图片，还可以设置一段时间后随机变化背景图片，以及设置图片的透明度等等。

View -> Set backgroundImage

### activate-power-mode

这是一款让你在编码的时候，整个屏幕都为之颤抖的插件。

Window > activate-power-mode 就可以开启或者关闭相对应的效果

### Nyan progress bar

这是一个将你idea中的所有的进度条都变成萌新动画的小插件。

### Rainbow Brackets

彩虹颜色的括号  看着很舒服 敲代码效率变高

## idea配置maven java compiler为1.8

|  |
| --- |
| [java maven compiler作用是设置编译时候默认1.8](https://www.cnblogs.com/DjangoBlogs/p/9174008.html) 方法一：  <build>  <plugins>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.1</version>  <configuration>  <source>1.8</source>  <target>1.8</target>  </configuration>  </plugin>  </plugins>  </build>  方法二（推荐）：  在maven配置文件settings.xml中设置  <profile>  <id>jdk-1.8</id>  <activation>  <activeByDefault>true</activeByDefault>  <jdk>1.8</jdk>  </activation>  <properties>  <maven.compiler.source>1.8</maven.compiler.source>  <maven.compiler.target>1.8</maven.compiler.target>  <maven.compiler.compilerVersion>1.8</maven.compiler.compilerVersion>  </properties>  </profile> |

## eclipse

# 日志处理

|  |
| --- |
| logger.debug方法正确使用方式 原创[chuixue24](https://me.csdn.net/chuixue24) 发布于2019-01-17 11:29:26 阅读数 3820  收藏  展开  看开源框架源代码时，经常看到这样的代码：  if (logger.isDebugEnabled()) {        logger.debug(……); }  有没有思考过为什么是先判断再输出，而不是直接输出？  在阿里巴巴开发手册中有这样的规范：  **说明：logger.debug("Processing trade with id: " + id + " and symbol: " + symbol); 如果日志级别是warn，上述日志不会打印，但是会执行字符串拼接操作，如果symbol是对象，会执行toString()方法，浪费了系统资源，执行了上述操作，最终日志却没有打印。 正例：（条件）if (logger.isDebugEnabled()) {logger.debug("Processing trade with id: " + id + " and symbol: " + symbol);}正例：（占位符）logger.debug("Processing trade with id: {} and symbol : {} ", id, symbol);** |

# 前端

## ajax的属性

|  |
| --- |
|  |

## Ajax怎么跨域

## Bootstrap和jQuery有什么区别？

|  |
| --- |
| Bootstrap以及jQuery主要以Web 场景为主 也就是网站的页面展示层 能够提供很多艳丽的效果  　　Bootstrap是一个前台框架包含css和一些jquery插件，依赖与jquery，所以必须一起使用  　　bootstrap是依赖jquery写的扩展，你要使用bootstrap，必须要先引入juqery你不引入jquery，bootstrap的功能你还不能用  　　bootstrap定义了布局、间距等，还提供了less进行管理，可以看作是css规范、框架。  　　利用他提供的样式随意组合，基本上能满足了一些页面的样式需求。  　　用Jquery 的话， 如果要做一个网站，PC 端用jQuery-ui 手机端用jQuery mobile【需要做两套网站】  　　如果用bootstrap的话，【只需要一套就够了】。  　　因为bootstrap 是响应式布局，根据你的浏览设备不同，给你显示页面的效果也不同。当然如果你需要 支持IE 6---IE7 最好还是用 jQuery 低版本吧。  　　现在比较流行的前端框架有bootstrap + angularJS  jQuery是JavaScript的工具集，说白了jQuery就是JavaScript。  而Bootstrap是一个开发框架，集成了很多现成的网页格式和网页布局，封装了很多CSS样式和JS代码，可以拿来直接用，只需要把里面想换成自己的内容替换掉。  jquery是js脚本库，主要是通过简化js操作html元素及前端数据的。  bootstrap是前端页面设计数据显示框架，主要是它的UI部分，显示效果很不错。  bootstrap = 样式库 + 一堆jQuery插件  顺便解释一句，如果你只需要bootstrap的样式，你完全不用引入jQuery和bootstrap的js文件 |

# 常见的7种设计模式

|  |
| --- |
| <https://www.cnblogs.com/Diyo/p/11415588.html> |

# 算法题

## 阶乘

// int i = 3;

long b = 1;

StringBuilder sb = new StringBuilder();

for (long i=7;i > 0;i--){

b=b\*i;//3=3\*1\*2

if("".equals(sb.toString())){

sb.append(i);

}else{

sb.append("\*"+i);

}

}

System.out.println("b = "+sb+" = "+b);

## 1+2+3+…

int a=0;

StringBuilder sbd = new StringBuilder();

for (int j =i;j <= i&&j!=0;j--){

a +=j;

if ("".equals(sbd.toString())){

sbd.append(j);

}else{

sbd.append("+"+j);

}

}

System.out.println(sbd.toString());

# 正则表达式

|  |
| --- |
|  |

# 操作系统与计算机网络

|  |
| --- |
|  |

# JQuery

# css

# javascript

# Html

# vue

# K8S一键部署

# Linux

## 常用的命令

**1、查看IP：**

1.进入Linux 系统，在主页面空旷的地方右键，选择 打开终端。

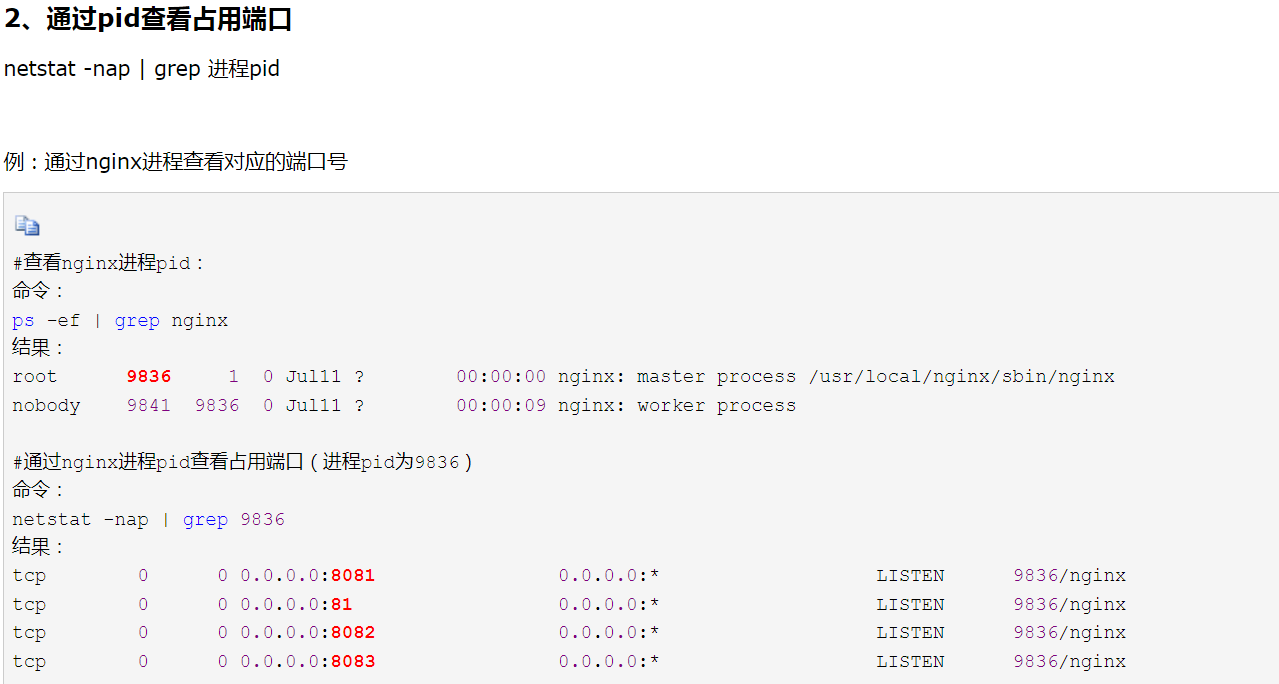
2. 在显示的界面中输入 ifconfig -a  就可以查看到Linux 的地址了。

快速查看IP 和修改：

1.点击应用程序 ->选择系统工具-》选择设置 ->找到网络-》点击进入

2.有线就选择有线。选择后，右侧会出现 ip 地址的详细信息。选择 ipv4地址即可。

3.点击右下角的设置图标，可以自行设置ip.



Linux中编辑文件 命令 快速到最后一行 快速到第一行

Vim 文件名- （是vi命令的升级版本）

Vi 编辑文件（文件如果没有就在编辑并保存后给创建）

进入之后gg调到第一行,

G跳到最后一行

输入/查询字符 可以查找 按n下一个

cd 切换目录

ll是把当前目录下的文件和文件夹都给创建出来

ls是把当前目录的文件都给以列表形式显示出来

CP是复制

Mv是移动和重命名的

Mkdir是创建文件夹

Touch是创建文件

rz是上传文件

sz是从Linux上下载到本地路径

**pwd­­­­­­­­­­­­­­­­­­­­­­­** 目前所在的工作目录的绝对路径名称

**线程与进程**

**进程:电脑中运行的程序就是一个进程**

**线程:线程是进程中的一个执行单元,一个进程至少有一个线程,如果有多个线程,那么这个进程就是多线程的**

注意:

1. 操作系统是以进程 为单位分配系统资源(系统资源主要是CPU和内存), 如果这个进程 有多个线程,这些线程共享进程的资源
2. 线程之间是相互独立的, 每个线程都有自己的线程（栈空间）, **共享堆区与方法区**

27. SQL优化

SQL优化：

1) 对查询就行优化，避免全表扫描，在where 及 order by 涉及的列上建立索引。

2) 对于主键id，可以使用数据库自带的自增ID，代替UUID，因为，自增ID的连续性可以增加查询和写入的效率。

3) 避免在where中对字段进行空值的判断，以免放弃索引进而全表扫描，对于空值，可以用0替代

4) 尽量避免在where中对字段进行函数操作

5) 避免频繁创建删除临时表

6) 不要使用select \*

7) 对于多张大数据表进行join，要分页再join

### 存储引擎

***InnoDB***：支持事务处理，支持外键，支持崩溃修复能力和并发控制。如果需要**对事务的完整性要求比较高**（比如银行），**要求实现并发控制**（比如售票），那选择InnoDB有很大的优势。如果需要**频繁的更新、删除**操作的数据库，也可以选择InnoDB，因为支持事务的提交（commit）和回滚（rollback）。

***MyISAM***：插入数据快，空间和内存使用比较低。如果表主要是**用于插入新记录和读出记录**，那么选择MyISAM能实现处理高效率。如果应用的完整性、并发性要求比 较低，也可以使用。

***MEMORY***：所有的数据都在内存中，数据的处理速度快，但是安全性不高。如果需要**很快的读写速度**，对数据的安全性要求较低，可以选择MEMOEY。它对表的大小有要求，不能建立太大的表。所以，这类数据库只使用在相对较小的数据库表。

注意，同一个数据库也可以使用多种存储引擎的表。如果一个表要求比较高的事务处理，可以选择InnoDB。这个数据库中可以将查询要求比较高的表选择MyISAM存储。如果该数据库需要一个用于查询的临时表，可以选择MEMORY存储引擎。

1. 修改表引擎方法  
   alter table table\_name engine=innodb;  
   2、查看系统支持的[存储引擎](https://www.baidu.com/s?wd=%E5%AD%98%E5%82%A8%E5%BC%95%E6%93%8E&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)  
   show engines;  
   3、查看表使用的[存储引擎](https://www.baidu.com/s?wd=%E5%AD%98%E5%82%A8%E5%BC%95%E6%93%8E&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)  
   两种方法：  
   a、show table status from db\_name where name='table\_name';  
   b、show create table table\_name;  
   如果显示的格式不好看，可以用\g代替行尾分号  
   有人说用第二种方法不准确  
   我试了下，关闭掉原先默认的Innodb引擎后根本无法执行show create table table\_name指令，因为之前建的是Innodb表，关掉后默认用MyISAM引擎，导致Innodb表数据无法被正确读取。  
   4 关闭Innodb引擎方法  
   关闭mysql服务： net stop mysql  
   找到mysql安装目录下的my.ini文件：  
   找到default-storage-engine=INNODB 改为default-storage-engine=MYISAM  
   找到#skip-innodb 改为skip-innodb  
   启动mysql服务：net start mysql
2. <https://nanairo.co/zh-hans/videos/2850>

## Ls和ll的区别？

**1**. [ls](https://www.baidu.com/s?wd=ls&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao) 命令可以说是linux下最常用的命令之一。ll不是命令，是[ls](https://www.baidu.com/s?wd=ls&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao) [-l](https://www.baidu.com/s?wd=-l&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)的别名相当于windows里的快捷方式。所以"ll"和“[ls](https://www.baidu.com/s?wd=ls&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao) [-l](https://www.baidu.com/s?wd=-l&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)”的功能是相同的。

2.ll 列出来的结果详细，有时间，是否可读写等信息，像windows里的 详细信息；ls 只列出文件名或目录名 就像windows里的 列表。

## Linux关机，重启，注销

关机：

　　shutdown -h now  #立刻关机重启，工作中常用

　　shutdown -h +1    #1分钟后关机

　　init 0

　　halt                        #立即停止系统，需要人工关闭电源

　　halt -p                    #

　　poweroff　　　　  #立即停止系统，并且关闭电源

重启：

　　reboot　　　　　　#工作中常用

　　shutdown -r now      #工作中常用

　　shutdown -r +1　　 #一分钟后重启

　　init 6

注销：

　　logout

　　exit　　　　　　#工作中常用

　　ctrl+d　　　　　#工作中常用

## Vim怎么编辑文件内容（文字内容认真阅读）？

编辑模式：  
在一般模式（非编辑模式）中可以进行删除、复制、粘贴等的操作，但是却无法编辑文件的内容，只有当到你按下【i, I, o, O, a, A, r, R】等任何一个字母之后才会进入编辑模式。这时候屏幕的左下方会出现【INSERT或 REPLACE】的字样，此时才可以进行编辑。而如果要回到一般模式时， 则必须要按下【Esc】即可退出编辑模式。

命令行模式：  
输入【 : / ? 】三个中的任何一个，就可以将光标移动到最底下那一行。在这个模式中， 可以提供查找、读取、存盘、替换字符、离开vi、显示行号等的动作则是在此模式中完成的！

误删，备件自动备份

当我们在使用vim编辑时，vim会在与被编辑的文件的目录下，再建立一个名为 .filename.swp的文件。如果你的系统因为某些原因断线了， 导致你编辑的文件还没有保存，这个时候 .filenam.swp 就能够发会救援的功能了。

## 45道Linux面试题

1. **问题一：**
2. 绝对路径用什么符号表示？当前目录、上层目录用什么表示？主目录用什么表示? 切换目录用什么命令？
3. **答案：**  
   绝对路径：如/etc/init.d  
   当前目录和上层目录：./  ../  
   主目录：~/  
   切换目录：cd  
     
   **问题二：**
4. 怎么查看当前进程？怎么执行退出？怎么查看当前路径？  
   **答案：**  
   查看当前进程：ps  
   执行退出：exit  
   查看当前路径：pwd  
     
   **问题三：**
5. 怎么清屏？怎么退出当前命令？怎么执行睡眠？怎么查看当前用户 id？查看指定帮助用什么命令？  
   **答案：**  
   清屏：clear  
   退出当前命令：ctrl+c 彻底退出  
   执行睡眠 ：ctrl+z 挂起当前进程fg 恢复后台  
   查看当前用户 id：”id“：查看显示目前登陆账户的 uid 和 gid 及所属分组及用户名  
   查看指定帮助：如 man adduser 这个很全 而且有例子；adduser --help 这个告诉你一些常用参数；info adduesr；  
     
   **问题四：**
6. Ls 命令执行什么功能？可以带哪些参数，有什么区别？  
   **答案：**  
   ls 执行的功能：列出指定目录中的目录，以及文件  
   哪些参数以及区别：a 所有文件l 详细信息，包括大小字节数，可读可写可执行的权限等  
     
   **问题五：**
7. 建立软链接(快捷方式)，以及硬链接的命令。  
   **答案：**  
   软链接：ln -s slink source  
   硬链接：ln link source  
     
   **问题六：**
8. 目录创建用什么命令？创建文件用什么命令？复制文件用什么命令？  
   **答案：**  
   创建目录：mkdir  
   创建文件：典型的如 touch，vi 也可以创建文件，其实只要向一个不存在的文件输出，都会创建文件  
   复制文件：cp 7. 文件权限修改用什么命令？格式是怎么样的？  
   文件权限修改：chmod  
   格式如下：
9. $ chmod u+x file 给 file 的属主增加执行权限  
   $ chmod 751 file 给 file 的属主分配读、写、执行(7)的权限，给 file 的所在组分配读、执行(5)的权限，给其他用户分配执行(1)的权限  
   $ chmod u=rwx,g=rx,o=x file 上例的另一种形式  
   $ chmod =r file 为所有用户分配读权限  
   $ chmod 444 file 同上例  
   $ chmod a-wx,a+r file同上例  
   $ chmod -R u+r directory 递归地给 directory 目录下所有文件和子目录的属主分配读的权限
10. **问题八：**
11. 查看文件内容有哪些命令可以使用？  
    **答案：**  
    vi 文件名 #编辑方式查看，可修改  
    cat 文件名 #显示全部文件内容  
    more 文件名 #分页显示文件内容  
    less 文件名 #与 more 相似，更好的是可以往前翻页  
    tail 文件名 #仅查看尾部，还可以指定行数  
    head 文件名 #仅查看头部,还可以指定行数  
      
    **问题九：**
12. 随意写文件命令？怎么向屏幕输出带空格的字符串，比如”hello world”?
13. **答案：**
14. 写文件命令：vi
15. 向屏幕输出带空格的字符串:echo hello world
16. **问题十：**
17. 终端是哪个文件夹下的哪个文件？黑洞文件是哪个文件夹下的哪个命令？  
    **答案：**  
    终端  /dev/tty
18. 黑洞文件  /dev/null  
      
    **问题十一：**
19. 移动文件用哪个命令？改名用哪个命令？  
    **答案：**  
    mv mv  
      
    **问题十二：**
20. 复制文件用哪个命令？如果需要连同文件夹一块复制呢？如果需要有提示功能呢？  
    **答案：**  
    cp cp -r  ？？？？  
      
    **问题十三：**
21. 删除文件用哪个命令？如果需要连目录及目录下文件一块删除呢？删除空文件夹用什么命令？  
    **答案：**  
    rm rm -r rmdir  
      
    **问题十四：**
22. Linux 下命令有哪几种可使用的通配符？分别代表什么含义?  
    **答案：**  
    “？”可替代单个字符。
23. “\*”可替代任意多个字符。
24. 方括号“[charset]”可替代 charset 集中的任何单个字符，如[a-z]，[abABC]
25. **问题十五：**
26. 用什么命令对一个文件的内容进行统计？(行号、单词数、字节数)  
    **答案：**
27. wc 命令 - c 统计字节数 - l 统计行数 - w 统计字数。  
      
    **问题十六：**
28. Grep 命令有什么用？如何忽略大小写？如何查找不含该串的行?  
    **答案：**  
    是一种强大的文本搜索工具，它能使用正则表达式搜索文本，并把匹 配的行打印出来。  
    grep [stringSTRING] filename grep [^string] filename  
      
    **问题十七：**
29. Linux 中进程有哪几种状态？在 ps 显示出来的信息中，分别用什么符号表示的？  
    **答案：**  
    （1）、不可中断状态：进程处于睡眠状态，但是此刻进程是不可中断的。不可中断， 指进程不响应异步信号。  
    （2）、暂停状态/跟踪状态：向进程发送一个 SIGSTOP 信号，它就会因响应该信号 而进入 TASK\_STOPPED 状态;当进程正在被跟踪时，它处于 TASK\_TRACED 这个特殊的状态。  
    “正在被跟踪”指的是进程暂停下来，等待跟踪它的进程对它进行操作。
30. （3）、就绪状态：在 run\_queue 队列里的状态
31. （4）、运行状态：在 run\_queue 队列里的状态  
    （5）、可中断睡眠状态：处于这个状态的进程因为等待某某事件的发生（比如等待 socket 连接、等待信号量），而被挂起  
    （6）、zombie 状态（僵尸）：父亲没有通过 wait 系列的系统调用会顺便将子进程的尸体（task\_struct）也释放掉  
    （7）、退出状态
32. D 不可中断 Uninterruptible（usually IO）  
    R 正在运行，或在队列中的进程  
    S 处于休眠状态  
    T 停止或被追踪  
    Z 僵尸进程  
    W 进入内存交换（从内核 2.6 开始无效）  
    X 死掉的进程
33. **问题十八：**
34. 怎么使一个命令在后台运行?  
    **答案：**  
    一般都是使用 & 在命令结尾来让程序自动运行。(命令后可以不追加空格)
35. **问题十九：**
36. 利用 ps 怎么显示所有的进程? 怎么利用 ps 查看指定进程的信息？  
    **答案：**  
    ps -ef (system v 输出)
37. ps -aux bsd 格式输出
38. ps -ef | grep pid  
      
    **问题二十：**
39. 哪个命令专门用来查看后台任务?
40. **答案：**
41. job -l
42. **问题二十一：**
43. 把后台任务调到前台执行使用什么命令?把停下的后台任务在后台执行起来用什么命令?  
    **答案：**  
    把后台任务调到前台执行 fg
44. 把停下的后台任务在后台执行起来 bg
45. **问题二十二：**
46. 终止进程用什么命令? 带什么参数?
47. **答案：**
48. kill [-s <信息名称或编号>][程序] 或 kill [-l <信息编号>]
49. kill-9 pid
50. **问题二十三：**
51. 怎么查看系统支持的所有信号？
52. **答案：**
53. kill -l  
      
    **问题二十四：**
54. 搜索文件用什么命令? 格式是怎么样的?
55. **答案：**
56. find <指定目录> <指定条件> <指定动作>
57. whereis 加参数与文件名
58. locate 只加文件名
59. find 直接搜索磁盘，较慢。
60. find / -name "string\*"
61. **问题二十五：**
62. 查看当前谁在使用该主机用什么命令? 查找自己所在的终端信息用什么命令?  
    **答案：**  
    查找自己所在的终端信息：who am i
63. 查看当前谁在使用该主机：who
64. **问题二十六：**
65. 使用什么命令查看用过的命令列表?
66. **答案：**
67. history
68. **问题二十七：**
69. 使用什么命令查看磁盘使用空间？空闲空间呢?
70. 答案：
71. df -hl  
    文件系统 容量 已用 可用 已用% 挂载点  
    Filesystem Size Used Avail Use% Mounted on /dev/hda2 45G 19G 24G 44% /  
    /dev/hda1 494M 19M 450M 4% /boot  
      
    **问题二十八：**
72. 使用什么命令查看网络是否连通?  
    **答案：**  
    netstat  
      
    **问题二十九：**
73. 使用什么命令查看 ip 地址及接口信息？
74. **答案：**
75. ifconfig  
      
    **问题三十：**
76. 查看各类环境变量用什么命令?
77. **答案：**
78. 查看所有 env  
    查看某个，如 home：env $HOME  
      
    **问题三十一：**
79. 通过什么命令指定命令提示符?
80. **答案：**
81. \u：显示当前用户账号
82. \h：显示当前主机名
83. \W：只显示当前路径最后一个目录
84. \w：显示当前绝对路径（当前用户目录会以~代替）
85. $PWD：显示当前全路径
86. \$：显示命令行’$'或者’#'符号
87. \#：下达的第几个命令
88. \d：代表日期，格式为week day month date，例如："MonAug1"
89. \t：显示时间为24小时格式，如：HH：MM：SS
90. \T：显示时间为12小时格式
91. \A：显示时间为24小时格式：HH：MM
92. \v：BASH的版本信息 如export PS1=’[\u@\h\w\#]\$‘
93. **问题三十二：**
94. 查找命令的可执行文件是去哪查找的? 怎么对其进行设置及添加?
95. **答案：**
96. whereis [-bfmsu][-B <目录>...][-M <目录>...][-S <目录>...][文件...]
97. 补充说明：whereis 指令会在特定目录中查找符合条件的文件。这些文件的烈性应属于原始代码，二进制文件，或是帮助文件。
98. -b   只查找二进制文件。
99. -B<目录> 只在设置的目录下查找二进制文件。-f 不显示文件名前的路径名称。  
    -m   只查找说明文件。  
    -M<目录> 只在设置的目录下查找说明文件。-s 只查找原始代码文件。  
    -S<目录> 只在设置的目录下查找原始代码文件。-u 查找不包含指定类型的文件。  
    which 指令会在 PATH 变量指定的路径中，搜索某个系统命令的位置，并且返回第一个搜索结果。  
    -n 指定文件名长度，指定的长度必须大于或等于所有文件中最长的文件名。  
    -p 与-n 参数相同，但此处的包括了文件的路径。-w 指定输出时栏位的宽度。  
    -V   显示版本信息
100. **问题三十三：**
101. 通过什么命令查找执行命令?  
     **答案：**  
     which 只能查可执行文件
102. whereis 只能查二进制文件、说明文档，源文件等
103. **问题三十四：**
104. 怎么对命令进行取别名？  
     **答案：**  
     alias la='ls -a'  
       
     **问题三十五：**
105. du 和 df 的定义，以及区别？  
     **答案：**
106. du 显示目录或文件的大小
107. df 显示每个<文件>所在的文件系统的信息，默认是显示所有文件系统。  
     （文件系统分配其中的一些磁盘块用来记录它自身的一些数据，如 i 节点，磁盘分布图，间接块，超级块等。这些数据对大多数用户级的程序来说是不可见的，通常称为 Meta Data。） du 命令是用户级的程序，它不考虑 Meta Data，而 df 命令则查看文件系统的磁盘分配图并考虑 Meta Data。  
     df 命令获得真正的文件系统数据，而 du 命令只查看文件系统的部分情况。  
       
     **问题三十六：**
108. awk 详解。  
     **答案：**
109. awk '{pattern + action}' {filenames}  
     #cat /etc/passwd |awk -F ':' '{print $1"\t"$7}' //-F 的意思是以':'分隔 root /bin/bash  
     daemon /bin/sh 搜索/etc/passwd 有 root 关键字的所有行
110. #awk -F: '/root/' /etc/passwd root:x:0:0:root:/root:/bin/bash
111. **问题三十七：**
112. 当你需要给命令绑定一个宏或者按键的时候，应该怎么做呢？
113. **答案：**
114. 可以使用bind命令，bind可以很方便地在shell中实现宏或按键的绑定。
115. 在进行按键绑定的时候，我们需要先获取到绑定按键对应的字符序列。
116. 比如获取F12的字符序列获取方法如下：先按下Ctrl+V,然后按下F12 .我们就可以得到F12的字符序列 ^[[24~。
117. 接着使用bind进行绑定。
118. [root@localhost ~]# bind ‘”\e[24~":"date"'
119. 注意：相同的按键在不同的终端或终端模拟器下可能会产生不同的字符序列。
120. 【附】也可以使用showkey -a命令查看按键对应的字符序列。
121. **问题三十八：**
122. 如果一个linux新手想要知道当前系统支持的所有命令的列表，他需要怎么做？
123. **答案：**
124. 使用命令compgen ­-c，可以打印出所有支持的命令列表。
125. [root@localhost ~]$ compgen -c
126. l.
127. ll
128. ls
129. which
130. if
131. then
132. else
133. elif
134. fi
135. case
136. esac
137. for
138. select
139. while
140. until
141. do
142. done
143. …
144. **问题三十九：**
145. 如果你的助手想要打印出当前的目录栈，你会建议他怎么做？
146. **答案：**
147. 使用Linux 命令dirs可以将当前的目录栈打印出来。
148. [root@localhost ~]# dirs
149. /usr/share/X11
150. 【附】：目录栈通过pushd popd 来操作。
151. **问题四十：**
152. 你的系统目前有许多正在运行的任务，在不重启机器的条件下，有什么方法可以把所有正在运行的进程移除呢？
153. **答案：**
154. 使用linux命令 ’disown -r ’可以将所有正在运行的进程移除。
155. **问题四十一：**
156. bash shell 中的hash 命令有什么作用？
157. **答案：**
158. linux命令’hash’管理着一个内置的哈希表，记录了已执行过的命令的完整路径, 用该命令可以打印出你所使用过的命令以及执行的次数。
159. [root@localhost ~]# hash
160. hits command
161. 2 /bin/ls
162. 2 /bin/su
163. **问题四十二：**
164. 哪一个bash内置命令能够进行数学运算。
165. **答案：**
166. bash shell 的内置命令let 可以进行整型数的数学运算。
167. #! /bin/bash  
     …  
     …  
     let c=a+b  
     …  
     …
168. **问题四十三：**
169. 怎样一页一页地查看一个大文件的内容呢？
170. **答案：**
171. 通过管道将命令”cat file\_name.txt” 和 ’more’ 连接在一起可以实现这个需要.
172. [root@localhost ~]# cat file\_name.txt | more
173. **问题四十四：**
174. 数据字典属于哪一个用户的？
175. **答案：**
176. 数据字典是属于’SYS’用户的，用户‘SYS’ 和 ’SYSEM’是由系统默认自动创建的
177. **问题四十五：**
178. 怎样查看一个linux命令的概要与用法？假设你在/bin目录中偶然看到一个你从没见过的的命令，怎样才能知道它的作用和用法呢？
179. **答案：**
180. 使用命令whatis 可以先出显示出这个命令的用法简要，比如，你可以使用whatis zcat 去查看‘zcat’的介绍以及使用简要。
181. [root@localhost ~]# whatis zcat
182. zcat [gzip] (1) – compress or expand files
183. **问题四十六：**
184. 使用哪一个命令可以查看自己文件系统的磁盘空间配额呢？
185. **答案：**
186. 使用命令repquota 能够显示出一个文件系统的配额信息
187. 【附】只有root用户才能够查看其它用户的配额。

# Java需要看的书籍

|  |
| --- |
| 《剑指offer》  刷《剑指Offer》的感受  我的刷题感受  因为大型公司笔试面试的需要，我决定开始慢慢刷题，每天累积一点，问了一些学长刷什么样的题目比较好，有些搞过ACM的学长建议leetcode，leetcode我看了一下，题目还是有一定的难度的，后来在牛客网上看到《剑指offer》，就开始在上面刷题了。  牛客网上的《剑指offer》总共66题，我完成了63题，还有些题目比较古怪（能力有限），就不想做了，牛客网对于题目做了一些分类，这个对于训练还是很好的，比如：二叉树、程序的鲁棒性、回朔法等等。我是从前往后刷题，做完了一遍，将不会的或者做的不对的收藏起来，按照分类收藏，最后进行总结，然后看看一些大神的代码，其实这个总结深化的过程对自己理解程序还是有很大帮助的。《剑指offer》还有leetcode希望coder能够不用IDE，直接写代码，这个还没有能力，毕竟没有达到炉火纯青的境界，不能水题，自然没那个自信直接在他的代码提交框里写代码。后期我准备试水leetcode，做一些简单、中档难度的，高档的暂时还无能为力。  ————————————————  版权声明：本文为CSDN博主「李连毛」的原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接及本声明。  原文链接：https://blog.csdn.net/jianloubang5433/article/details/52302087  入门阶段  《Head First Java(中文版)》—因为图文并茂  《Java开发实战经典》 ---实际案例  进阶阶段  《Java编程思想》更适合对于Java比较熟悉的程序员，可以当做查知识点的工具说使用，Java语言上多数问题可以在这里找到答案。  《Effective Java中文版 第2版》涵括了java编程中78条极具实用价值的经验规则，非常有益于程序员提高编程效率、代码质量。  读完这本书，对于对对象持有、泛型、容器、多线程、等的认识都会加深一步。  深入阶段  《大型网站系统与 Java 中间件实践》  《大型网站技术架构：核心原理与案例分析》  《JavaScript编程全解》—适合系统学习  《Spring实战(第4版)》--- Spring入门书和工具书  《深入理解Java虚拟机》－－Java程序员必读经典  《Java并发编程实战》  《颈椎病康复指南》    / |

# \*-\*项目中的问题

## 接口保证幂等性是基本的要求，那么幂等性你们是怎么做的❤?

|  |
| --- |
|  |

## 请介绍一下你做的这个P2P项目？

该项目是一个基于互联网金融的网贷平台，有理财端和借款端，我主要是参与做的理财端，该项目主要包括PC站、M站、APP客户端（Android、iOS），由多个项目系统构成，包括前端业务系统PC端和H5端、数据接口系统、核心系统、结算系统、支付系统、定时任务系统、营销活动系统，红包系统，合同签章系统、实名认证接口系统、轮播图系统等。  
整个项目采用分布式开发模式，Tomcat集群部署，采用Nginx实现负载均衡和动静分离，MySQL主从复制模式，Redis集群模式，ActiveMQ的异步处理、Zookeeper做为注册中心，Spring session共享，Redis集群缓存热点数据提升系统吞吐量，整个项目采用的技术架构主要是：Spring mvc + Spring + Dubbo +MyBatis, 采用Dubbo实现项目间的RPC调用，采用分布式文件系统FastDFS存储投资借款合同。  
我在这个平台中参与过前端业务系统，Dubbo数据接口系统，支付接口系统、合同签章系统的开发，对这几个系统比较熟悉，通过这个项目，遇到过一些问题，也学到很多东西。比如最早的时候，对Dubbo开发框架不是很了解，通过该项目，现在比较得心应手。

# 附录

码云地址:<https://gitee.com/gongwangwang>

Github: <https://github.com/gongwangwang>

博客园：<https://www.cnblogs.com/gongwangwang/>

CSDN: <https://me.csdn.net/weixin_45437174>