看到一篇讲述 Tomcat 调优的文章，写得挺不错的，由于之前写了好几篇关于 Tomcat 的文章，这篇分享给各位读者，做一下补充。

**1 描述**

最近，在补充自己的短板，刚好整理到 Tomcat 调优这块，基本上面试必问，于是就花了点时间去搜集一下 Tomcat 调优都调了些什么，先记录一下调优手段，更多详细的原理和实现以后用到时候再来补充记录，下面就来介绍一下，Tomcat 调优大致分为两大类。

**1.1 Tomcat的自身调优**

采用动静分离节约 Tomcat 的性能调整 Tomcat 的线程池调整 Tomcat 的连接器修改 Tomcat 的运行模式禁用 AJP 连接器

**1.2 JVM的调优**

调优Jvm内存

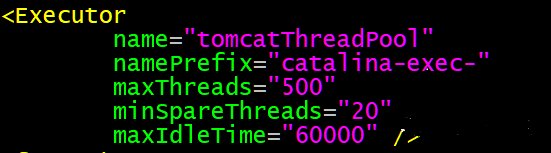
**2 Tomcat自身调优**

**2.1 采用动静分离**

静态资源如果让 Tomcat 处理的话 Tomcat 的性能会被损耗很多，所以我们一般都是采用：Nginx+Tomcat 实现动静分离，让 Tomcat 只负责 jsp 文件的解析工作，Nginx 实现静态资源的访问。

**2.2 调优Tomcat 线程池**

打开tomcat的serve.xml，配置Executor，相关参数说明如下。



name：给执行器（线程池）起一个名字；

namePrefix：指定线程池中的每一个线程的 name 前缀；

maxThreads：线程池中最大的线程数量，假设请求的数量超过了 750，这将不是意味着将 maxThreads 属性值设置为 750，它的最好解决方案是使用「Tomcat集群」。也就是说，如果有 1000 请求，两个 Tomcat 实例设置 maxThreads = 500，而不在单 Tomcat 实例的情况下设置 maxThreads=1000。

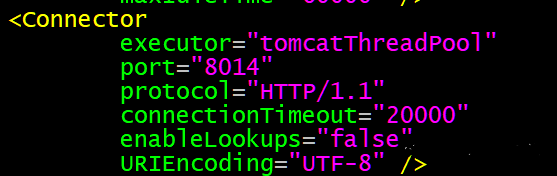
minSpareThreads：线程池中允许空闲的线程数量（多余的线程都杀死）；

maxIdLeTime：一个线程空闲多久算是一个空闲线程；

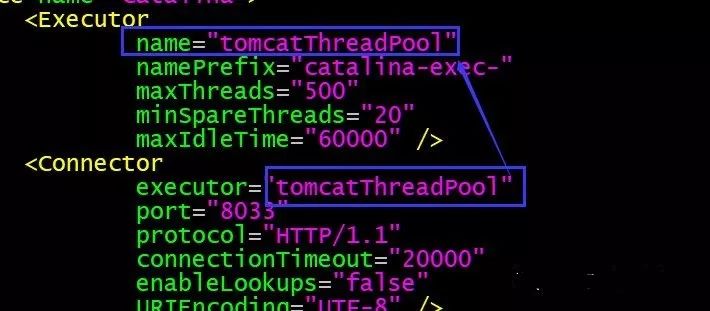
其他的配置其实阅读官方文档是最好的「见参考链接」。

**2.3 调优 Tomcat 的连接器 Connector**

打开 Tomcat 的 serve.xml，配置 Connector，参数说明如下。



executor：指定这个连接器所使用的执行器（线程池）；



enableLookups=false：关闭 DNS 解析，减少性能损耗；

minProcessors：服务器启动时创建的最少线程数；

maxProcessors：最大可以创建的线程数；

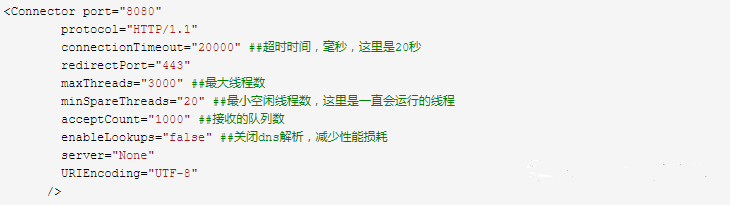
acceptCount=1000：线程池中的线程都被占用，允许放到队列中的请求数；

maxThreads=3000：最大线程数；

minSpareThreads=20：最小空闲线程数，这里是一直会运行的线程；

与压缩有关系的配置：如果已经对代码进行了动静分离，静态页面和图片等数据就不需要 Tomcat 处理了，那么也就不需要配置在 Tomcat 中配置压缩了；

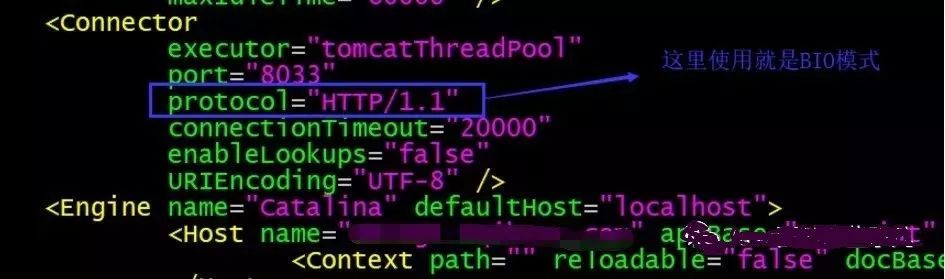
一个完整的配置如下。



**2.4 通过修改 Tomcat 的运行模式**

BIO

Tomcat8 以下版本，默认使用的就是 BIO「阻塞式IO)」模式。



对于每一个请求都要创建一个线程来进行处理，不适合高并发。

NIO

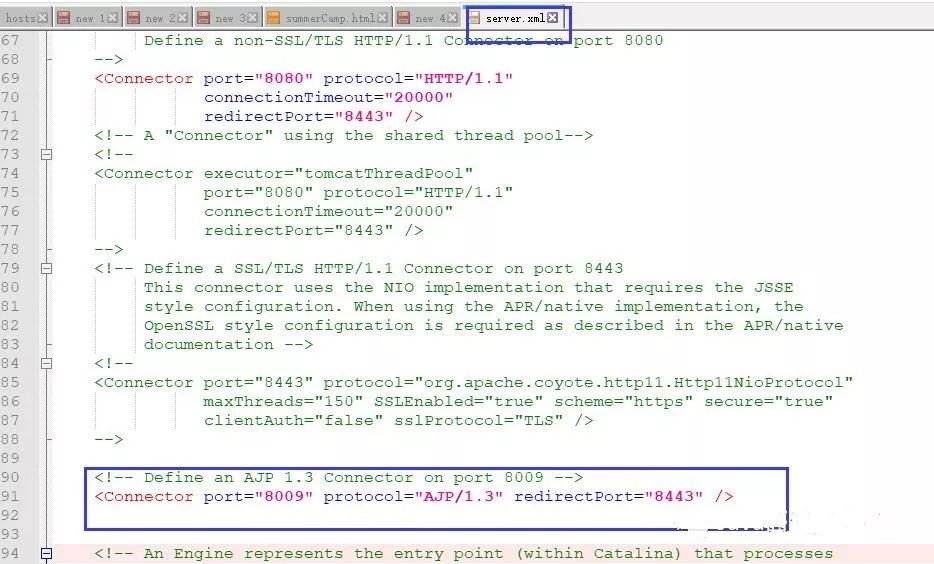
Tomcat8 以上版本，默认使用的就是NIO模式「非阻塞式 IO」。

APR

全称 Apache Portable Runtime，是Tomcat生产环境运行的首选方式，如果操作系统未安装 APR 或者 APR 路径未指到 Tomcat 默认可识别的路径，则 APR 模式无法启动，自动切换启动 NIO 模式。所以必须要安装 APR 和 Native，直接启动就支持 APR，APR是从操作系统级别解决异步 IO 问题，APR 的本质就是使用 JNI 技术调用操作系统底层的 IO 接口，所以需要提前安装所需要的依赖提升 Tomcat 对静态文件的处理性能，当然也可以采用动静分离。

**2.5 禁用 AJP 连接器**

AJP的全称 Apache JServer Protocol，使用 Nginx+Tomca t的架构，所以用不着 AJP 协议，所以把AJP连接器禁用。



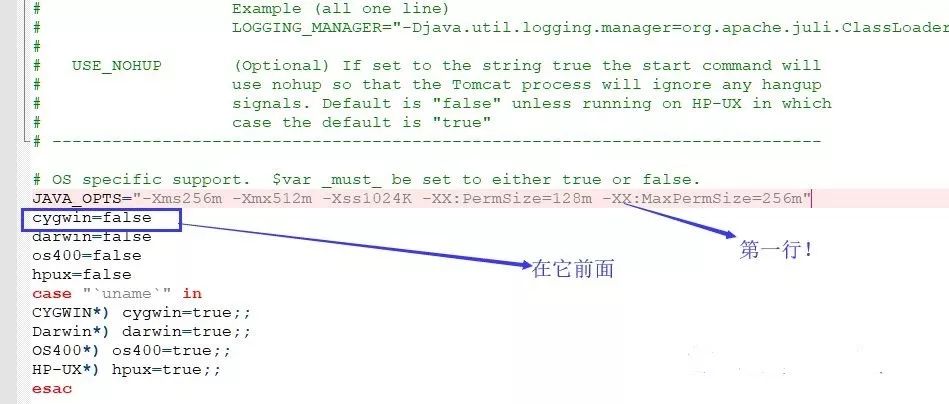
**3 JVM调优**

Tomcat 是运行在 JVM 上的，所以对 JVM 的调优也是非常有必要的。

找到 catalina.sh；



添加；



参数设置；

JAVA\_OPTS="-Djava.awt.headless=true -Dfile.encoding=UTF-8-server -Xms1024m -Xmx1024m -XX:NewSize=512m -XX:MaxNewSize=512m -XXermSize=512m -XX:MaxPermSize=512m -XX:+DisableExplicitGC"

调整堆大小的的目的是最小化垃圾收集的时间，以在特定的时间内最大化处理客户的请求。