## 讲课内容

回顾下上节课的作业----发现一个问题，大部分的学员没有注意到一个问题--其实那个启动Tomcat有两种方式，一个是通过Tomcat启动的，一个是通过catalina启动的，大家以后注意一下。

## 性能优化是一个系统工程

这三个指标并不是单独的，而是一起的，他们是一个三角关系，各个指标都是建立在其他的指标基础上的。他们相互影响，相互关联。

理想是丰满的，现实是骨感的，优化中讲究积少成多，每个方面一点点的优化，多起来的话对全局就有影响了。性能优化其实也是这样的一个主题。

## **Tomcat性能优化中使用的工具**

meter这个工具是纯java写的，它也是一个分布式的工具，可以做单机测试，也可以做分布式的多机的压力测试。具体怎么做分布式压力测试可以百度到，我们今天只用这个做单机测试。他还可以和一些录制脚本进行无缝结合，比如badboy这个工具可以录制脚本

King老师喜欢跟大家分析深入的内容。那我们来探究下它是怎么启动。知其然，必须知其所然。

## **Tomcat中server.xml优化**

老师在家附近的一个很大的理发店办了张会员卡，那个店子生意太好，每次去剪头发都需要排队，这个剪头发的过程本身就是一个长时间的过程，是一个长连接。如果是采用BIO的方式，一个理发师只能同时服务一个人，如果采用NIO的话，可以把过程进行拆解，把剪发分为洗发、剪发、吹法。现在理发店都是采用雇佣很多年轻的小伙子，不管人多少，洗发这个过程可以不阻塞，洗完可能需要等理发师剪发，理发师剪完发后再转给之前洗发的小伙子给你吹干。前面老师这个剪头发的流程就是一个典型的NIO。最大压榨了理发师，把理发师的时间片段都利用了起来。这个就是Tomcat 8.5已经9的版本后把bio给废掉了的原因。

线程池的好处在于统一管理，使用线程池的话如果发现线程空闲时间达到比如（60秒的时间），会自动回收。如果不使用线程池的话就把线程池丢给的gc去回收。（jvm的回收机制），这个就很难控制了，所以使用线程池最大的好处是保证系统的稳定性，很多时候我们做项目，稳定性是排在第一位的。

## **Tomcat中web.xml优化**

JSP中其实是jsp -> java -> class的一个过程。

Tomcat在运行过程中会把你访问的这些jsp做这些操作，其实我们可以预先编译下，节约性能。

## **SpringBoot中Tomcat的优化**

springboot中用了TomcatcontextCustomizer 和TomcatConnectorCustomizer来实现Tomcat,可以看看这个方面的源码（按照第三节课 嵌入式Tomcat的方式去分析和看就行了）