https://www.nowcoder.com/discuss/286201?type=all&order=time&pos=&page=1&channel=666&source\_id=search\_all

1. 数据库的事务理解

数据库事务是指把一系列数据库操作封装在一起，这些操作要么全部成功，要么全部失败。

事务的特性：ACID（原子、一致、隔离、持久）

1. A：原子性，整个事务中的操作只有全部成功执行、全部不执行；
2. C：一致性，执行事务前后，数据的一致性不会改变，

eg：进行汇款操作，A账户汇款该B账户，无论汇款成功与否，A + B账户的总额都是不变的。

1. I：独立性，两个事务不会相互干扰，这边就有：脏读、不可重复读、幻读等情况；
2. D：持久性，事务执行成功后，数据会保存在数据库中，不会倍回滚。
3. 线程池
4. 7个参数：

核心线程、最大线程、工作队列、拒绝策略、空线程存活时间、时间单位、线程工厂；

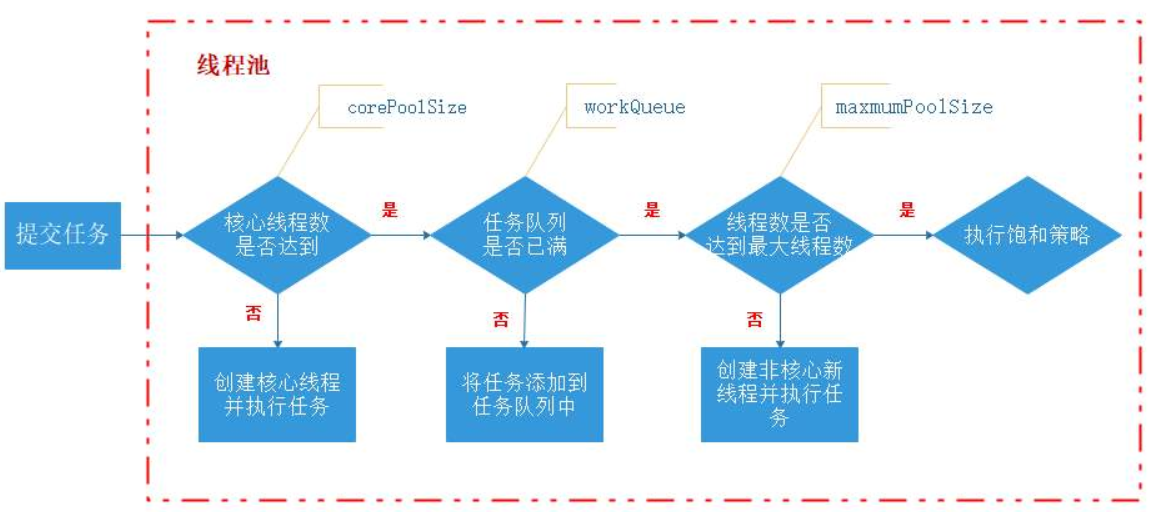
1. 4个线程池类型：

固定核心线程数、单个线程、可根据需要创建线程、定时和延时执行

1. 4个拒绝策略：

直接绝任务、丢弃任务并抛出异常、丢弃任务且不做处理、丢弃最早进入工作队列的任务然后添加本次拒绝的任务；

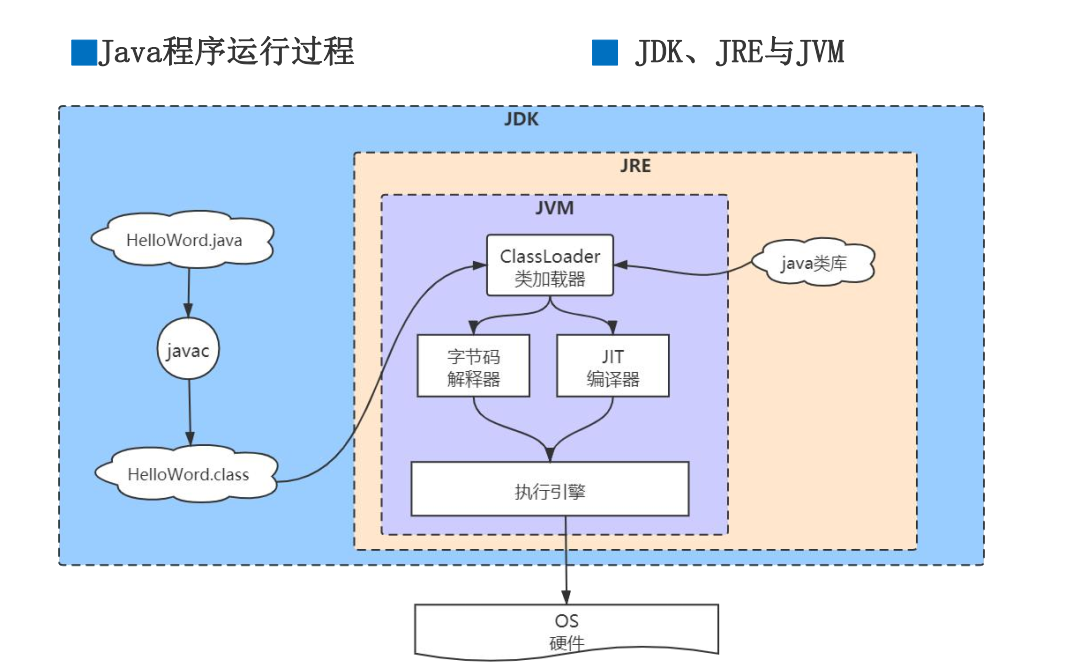
线程池的运行原理：



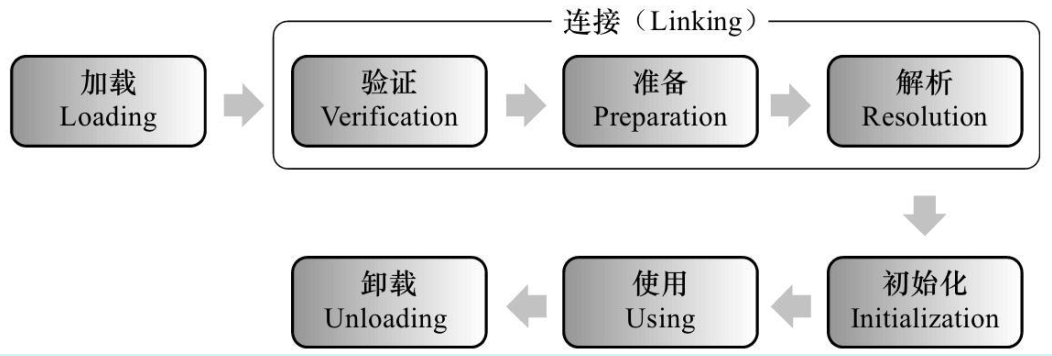
1. jdk和jre

jdk，java development kit，java开发工具包，包含了：jre，还多出了其他的工具包，eg：javac（编译成.class文件）、javap（显示字节码文件）

jre：java runtime enviromentt，java运行环境，包含了JVM，以及其他的类库：java.lang、java.util等等。



类加载过程：



1. javahome和classpath、path是啥：
2. java\_home：

指的是jdk安装的路径，在设置path时，可使用%JAVA\_HOME%\bin来简化输入，通过修改JAVA\_HOME值来修改jdk版本；

1. class\_path：

在运行java程序时，jre在class\_path路径下搜索需要的.class文件；

1. path：

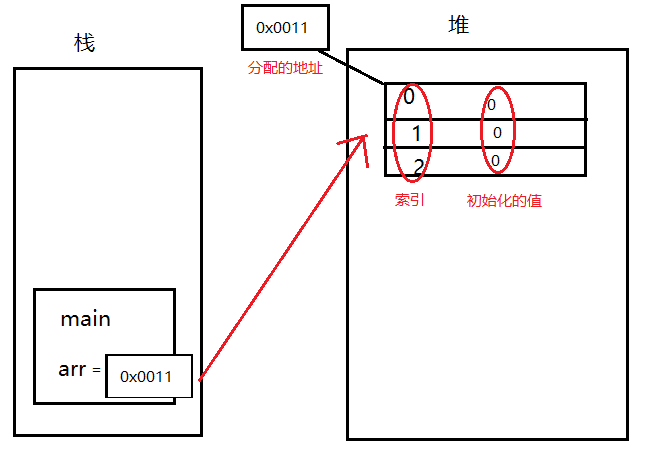
环境变量，用来配置cmd中的命令，eg：安装java后，cmd中找不到java命令，在PATH中配置java的安装目录中的bin路径，系统就可以找到根据这个路径找到java程序运行。

1. 怎么在linux部署java环境
2. 线程和进程
3. java怎么反编译
4. java栈和堆的区别
5. 栈：

是一片内存区域，主要存放局部变量，eg：for循环中的局部变量，方法进入栈中，然后定义局部变量，出栈后，局部变量的空间会被释放；

1. 堆：

存放的是对象，所有new的对象都存放堆中，GC收集的就是堆内存。



1. 为什么要手动gc
2. 什么是jar包
3. 怎么处理依赖
4. IOC具体是怎么实现的，这种方式有什么优点
5. 实现：

springboot中使用DI方式实现IOC：主要有构造器注入、set方法注入、自动注入（byname、bytype）

1. 优点：

方便管理对象；

1. AOP：

概念：织入-》切面-》切点-》通知

1. 编译时织入：静态代理，AspectJ框架；
2. 运行时织入：动态代理，Spring AOP框架，仿照AspectJ。分为JDK动态代理和CGLIB动态代理
3. Spring 事务：
4. 分布式事务:
5. 分布式锁:
6. Springboot中使用的设计模式：

1. Linux统计某个文件中，某个字符出现的行数：

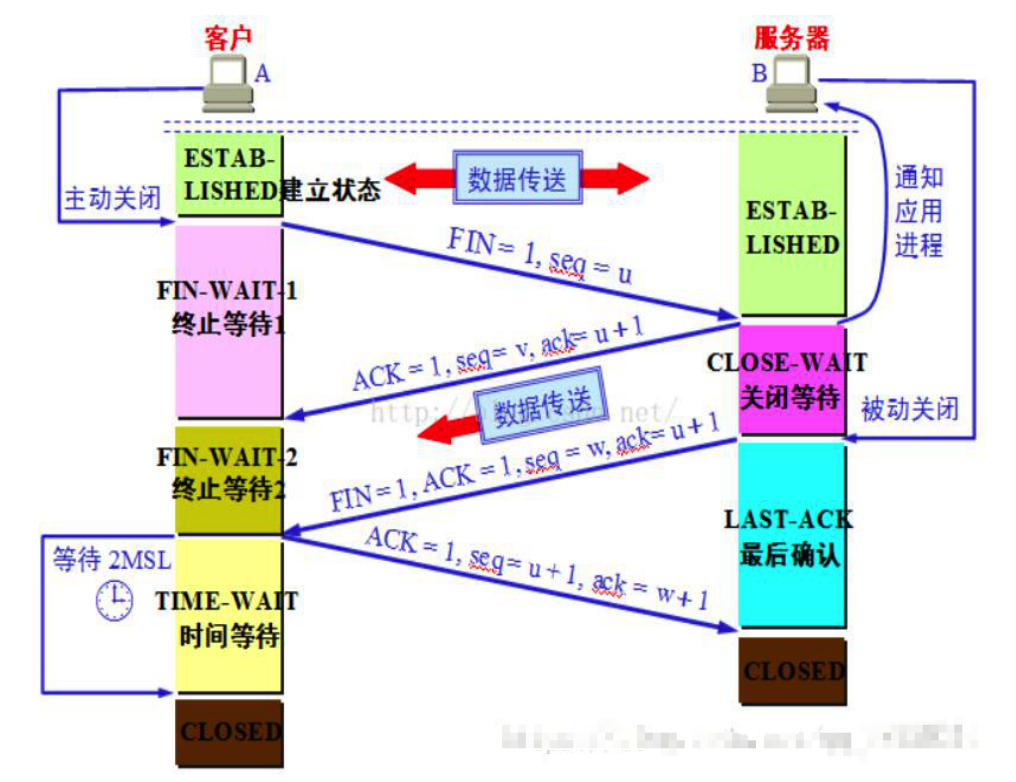
find aaa.txt | xargs cat | grep .\*aa.\*|wc -l  
例子说明：统计含"aa"字符串的总行数

1. http的close\_wait状态

两个基本问题：

1. Q: 我们看到CLOSE\_WAIT出现在什么时候呢？  
   A: 在Sever端收到Client的FIN消息之后。
2. Q: 状态CLOSE\_WAIT在什么时候转换成下一个状态呢？  
   A: 在Server端向Client发送FIN消息之后。

至此似乎明白了为什么会出现CLOSE\_WAIT的状态：如果Server端一直没有向client端发送FIN消息(调用close() API)，那么这个CLOSE\_WAIT会一直存在下去。



1. Tomcat执行流程：Spring响应用户请求的过程

**客户端：**

1、Web客户向Servlet容器（Tomcat）发出Http请求   
**服务端：**  
2、Servlet容器分析客户的请求信息   
3、Servlet容器创建一个HttpRequest对象，将客户请求的信息封装到这个对象中   
4、Servlet容器创建一个HttpResponse对象   
5、Servlet容器调用HttpServlet对象的service方法，把HttpRequest对象与HttpResponse对象作为参数传给 HttpServlet对象   
6、HttpServlet调用HttpRequest对象的有关方法，获取Http请求信息   
7、HttpServlet调用HttpResponse对象的有关方法，生成响应数据   
8、Servlet容器把HttpServlet的响应结果传给Web客户