7 月 12 日
java 面试中的网络知识考点：
1-4：。。。
会话层：建立不同机器上的用户之间建立和管理会话；
表示层：解决不同系统之间发包通信语法方面的问题，将数据根据网络所表示的形式进行格式化；
应用层：规定消息头的长度，以便用户可以将消息划分出来，

TCP 建立连接的流程图如下：第一次握手，建立连接时，客户端发送 syn 包到服务器，并进入 syn_send 状态，等待服务器的确认；第二次握手：服务器收到 syn 包，必须确认客户的 syn，同时自己也发送一个 syn 包，此时服务器进入 syn-recv 状态；第三次握手：客户端收到服务器的包，向服务器发送确认包，此包发送完毕，客户端和服务器进入 established 状态，完成三次握手。
为什么需要三次握手才能建立起来连接：为了初始化 sequence number 的初始值；
首次握手的隐患——syn 超时：问题起因--server 收到 client 的 syn，回复 syn-ack 的时候未收到 ack 确认；server 不断充实直至超时，Linux 默认等待 63 秒才能断开连接；
针对 syn flood 的防护措施：syn 队列满后，通过 tcp-syncoolies 参数回发 syn cookie；若为正常连接则 client 会回发 syn coolie，直接建立连接；
保活机制：向对方发送保活探测报文，如果为收到相应则继续发送，尝试次数达到保活探测次数仍未收到响应则中断连接；
tcp 断开连接时候的四次挥手：第一次挥手 client 发送一个 fin，用来断臂 client 到 server 的数据传送，client 进入 fin_wait1 状态；第二次挥手：server 收到 fin 后，发送一个 ack 给 client 确认序号为收到序号+1，server 进入 close_wait 状态；第三次挥手：server 发送一个 fin 用来关闭 server 到 client 的数据传送，server 进入 last_ack 状态；第四次挥手：client 收到 fin 后 client 进入 time——wait 状态，接着发送一个 ack 给 server，确认序号为收到序号+1，server 进入 closed 状态完成四次挥手。
为什么会有 time——wait 状态：确保有足够的时间让对方收到 ack 包，如果被动端没有收到 ack 就触发触发关闭端重新发送 ack 包，2.避免新旧链接混淆也就是防止与后面的包混在一起
服务器出现大量的 close——wait 状态的原因：主要表现为客户端一直给服务器发送请求但是服务器一直没有接收到请求 ，原因是对方关闭 socket 连接，我方忙于读或写，没有即使关闭连接，也就是说服务器一直处在 close——wait 这个阶段，解决办法是 1.检查代码，特别是释放资源的代码；检查配置，特别是处理请求的现成配置；
超文本传输协议的主要特点：支持客户服务器模式，简单快速，灵活，无连接，无状态；
http 请求或者相应的步骤：客户端连接到尾部服务器，发送 http 请求；服务器接收请求并返回 http 相应；释放连接 tcp 连接，客户端浏览器解析 HTML 内容；
在浏览器地址栏键入 URL，按下回车之后经历的流程：浏览器 DNS 解析（注意查询缓存并解析 URL 中对应的地址，从近到远依次是：浏览器缓存，系统缓存，路由器缓存，IPS 缓存，域名服务器缓存和顶级域名服务器缓存），建立 tcp 连接，发送 http 请求，服务器处理请求并返回 http 报文；浏览器解析渲染页面，连接结束；
http 的状态码的五种肯能的取值：1xx 指示信息--表示请求已接收，继续处理；2xx 成功则表示请求一杯成功接收、理解、接收；3xx 重定向--要完成请求必须进行更进一步的操作；4xx 客户端错误--请求有语法报错或者是请求无法实现；5xx 服务器端错误--服务器未能实现合法的请求；

http 中的常见的状态码：200：表示 OK 可以正常返回信息；400 bad request：表示客户端有语法错误，不能被服务器所理解；401 UNauthenticate：请求未经授权，这个状态码必须和 www。authenticate 爆头域一起使用；403 forbidden：服务器收到请求，但是拒绝提供服务；404 not found：请求资源不存在，输入了错误的 URL；500 intemal servererror：服务器发生不可预期的错误；503 serverunavailable：表示服务器当前不能处理客户端的请求，一段时间之后可能恢复正常

get 请求和 post 请求的区别：从三层来进行解答：

http 报文方面：get 请求信息放在 URL（因此长度是有限制的，虽然 URL 本身是没有限制的，但是浏览器会对信息做出相应的限制），post 放在报文体中（post 的信息是没有限制的）；

数据库层面：get 符合幂等性（多次操作和一次操作的结果是一样的）和安全性（没有改变数据库中的元素），post 不符合（post 会往数据库中放入数据，并且 post 是放在栈上面的即每一次请求都需要在站上放置一份资源）

其他层面：get 请求可以被缓存，get 请求可以放在浏览器的浏览记录当中，被存储；post 不能

cookie 和 session 的简介：是有服务器发给客户端的特殊的信息，以文本的形式存放在客户端，客户端再次请求的时候，会把 cookie 回发；服务器收到后，会解析 cookie 生成与客户端相对应的内容：cookie 的设置以及发送过程：1.客户端向服务器发送一个 http 请求；2.服务器回复给客户端一个相应和设定 cookie 的，客户端向服务器端发送了一个包含 cooKies 的请求到服务器端；4.服务器向客户端发出一个 http 响应；

session：服务器端的机制，在服务器上保存信息，解析客户端请求并操作 sessionID 按需保存状态；session 的实现方式有两种，第一种是通过 cookie 来实现，在 cookie 的第二步和第三步上加上 Jsession-xxx 的请求信息和应答信息；第二种是通过 URL 的回写来实现，在客户端向服务器发送的所有 URL 中都携带 jsession 的参数，这样客户端点击任何一个链接都可以把 jsession 的参数带回服务器

cookie 和 session 的区别：1.cookie 数据存放在客户的浏览器上，session 的数据存放在服务器上，session 相对于 cookie 更加安全；如果考虑减轻服务器的负担，应当使用 cookie；

http 和 HTTPS：http 包含 http，tcp 和 ip；HTTPS 包含 http ssl 或者是 tls 和 tcp ip；

ssl（安全套接层）为网络通信提供安全及数据完整性的一种安全协议；2、是操作系统对外的 api，ssl3.0 之后更名为 tls；采用身份验证和数据加密来保证网络通信的安全和数据的完整性；

https 数据传输流程：1.浏览器将支持的一套加密算法信息发送给服务器，服务器选择一套浏览器支持的加密算法，以证书的形式回发给浏览器；浏览器收到验证整数的合法性，并结合证书公钥加密信息发送给服务器；服务器使用私钥揭秘信息，验证哈希，加密相应消息回发浏览器；浏览器解密相应消息，并对消息进行验证，之后进行加密交互数据；

http 和 HTTPS 区别：HTTPS 需要到 ca 申请证书，http 不需要，HTTPS 采用密文传输，http 使用的是明文传输；连接方式不同，HTTPS 默认使用 443 端口，http 使用 80 端口；HTTPS=http+加密+认证+完整性保护，较 http 安全；

HTTPS 真的安全吗：浏览器默认填充 http://请求需要进行跳转；

socket 简介：socket 是对 tcp/ip 协议的抽象，是操作系统对外开放的接口；

socket 通信流程：服务端：.....客户端：.....

第三章

第四章

第五章

第六章：jvm 的底层知识

谈谈你对 java 的理解（需要扩充）：1.平台无关性；2.GC；3.语言特性；4.面向对象；5.类库；6.异常处理；

平台无关性是如何实现的：java 源码首先被编译成字节码，再由不同平台的 jvm 进行解析，java 语言在不同的平台上运行时不需要进行重新编译，java 虚拟机在执行字节码的时候把字节码转换成具体平台上的机器指令；

为甚 jvm 不直接将源码解析成机器码去执行：准备工作：每次执行都需要各种检查；兼容性：也可以将别的语言解析成字节码；

jvm 如何加载。class 文件：jvm 主要由 class loader；runtime data Area ；Execution Engine；和 native interface 这几个部分组成；主要通过 class loader 将 class 文件加载到内存里，然后通过 excution engine 对命令进行解析字节码，并提交到操作系统中去执行；

（需要扩充）谈谈类加载器的双亲委派机制：首先自底向上检查类是否已经加载；然后自顶向下尝试类加载

java 的内存模型

地址空间划分为内核空间和用户空间；java 的内存模型是指 jvm 中的 runtime area 区域；jvm 的内存模型可以根据线程私有和线程共享分为两个部分：线程私有包含：程序计数器（字节码指令 nooom），虚拟机栈（java 方法 sof&oom）和本地方法栈（native 方法 SOF&oom）；所有线程共享包括元空间 Metaspace（类加载信息 oom）、常量池（字面量和符号引用量 OOM）和堆空间（数组和类对象 oom）

元空间（metespace）和永久代 permGen 的区别：1.元空间使用的是本地内存，而永久代使用的是 jvm 内存，

元空间（metespace）的优势：字符串常量池存在于永久代中，容易出现性能问题和内存溢出；2.类方法的信息大小难以确定，给永久代的大小指定带来困难；3.永久代会为 GC 带来不必要的复杂性；4.方便 Hotspot 与其他 jvm 如 jrokit 的集成；

java 堆（heap）：唯一目的是存放对象实例，java 堆可以处于磁盘空间不连续的内存中；java 堆是 gc 管理的主要区域；

jvm 三大性能调优参数-Xms -Xmx -Xss 的含义：-xss 规定了每个线程虚拟机栈（堆栈的大小）；-xms 规定了对的初始值，如果堆的容量不足将会自动扩容至-xmx；-xmx 表示堆能达到的最大值；

java 内存模型中堆和栈的区别：首先需要解释以下 jvm 中三种内存分配策略：静态存储：在编译时确定每个数据目标在运行时的存储空间；栈式存储：数据区要求编译时未知，运行时模块入口前确定；堆式存储：编译时或者是运行时模块入口都无法确定，动态分配；联系：引用对象、数组时，栈里面定义变量保存堆中目标的首地址；区别：1.管理方式上，栈会自动释放，堆需要 GC 进行自动回收；2.空间大小：栈比堆小；碎片相关：站产生的碎片远小于堆；4.分配方式：栈空间有静态分配和动态分配，堆支持动态分配；5.效率：栈的效率比堆高；

请解释以下 jdk6 和 jdk6+的 intern()方法的区别：djk6 当调用 intern 方法时，如果字符串

常量池先前已创建出该字符串对象，则返回池中的该字符串的引用。否则，将此字符串对象添加到字符串常量池中，并且返回字符串对象的引用。当调用 intern 方法时，如果字符串常量池先前已创建该字符串对象，则返回翅中的该字符串的引用。否则将此字符串兑现个添加到字符串常量池中，并且返回该字符串对象的引用。jdk6+：当调用 intern 方法时，如果子字符串常量池先前已创建出字符串对象，则返回池中的该字符串的引用，否则，如果该字符串对象已经存在于 java 堆中，则将堆中对此对象的引用添加到字符串常量池中，并且返回该引用，如果堆中不存在，则在翅中创建该字符串并返回其引用。

java 的垃圾回收机制：
方法一 引用计数法
判断对象的引用数量：1.通过判断对象的引用数量来决定对象是否可以被回收；2.每个对象的实例都有一个引用计数器，被引用则+1，完成引用则-1；任何引用计数为 0 的对象实例可以被当做垃圾收集；
优点：执行效率高，程序执行受影响较小；缺点：无法检测出循环引用的情况，导致内存泄漏；
方法二：可达性分析算法：通过判断对象的因用力按是否科大来决定对象是否可以被回收
可以作为 GCroot 的对象：虚拟机栈中引用的对象（栈帧中的本地变量表）；方法区中的常量引用的对象；本地方法栈中 jni 的引用对象；活跃线程的引用对象；
object 的 finalize()方法的作用是否与 c++的析构函数作用相同：1.析构函数调用时确定的，finalize()方法调用函数是不确定的，当垃圾回收器要宣告一个对象死亡时至少要标记两次，如果对象覆盖 finalize 方法并且未被引用过就会被放入 F-queue 队列；并在之后通过虚拟机建立的低优先级的线程来调用 finalize()方法；方法执行随时可能会被终止；基于对象最后一次重生的机会；
java 中的强引用，弱引用，软引用和虚引用有什么作用：强引用 object obj=new Object() 抛出 outofmemoryError 终止程序也不会回收具有强引用的对象，通过将对象设置为 null 来弱化引用，使其被回收；软引用：对象处在有用但非必须的状态，只有内存空间不足时，GC 会回收该引用的对象的内存，可以用来实现告诉缓存，例如泛型；弱引用，非必须的对象，比软引用更弱一些，GC 时可能会被回收，被回收的概率也不大，因为 GC 线程优先级比较低，适用于引用偶尔被使用且不影响垃圾收集的对象；虚引用：不会决定对象的生命周期，任何时候都可能会被垃圾收集器回收的活动，起哨兵作用，必须和引用队列 referenceQueue 联合使用(核心)；
引用队列：无实际存储结构，存储逻辑依赖于内部节点之间的关系来表达，存储关联的且被 GC 的软引用，弱引用和虚引用；