**1**、什么是**Nginx**

1. Nginx是一个web服务器和反向代理服务器，及电子邮件代理服务器，用于HTTP、HTTPS、SMTP、POP3和IMAP协议。

2. 基于REST架构风格，以统一资源描述符URI或者统一资源定位符URL作为沟通依据

**2**、为什么要用**Nginx**

1. 跨平台、配置简单，非阻塞、高并发连接；

2. Nginx具有占用内存少，稳定性高等优势；

3. 内置的健康检查功能, 稳定性高：宕机的概率非常小

4. 节省宽带：可以添加浏览器本地缓存

5. 使用异步逻辑：（从而削减了进程上下文切换的开销，因此并发服务能力更强。）

6. Nginx整体采用模块化设计，有丰富的模块库和第三方模块库，配置灵活。

7. 可以有多台nginx服务器 使用dns做负载均衡,事件驱动

8. Nginx使用epoll事件模型，在Linux系统上运行效率非常高。

**3**和**4**的集合、为什么**Nginx**性能这么高

得益于它的事件处理机制：异步非阻塞事件处理机制：运用了epoll模型，提供了一个队列，排队解决

**3. Nginx** 的采用异步逻辑和一客户一线程有什么区别？

异步逻辑：浏览器将请求发送到nginx服务器，它先将用户请求全部接收下来，再一次性发送给后端web服务器，极大减轻了web服务器的压力,一边接收web服务器的返回数据，一边发送给浏览器客户端, 网络依赖性比较低，只要ping通就可以负载均衡。

一客户一线程：服务器持续侦听，每接收到一个连接请求，就创建一个新线程进行处理。

异步逻辑削减了进程上下文切换的开销，因此并发服务能力更强。

**4.** 什么是**epoll**事件模型？

假设进程有10万个TCP连接，且只有几百个连接是有事件需要处理的。那么在每一个时刻进程只需要处理这几百个有事件需要处理的连接即可。

红色感觉不一定要记：

（1、调用epoll\_creat函数建立一个epoll对象（一颗红黑树，一个准备就绪list链表）。

2、调用epoll\_ctl函数把socket放到红黑树上，给内核中断处理程序注册一个回调函数，告诉内核，如果这个句柄的中断到了，就把这个socket放到准备就绪list链表里。

3、调用epoll\_wait到准备就绪list链表中处理socket，并把数据返回给用户。）

Note:不需要把全部的连接处理一遍，只需要去list链表里处理socket。

获取事件时，操作系统不需要遍历所有连接，提高了程序在高并发中只有少量连接活跃情况下CPU的利用率。

**5.** 为什么会需要类似**Nginx**这一类代理服务器呢？

web应用服务器往往达不到理想中的并发（其原因可能是带宽、系统IO等），进而自然想到通过分布式架构提升并发能力。但又因为公网IP的稀缺性，所以只能采用Nginx反向代理来做类似负载均衡的一些事情。

**6.nignx**如何做负载均衡调度算法？

1. weight
2. 轮询(默认，常用)：每个请求按时间顺序逐一分配到不同的后端服务器，如果后端服务器down掉，能自动剔除
3. ip\_hash（常用）：每个请求中客户端的ip的hash进行匹配，访问到后端服务器。
4. fair：智能调整调度算法，动态的根据后端服务器的请求处理到响应的时间进行均衡分配，响应时间短处理效率高的服务器分配到请求的概率高。
5. url\_hash：每个请求中客户端的url的hash进行匹配，访问到后端服务器。

优点是什么：负载均衡主要解决网络拥塞问题，提高服务器响应速度，服务就近提供，达到更好的访问质量，减少后台服务器大并发压力

**7.** 采用**ip\_hash**有什么优点**(**存在的价值。。。**)**？

把登录信息保存到了session中，那么跳转到另外一台服务器的时候就需要重新登录了，所以很多时候我们需要一个客户只访问一个服务器，那么就需要用iphash了，iphash的每个请求按访问ip的hash结果分配，这样每个访客固定访问一个后端服务器，可以解决session的问题。

**7**、使用**“**反向代理服务器**”**的优点是什么**?**

（具体做法和正向代理区别）**+**

（1）保证内网的安全，通常将反向代理作为公网访问地址，Web服务器是内网

（2）反向代理+负载均衡，通过反向代理服务器来优化服务器的负载

**8**、**Nginx**是如何处理一个请求的

1. nginx在启动时，会解析配置文件，得到需要监听的端口与ip地址 （解析文件获ip端口）

2. 在nginx的master进程里面先初始化好这个监控的socket，再进行listen （初始化）

3. fork出多个子进程,  子进程会竞争accept新的连接。

4. 客户端向nginx发起连接。

5. 三次握手，建立连接后

6. 子进程accept成功，创建nginx对连接的封装，根据事件调用相应的事件处理模块

7. nginx或客户端来主动关掉连接

**9.** 动态资源、静态资源分离

动态网站里的动态网页根据一定规则把不变的资源和经常变的资源区分开来。

根据静态资源的特点将其做缓存操作，这就是网站静态化处理的核心思路。

动态资源、静态资源分离简单的概括是：动态文件与静态文件的分离

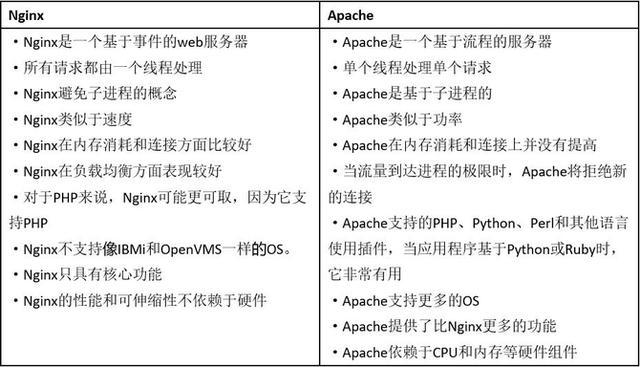
**10.** 为什么要做动、静分离

后台处理忽略静态文件

对资源的响应速度有要求的时候，我们应该使用这种动静分离的策略去解决（减少后台请求次数），动、静分离将网站静态资源

提高用户访问静态代码的速度，降低对后台应用访问，可以将静态资源放到nginx中，动态资源转发到tomcat服务器中

**11**、请列举**Nginx**和**Apache** 之间的不同点。

****

两者最核心的区别在于apache是同步多进程模型，一个连接对应一个进程，而nginx是异步的，多个连接（万级别）可以对应一个进程。

更为通用的方案是，前端nginx抗并发，后端apache集群，配合起来会更好。

本次课程设计采用**nginx**的主要目的和优点？

。。。。

加载zuul网关前，做负载均衡

（1）反向代理，保证内网的安全，通常将反向代理作为公网访问地址，Web服务器是内网

（2）反向代理+负载均衡，通过反向代理服务器来优化服务器的负载