# Java常见面试题总结

## 一、Java基础

### 1、String类为什么是final的

答：主要是为了“效率”和“安全性”的缘故。若String允许被继承,由于它的高度被使用率,可能会降低程序的性能。（final的类不能被继承）

### 2、HashMap的源码，实现原理，底层结构

答：

**容量(Capacity)：**Capacity就是bucket的大小。

**负载因子(Load factor)：**Load factor就是bucket填满程度的最大比例，一般为0.75。

**resize：**当bucket中的entries的数目大于capacity\*load factor时就需要调整bucket的大小为当前的2倍,重新计算index，把节点再放到新的bucket中。

**条件竞争(race condition)：**新调整HashMap大小的时候，确实存在条件竞争，因为如果两个线程都发现HashMap需要重新调整大小了，它们会同时试着调整大小。在调整大小的过程中，存储在链表中的元素的次序会反过来，因为移动到新的bucket位置的时候，HashMap并不会将元素放在链表的尾部，而是放在头部，这是为了避免尾部遍历(tail traversing)。如果条件竞争发生了，那么就死循环了。

**碰撞：**Java 8之前的实现中是用链表解决冲突的，在产生碰撞的情况下，进行get时，两步的时间复杂度是O(1)+O(n)，Java 8中，利用红黑树替换链表，复杂度就变成了O(1)+O(logn)。

**put函数**大致的思路为：

1.对key的hashCode()做hash，然后再计算index

2.如果没碰撞直接放到bucket里

3.如果碰撞了，以链表的形式存在buckets后

4.如果碰撞导致链表过长(大于等于TREEIFY\_THRESHOLD)，就把链表转换成红黑树

5.如果节点已经存在就替换old value(保证key的唯一性)

6.如果bucket满了(超过load factor\*current capacity)，就要resize

**get函数**的实现：

1.bucket里的第一个节点，直接获取

2.如果有冲突，则通过key.equals(k)去查找对应的entry  
3.若为树，则在树中通过key.equals(k)查找，O(logn)  
4.若为链表，则在链表中通过key.equals(k)查找，O(n)

其他：****CocurrentHashMap、Hashtable****

答案来源：<http://www.importnew.com/18633.html>

### 3、说说你知道的几个Java集合类：list、set、queue、map实现类

### 4、描述一下ArrayList和LinkedList各自实现和区别

### 5、Java中的队列都有哪些，有什么区别。

### 6、反射中，Class.forName和classloader的区别

### 7、Java7、Java8的新特性(baidu问的,好BT)

### 8、Java数组和链表两种结构的操作效率，在哪些情况下(从开头开始，从结尾开始，从中间开始)，哪些操作(插入，查找，删除)的效率高

### 9、Java内存泄露的问题调查定位：jmap，jstack的使用等等

### 10、string、stringbuilder、stringbuffer区别

### 11、hashtable和hashmap的区别

### 13、异常的结构，运行时异常和非运行时异常，各举个例子

### 14、String a= “abc” String b = "abc" String c = new String("abc") String d = "ab" + "c" .他们之间用 == 比较的结果

### 15、String 类的常用方法

### 16、Java 的引用类型有哪几种

### 17、抽象类和接口的区别

### 18、java的基础类型和字节大小。

### 19、Hashtable,HashMap,ConcurrentHashMap 底层实现原理与线程安全问题（建议熟悉 jdk 源码，才能从容应答）

### 20、如果不让你用Java Jdk提供的工具，你自己实现一个Map，你怎么做。说了好久，说了HashMap源代码，如果我做，就会借鉴HashMap的原理，说了一通HashMap实现

### 21、 Hash冲突怎么办？哪些解决散列冲突的方法？

### 22、HashMap冲突很厉害，最差性能，你会怎么解决?从O（n）提升到log（n）咯，用二叉排序树的思路说了一通

### 23、rehash

### 24、hashCode() 与 equals() 生成算法、方法怎么重写

## 二、Java IO

### 1、讲讲IO里面的常见类，字节流、字符流、接口、实现类、方法阻塞。

### 2、讲讲NIO。

### 3、String 编码UTF-8 和GBK的区别?

### 4、什么时候使用字节流、什么时候使用字符流?

### 5、递归读取文件夹下的文件，代码怎么实现

## 三、Java Web

### 1、session和cookie的区别和联系，session的生命周期，多个服务部署时session管理。

### 2、servlet的一些相关问题

### 3、webservice相关问题

### 4、jdbc连接，forname方式的步骤，怎么声明使用一个事务。举例并具体代码

### 5、无框架下配置web.xml的主要配置内容

### 6、jsp和servlet的区别

## 四、JVM

### 1、Java的内存模型以及GC算法

### 2、jvm性能调优都做了什么

### 3、介绍JVM中7个区域，然后把每个区域可能造成内存的溢出的情况说明

### 4、介绍GC 和GC Root不正常引用。

### 5、自己从classload 加载方式，加载机制说开去，从程序运行时数据区，讲到内存分配，讲到String常量池，讲到JVM垃圾回收机制，算法，hotspot。反正就是各种扩展

### 6、jvm 如何分配直接内存， new 对象如何不分配在堆而是栈上，常量池解析

### 7、数组多大放在 JVM 老年代（不只是设置 PretenureSizeThreshold ，问通常多大，没做过一问便知）

### 8、老年代中数组的访问方式

### 9、GC 算法，永久代对象如何 GC ， GC 有环怎么处理

### 10、谁会被 GC ，什么时候 GC

### 11、如果想不被 GC 怎么办

### 12、如果想在 GC 中生存 1 次怎么办

## 五、开源框架

### 1、hibernate和ibatis的区别

### 2、讲讲mybatis的连接池。

### 3、spring框架中需要引用哪些jar包，以及这些jar包的用途

### 4. springMVC的原理

### 5、springMVC注解的意思

### 6、spring中beanFactory和ApplicationContext的联系和区别

### 7、spring注入的几种方式（循环注入）

### 8、spring如何实现事物管理的

### 9、springIOC

### 10、spring AOP的原理

### 11、hibernate中的1级和2级缓存的使用方式以及区别原理（Lazy-Load的理解）

### 12、Hibernate的原理体系架构，五大核心接口，Hibernate对象的三种状态转换，事务管理。

## 六、多线程

### **1、Java创建线程之后，直接调用start()方法和run()的区别**

### **2、常用的线程池模式以及不同线程池的使用场景**

### **3、newFixedThreadPool此种线程池如果线程数达到最大值后会怎么办，底层原理。**

### **4、多线程之间通信的同步问题，synchronized锁的是对象，衍伸出和synchronized相关很多的具体问题，例如同一个类不同方法都有synchronized锁，一个对象是否可以同时访问。或者一个类的static构造方法加上synchronized之后的锁的影响。**

### **5、了解可重入锁的含义，以及ReentrantLock 和synchronized的区别**

### **6、同步的数据结构，例如concurrentHashMap的源码理解以及内部实现原理，为什么他是同步的且效率高**

### **7、atomicinteger和volatile等线程安全操作的关键字的理解和使用**

### **8、线程间通信，wait和notify**

### **9、定时线程的使用**

### **10、场景：在一个主线程中，要求有大量(很多很多)子线程执行完之后，主线程才执行完成。多种方式，考虑效率。**

### **11、进程和线程的区别**

### **12、什么叫线程安全？举例说明**

### **13、线程的几种状态**

### **14、并发、同步的接口或方法**

### **15、HashMap 是否线程安全，为何不安全。 ConcurrentHashMap，线程安全，为何安全。底层实现是怎么样的。**

### **16、J.U.C下的常见类的使用。 ThreadPool的深入考察； BlockingQueue的使用。（take，poll的区别，put，offer的区别）；原子类的实现。**

### **17、简单介绍下多线程的情况，从建立一个线程开始。然后怎么控制同步过程，多线程常用的方法和结构**

### **18、volatile的理解**

### **19、实现多线程有几种方式，多线程同步怎么做，说说几个线程里常用的方法**

## 七、网络通信

### 1、http是无状态通信，http的请求方式有哪些，可以自己定义新的请求方式么。

### 2、socket通信，以及长连接，分包，连接异常断开的处理。

### 3、socket通信模型的使用，AIO和NIO。

### 4、socket框架netty的使用，以及NIO的实现原理，为什么是异步非阻塞。

### 5、同步和异步，阻塞和非阻塞。

### 6、OSI七层模型，包括TCP,IP的一些基本知识

### 7、http中，get post的区别

### 8、说说http,tcp,udp之间关系和区别。

### 9、说说浏览器访问www.taobao.com，经历了怎样的过程。

### 10、HTTP协议、 HTTPS协议，SSL协议及完整交互过程；

### 11、tcp的拥塞，快回传，ip的报文丢弃

### 12、https处理的一个过程，对称加密和非对称加密

### 13、head各个特点和区别

### 14、说说浏览器访问www.taobao.com，经历了怎样的过程。

## 八、数据库MySql

### 1、MySql的存储引擎的不同

### 2、单个索引、联合索引、主键索引

### 3、Mysql怎么分表，以及分表后如果想按条件分页查询怎么办(如果不是按分表字段来查询的话，几乎效率低下，无解)

### 4、分表之后想让一个id多个表是自增的，效率实现

### 5、MySql的主从实时备份同步的配置，以及原理(从库读主库的binlog)，读写分离

### 6、写SQL语句。。。

### 7、索引的数据结构，B+树

### 8、事务的四个特性，以及各自的特点（原子、隔离）等等，项目怎么解决这些问题

### 9、数据库的锁：行锁，表锁；乐观锁，悲观锁

### 10、数据库事务的几种粒度；

### 11、关系型和非关系型数据库区别

## 九、设计模式

### 1、单例模式：饱汉、饿汉。以及饿汉中的延迟加载,双重检查

### 2、工厂模式、装饰者模式、观察者模式。

### 3、工厂方法模式的优点（低耦合、高内聚，开放封闭原则）

## 十、算法

### 1、使用随机算法产生一个数，要求把1-1000W之间这些数全部生成。（考察高效率，解决产生冲突的问题）

### 2、两个有序数组的合并排序

### 3、一个数组的倒序

### 4、计算一个正整数的正平方根

### 5、说白了就是常见的那些查找、排序算法以及各自的时间复杂度

### 6、二叉树的遍历算法

### 7、DFS,BFS算法

### 9、比较重要的数据结构，如链表，队列，栈的基本理解及大致实现。

### 10、排序算法与时空复杂度（快排为什么不稳定，为什么你的项目还在用）

### 11、逆波兰计算器

### 12、Hoffman 编码

### 13、查找树与红黑树

## 十一、并发与性能调优

### 1、有个每秒钟5k个请求，查询手机号所属地的笔试题(记得不完整，没列出)，如何设计算法?请求再多，比如5w，如何设计整个系统?

### 2、高并发情况下，我们系统是如何支撑大量的请求的

### 3、集群如何同步会话状态

### 4、负载均衡的原理

### 5、如果有一个特别大的访问量，到数据库上，怎么做优化（DB设计，DBIO，SQL优化，Java优化）

### 6、如果出现大面积并发，在不增加服务器的基础上，如何解决服务器响应不及时问题“。

### 7、假如你的项目出现性能瓶颈了，你觉得可能会是哪些方面，怎么解决问题。

### 8、如何查找 造成 性能瓶颈出现的位置，是哪个位置照成性能瓶颈。

### 你的项目中使用过缓存机制吗？有没用用户非本地缓存

## 十二、其他

### 1、常用的linux下的命令