**面试注意事项：**

第一个问题是：螺丝钉，如何去接触更多的知识，对员工的发展，以及职业发展的问题。

第二个问题是：对于研发人员来讲，整天闷头写代码，有些team building，有没有公司的一些club

热情和激情，千万不要批判之前的公司和我本身，要强调自己的有点和信心，可以为公司做什么？

要把jd和简历，记在脑子里，面试的时候，不要一直看简历，注意与面试官交流。

问题：如果，投开发，然后去测试的话，

如果写程序的时候，写不出来吗？

着装：对于银行和财经类的，西装和皮鞋最安全，黑色袜子，落座后西装的上衣扣子都可以解开，皮带，准时抵达（10-15分钟提前），保证体力充足，关于offer的拒绝，要确定一下下一轮面试的时间，东西不要遗漏，还有离开时门要关上吗？有一个寒暄，然后握手去介绍自己

**一 、思科项目介绍**

1. CMTS: (**C**able **M**odem **T**ermination **S**ystem) 实质是一种接入路由器,一端接用户,一端接接入网或边缘路由器.放置在运营商机房,通过光纤出去，然后接入用户的叫HFC光纤同轴混合网。具有DHCP TFTP等功能。

A description...

A description...

其中：Downstream: CMTS to CMs. 广播的形式。所有CM可以收到，通过接收端的MAC或DOCSIS3.0后DS ID辨别目的地。

Cable Modem和普通的Modem一样，是数据信号在模拟信道上交互传输的过程，但是，普通的Modem的传输介质在用户与访问

务器之间是独立的，用户独享传输介质，而Cable Modem的传输介质是HFC网络，将数据信号调制到某个传输带宽与有线电视或者电话信号共享介质，并且Cable Modem结构也较普通的Modem复杂，无须拨号上网不占用电话线，可以随时在线连接。

Cable Modem实现是从87MHz—860Mhz电视频道中分离出一条6Mhz用于下行数据传输采用64QAM,下行数据一般是从：5MHz—65Mhz之间的频谱上进行传送，为了有效抑制上行噪音积累，一般用qpsk，因为它比64QAM更适合噪音环境，但是速率较低。

1. RF-Switch:

Rf-switch 的左边的in 去接linecard，右边的out是给CM。来是实现n + 1冗余功能。7+1冗余，应该是8个linecard，其中在cmts上已经有了8 个linecard（7个常用的一个备份的），然后当有时候运行商要更改某个linecard或者是当其中有一个linecard坏掉的时候，就将其转换到备份的linecard上，然后同时？

Rf-switch:本来作用是实现N+1冗余，也就是说实现当有一个linecard坏掉的时候，就把他其中坏掉的一个

换到protected，（实际上Rf-switch仅仅起一个开关作用，in 和 out 在内部通过开关是连接在一起的）之前是 rfswitch 连接各个linecard，然后output出来后，一路ds 和一路us 混频出去就相当于一个fiber note(FN),然后FN，再经过HFC到modem，在实际应用中，整套设备是放在运营商的机房中，然后机房到用户家是通过HFC网络。但是在实验室中存在的问题是： modem资源比较紧张，很多工程师都在去用modem资源利用，如果能够实现modem可以由工程师在web上自动去切换或者预约modem，这样就比较好。所以我们充分利用Rf-switch可以实现自动

换的作用 的功能，就可以实现自动

换modem供给工程师去用。然后输出是通过ds 、us 混频出去，去接fiber note。

A description...

A description...

A description...

A description...

1. 关于工程的框架实现：
2. Sdn

软件定义网络，把数据层和控制层分开，分为3层，最底层：数据转发平面，提供物理连接网络并根据转发规则转发数据，中间层是控制平面，是一个软件层，负责提供网络基础

务，提升网络的可编程性。上层：应用层负责网络的管理、优化和使用。

1. OpendayLight

处在SDN的控制平面，具有拓扑管理，交换机管理，路径转发，主机管理、网络资源切片管理这5哥基础功能。支持Openflow,Lisp,BGP等协议，多个控制器之间可以采用集群的工作方式。ODL采用基于osgi的karaf作为实现架构，使用maven管理代码，允许在控制器运行时进行能功能模块的安装、删除和更新。下图是ODL架构图：

ＯＤＬ在设计上，自底向上分为４层：南向协议层、服务抽象层、控制层，以及北向接口层。框架视图如图１所示。南向协议层包含Ｏｐｅｎｆｌｏｗ、ＬＩＳＰ、ＧＢＰ等协议插件，负责与实际的网络设备进行通信、收集设备信息、监听相应端口。服务抽象层SAL：屏蔽了不同南向协议插件之间的差异，负责路由和适配的工作，使得南向协议层呈现统一的北向接口控制层的调用。控制层：负责交换机管理等拓扑结构管理等。北向接口层为网络应用程序提供访问的接口，ODL采用REST API

A description...

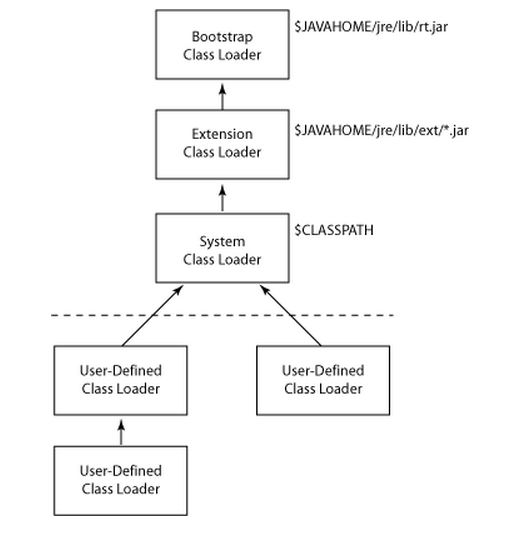
1. OSGI及Karaf
2. OSGI <http://www.sdnlab.com/odlcommunity/question/62> <http://blog.csdn.net/liubag/article/details/22825127>

是什么导致了OSGI的产生和发展？？

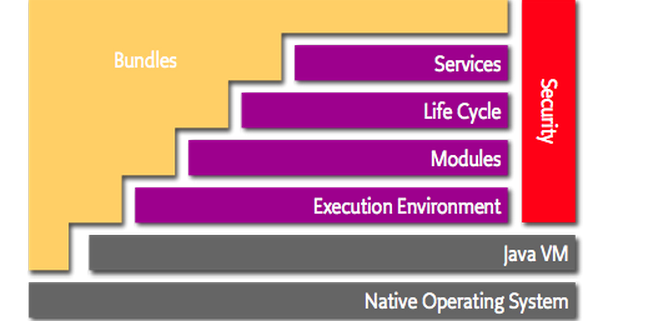
Java的类加载器(Classloader)是一种分层结构，如下图所示，分为引导类加载器(Bootstrap Class Loader)，扩展类加载器(Extension Class Loader)，系统类加载器(System Class Loader)以及用户定义的类加载器(User-defined Class Loader)。引导类加载器在JVM时负责加载rt.jar里面的类，扩展类加载器负责加载在扩展目录下的jar文件中的类，系统类加载器则在Classpath上面搜索类加载器，用户定义的类加载器则从用户指定的路径（比如一个网络URI）加载类。在该类加载体系中，一个类加载器总是先去上层类加载器加载类，一层一层迭代，当无法找到需要的类时在自己加载。

在这种类加载机制中，存在以下几个问题：1）类版本冲突：当类路径上存在同一个类的不同版本时，如果类加载器找到一个版本，则不再搜索加载下一个版本；2）无法确定jar之间的依赖关系：现有的JAR

准中缺乏对与Jar文件之间依赖关系的定义支持，因此只有在运行时间无法找到所需的类时，才会打出java.lang.ClassNotFoundException，但这通常不能有效帮助开发人员解决问题；3）信息隐藏：如果一个jar在类路径上并且被加载，那么所有该jar中的公共类（public class）都会被加载，无法避免某些类被隐藏从而不被加载。尽管在J2EE中改进了类加载机制，可以支持以war或者ear应用为单元进行加载，但是这些问题还是没有被很好地解决，并且热部署效果让人忧心。



      OSGi就是为了克服这些问题而生，却又不局限与这些问题，对Java开发人员而言的确是好多顿大餐。OSGi是一个动态的Java模块（Module）系统，它规定了如何定义一个Module以及这些模块之间如何交互。每个OSGi的Java模块被称为一个bundle。每个bundle都有自己的类路径，可以精确规定哪些Java包和类可以被导出，需要导入哪些其它bundle的哪些类和包，并从而指明bundle之间的依赖关系。另外bundle可以被在运行时间安装，更新，卸载并且不影响整个应用。通过这种方式，分层的类加载机制变成了网状的类加载机制。在应用程序启动之前，OSGi就可以检测出来是否所有的依赖关系被满足，并在不满足时精确报出是哪些依赖关系没被满足。



OSGi service platform是一个开放并且提供统一接口

准的体系框架，基于这个体系框架，服务提供商，程序开发人员，。软件提供商，服务网管运营商，设备提供商能够协调地联合起来开发，部署以及管理向用户提供的各种

务.

灰色部分，是OSGI的基础，可以在任意一个本地操作系统中安装有Java虚拟器上可以运行OSGI.

Execution Environment：是CVM + CDC + FP + PP组成的JVM配置。

Module：这一方面功能将主要负责bundle的安装部署，更新和卸载，以及bundle在设备的物理存储（如果有的话）。在这个层次，每个bundle都是独立的，它的安装，升级和卸载完全不依赖任何其他bundle，这点framework提供了强大的隔离性。

Life Cycle专门负责对bundle的解析（比如关联两个有相互依赖关系的bundle），启动（相当于运行应用程序）和停止（相当于停止应用程序）。这个层次中，bundle间的逻辑关系被创建起来，这些关系能否成功的创建，将会直接影响bundle的成功解析和启动。Service Registry：可以认为是一个数据库，bundle启动后，可以向这个数据库注册它动态提供的服务。只要bundle不被停止，且bundle不主动撤销注册的服务，这个

务将一直保存在这个数据库中供其它bundle来查询和使用。而Services就是由bundle运行时提供的具体服务对象，这些服务对象的存在，使得framework具有极其动态的特征，并为framework运行时提供灵活强大的功能。  
Jar文件是bundle的唯一格式，也就是说，我们要运行bundle，必须把代码打成jar文件。而jar文件可以带有manifest文件，这个文件对bundle是不可缺少的。

1. Karaf

Karaf是Apache旗下的一个开源项目.Karaf同时也是一个基于OSGi的运行环境,Karaf提供了一个轻量级的OSGi容器,可以用于部署各种组件。OSGi是OpenDaylight的旧版本，就在氢版本中，他加载每一个模块，如OpenFlow plugin，GUI，controller等；  
Karaf用在Helium氦版本中，他可以加载你需要的模块。Karaf工具是基于OSGi的OpenDaylight特性容器，用于实现OpenDaylight各功能组件的热插拔

1. 我所做工作部分
2. Mongodb

Mongodb效率高的原因：首先是内存映射机制，数据不是持久化到存储设备中的，而是暂时存储在内存中，这就提高了在IO上效率以及操作系统对存储介质之间的性能损耗。（毕竟内存读取最快）其次，NoSQL并不是不使用sql，只是不使用关系。没有关系的存在，就表示每个数据都好比是拥有一个单独的存储空间，然后一个聚集索引来指向。搜索性能一定会提高的。

Insert方面MongoDB比SQL Server快十倍左右，Update快20倍左右，Select快2倍左右。mongodb存储量很大可以达到P级别，10TB足够用数据表和数据集，对于存储的大多是文字类的消息是可以用no-sql用类似json的Bson格式存储数据，如果要是存储一片博客的话，包括内容和评论那么可能nosql只在一个document里就存储了，而sql则可能要存储在几个变中。预定义结构VS.动态结构.sql需要预定义好这样的数据结构，后期如果再修改会很麻烦，而nosql是可以动态数据存储。结构化查询和非结构化查询。关系型表中主键的概念对应非关系存储中的文档Id。SQL数据库使用预定义优化方式(比如列索引定义)。

例如：存取一篇博客这里我们举一个例子，博客文章。大多数人在编程的时候会使用“类/对象”的架构来进行建模，但使用关系型数据库来存储博客数据就会遇到麻烦，因为每一个输入的元素会分散到多个表当中。如下图所示：

A description...

用MongoDB，你的博客文章就能够存储在一个单一的集合当中，每一个元素都很明显。

主要实现的功能是：增删改查以及按照时间在指定的时间内后台数据会自动删除等。

难点以及自己认为比较难实现的地方：就是数据库管理和自动删除的功能

数据库关联的实现：两种方法人工关联和Dbref因为想到利用之前的代码，就想到用人工关联的方式

情景：，device table和group table，device table里面存储的是设备的name,ip,type,group 等，group table里面存储的是group的id A description...

实现思路：根据device的index去查grouptable中对应的index的group 名字是什么,有个for循环遍历，先将所有的数据得到放到list里面，然后得到每一个DBobject对象，然后把{ "deviceid" : { "$oid" : "55b01e08e4b04009dcbca85c"}变成{ "\_id" : "55b01e08e4b04009dcbca85c" }这种类型，然后去得到index,再用这个index去查对应的name,再put到原来的list里面。实现关联。在edit group的时候只是名字改变，但是index

变。

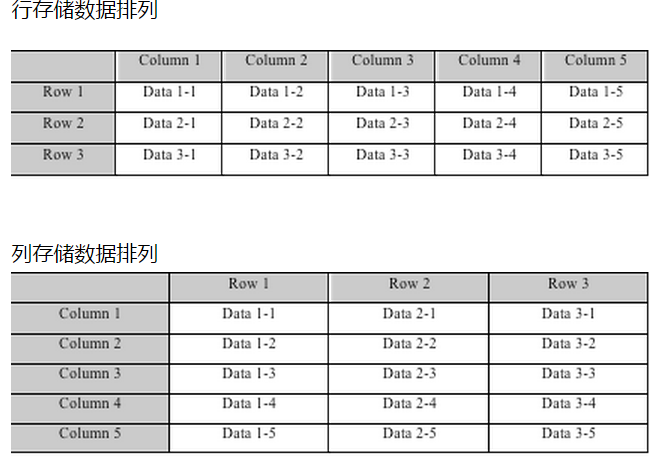
后台数据库自动删除： 用来Mongodb的ttl(文档的存活时间)，TTL依赖一个在mongo中的后台线程，该线程读取索引中的类型的值，从集合中删除过期的documents，每60s运行一次。可以设置在数据插入后的一定秒数后过期，也可以设置文档在确定的时钟时间过期（Date）.实现方法是对某个字段创建一个索引。

A description...

Capped collection:可以规定一个colleciton的最大的存储空间和最大document数量。

延伸—行存储和列存储：

大数据存储一般：行存储和列存储。关心数据库已经不适应这种巨大的存储量和计算要求，基本淘汰。其中Hadoop的HBase是列存储，MongoDB是行存储。他们的结构布局：



行存储是一行为单位，列存储是一列为单位。行存储读写都是从第一列开始最后一列结束。列存储的读取是列输出的一段或者全部数据，写入时，一行记录被分成多列，在每一列数据追加到对应列的末尾。

写入：行存储是一次写完，保证数据的完整性，列存储把一行数据拆分成单列保存，写入次数明显比行存储较多。同样数据修改也是类此写入，行存储仍然有较大的优势。

读取/查询：行存储读取时是将一行数据完全读取，如果只需要其中几列数据，就会存在冗余列，消除冗余列的过程是在内存中进行的，当数据量大时会影响到处理效率，而列存储则

会产生冗余数据，这对数据完整性要求不高的大数据处理领域很重要。

改进：行存储，在用户定义数据时避免冗余列的产生，优化数据存储记录结构。

列存储，计算机上可以安装多块硬盘，以多线程方式读写；对数据完整性问题：可以在写入过程中加入类似关系数据库的“回滚”机制，当某一列发生写入失败时，此前写入的数据全部失效等。

A description...

1. SNMP实现
2. 换：

Snmp协议介绍：snmp是一个从网络上手机设备信息的功用通信协议，并将这些信息记录存储在MIB中，信息包括设备的特性，数据吞吐量等。

我们来做一个最简单的Snmp操作：获取一台机器的机器名，假设其IP是192.168.1.1。首先要给192.168.1.1安装Snmp环境。Window组件的管理监视工具里包含Snmp，只需要安装一下就可以了。Snmp默认的通信端口是161/162，有的系统需要手动设置，这个可以通过创建筛选器列表来实现，具体怎么设置网上很多，大家可以去搜索一下。另外需要关闭192.168.1.1的防火墙。接下来需要一个能够进行Snmp通信的工具，比较简单的有snmputil，大家可以去网上下载。  
snmputil的命令规则是：  
snmputil [get|getnext|walk] agent community oid [oid ...]  
[get|getnext|walk]为消息类型，我们此次进行的操作是get  
agent

Snmp代理即你想进行操作的网络设备的ip或名称，即192.168.1.1  
community：分区域，即密码，默认是public  
oid：想要操作的MIB数据对象号，设备

称对应的MIB对象号是.1.3.6.1.2.1.1.5.0打开命令行窗口，进入snmputil所在路径，键入snmputil get 192.168.1.1 public .1.3.6.1.2.1.1.5.0如果参数都正确，控制台就会显示出192.168.1.1的机器名。

SNMP常用到的命令：

Get：管理站读取代理者处对象的值。它是SNMP协议中使用率最高的一个命令，因为该命令是从网络设备中获得管理信息的基本方式。

Set：管理站设置代理者处对象的值。它是一个特权命令，因为可以通过它来改动设备的配置或控制设备的运转状态。它可以设置设备的名称，关掉一个端口或清除一个地址解析表中的项等。

Trap：代理者主动向管理站通报重要事件。它的功能就是在网络管理系统没有明确要求的前提下，由管理代理通知网络管理系统有一些特别的情况或问题发生了。如果发生意外情况，客户会向服务器的162端口发送一个消息，告知服务器指定的变量值发生了变化。通常由服务器请求而获得的数据由服务器的161端口接收。Trap 消息可以用来通知管理站线路的故障、连接的终端和恢

、认证失败等消息。管理站可相应的作出处理。

Walk:比方取.1.3.6.1则这个下面的所有信息都取到了，顺序依次从做到右，包括system interfaces ip cisco Microsoft juniperMIB等都可以获取到。

getBulk:这个和walk是一样的

getNext：是获取下一个节点。

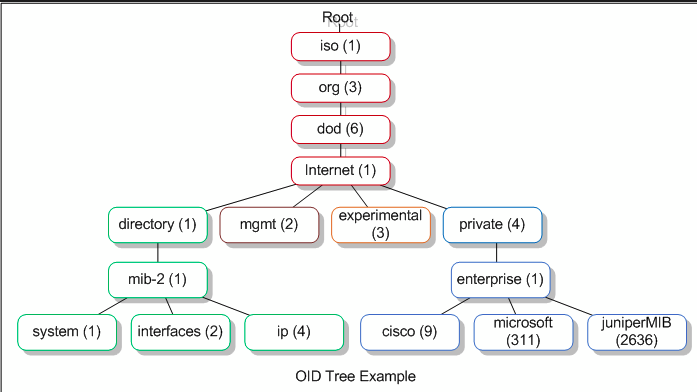
基本概念：

PDU：协议数据单元，是SNMP的数据区，即snmp通信时报文数据的载体。

MIB：管理信息库，是一个信息存储库。包含了代理中的有关配置和性能的数据。是一树状结构，类似文件系统，但是文件系统可以以相对路径，而MIB只能是绝对路径。

OID：每一个节点都有一个对象标识符（OID）来唯一标识，每个节点用数字和字符两种方式显示，比如网络设备名的oid是.1.3.6.1.2.1.1.5.0，其值为设备

称的字符串。下图就是oid图只有叶节点才会有字符数据。



1. Web ui以及LDAP

LDAP:是一种轻量级目录访问协议，目录

务是一种特殊的数据库系统，其专门针对读取浏览和搜索做了特定的优化处理，是一种基于TCP/IP的访问协议。LDAP以树形结构存储，树根通常是com往下则可能是一个或者多个组织单元，一个组织单元可能包括所有雇员或者打印机信息等。

实现思路：先获取username然后根据username去是不是有个这个name如果有，返回dn然后去匹配密码看看密码是不是一致，就Ok了。

类似一个雇员信息：

雇员信息如下：

　　　　dn: cn=Ray D. Jones, ou=Employees, o=Acme, c=US

　　　　objectClass: inetOrgPerson

　　　　cn: Ray D. Jones

　　　　sn: Jones

　　　　telephoneNumber: 444-555-6767

　　　　mail: jonesrd@acme.com

　　　　localityName: Houston

　　　　dn: cn=Eric S. Woods, ou=Employees, o=Acme, c=US

　　　　objectClass: inetOrgPerson

　　　　cn: Eric S. Woods

　　　　sn: Woods

　　　　telephoneNumber: 444-555-6768

　　　　mail: woodses@acme.com

　　　　localityName: Houston

1. Load-balance

load-balance的原理：load-balance，顾名思义为负载均衡，在CMTS中，当有多个channel同时使用时，如果存在有的channel负荷超重（流量大），有的channel负荷轻（流量小），则会将channel负荷超重的cable modem移到较轻负荷的channel工作，最终达到各个工作channel的负荷均衡。

1. JSP

并发，网络安全

(1)Web

务器如何调用并执行一个jsp页面： 浏览器向服务器发请求，不管访问的是什么资源，其实都是在访问Servlet，所以当访问一个jsp页面时，其实也是在访问一个Servlet，服务器在执行jsp的时候，首先把jsp翻译成一个Servlet，所以我们访问jsp时，其实不是在访问jsp，而是在访问jsp翻译过后的那个Servlet。当浏览器访问index.jsp时，服务器首先将index.jsp翻译成一个index\_jsp.class,在tomcat

务器下可以看到index\_jsp.java，这个类是继承HttpJspBase，这个类又是继承HttpServlet的，其实就是在访问index\_jsp，就是访问这个servlet这个servlet使用\_jspService来处理请求。(2)Html标签是如何发送到客户端的:是在\_jspService方法中使用out.write()输出给浏览器。在jsp中编写的java代码和html代码都会被翻译到\_jspService方法中去，在jsp中编写的java代码会原封不动地翻译成java代码，如<%out.print("Hello Jsp");%>直接翻译成out.print("Hello Jsp");，而HTML代码则会翻译成使用**out.write("<html标签>\r\n");**的形式输出到浏览器。在jsp页面中编写的html排版标签都是以**out.write("<html标签>\r\n");**的形式输出到浏览器，浏览器拿到html代码后才能够解析执行html代码。

(3)web

务器在调用jsp时，会给jsp提供8个java对象，如PageContext pageContext;HttpSession session;ServletContext application;ServletConfig config;JspWriter out;Object page = this;HttpServletRequest request, HttpServletResponse response

1. Servlet <http://blog.csdn.net/caohaicheng/article/details/38116481>
2. 概念：servlet是基于协议的请求和相应
3. 务的java类，http servlets是能被http命令调用的servlet。生命周期：servlet类加载（加载servlet，调用inti()方法）、请求到达时运行service()方法会自动派遣运行与请求想对应的doGet或者doPost方法、销毁（当
4. 务结束后调用servlet的distroy()方法）
5. 方法：request.getAttribute(获取到的是Object的对象) ，request.setAttribute(是应用服务器把这个页面放到该页面所在对应的以块内存中，当你的页面服务器重定向到另一个页面的时候，应用服务器会把这块内存拷贝到另一个页面所对应的内存当中，类似serssion)和request.getParameter(获取的是字符串而不是对象)
6. Servlet和cgi的区别：Servlet与CGI的区别在于Servlet处于服务器进程中，它通过多线程方式运行其service方法，一个实例可以服务于多个请求，并且其实例一般不会销毁，而CGI对每个请求都产生新的进程，服务完成后就销毁，所以效率上低于Servlet。CGI 的工作方式，从 Web
7. 务器的角度看，是在特定的位置（比如：http://www.example.com/wiki.cgi）定义了可以运行 CGI 程序。当收到一个匹配[URL](https://zh.wikipedia.org/wiki/URL)的请求，相应的程序就会被调用，并将客户端发送的数据作为输入。程序的输出会由 Web
8. 务器收集，并加上合适的档头，再发送回客户端。
9. Forward 和redirect的区别：地址栏显示：（1）从地址栏显示来说 forward是服务器请求资源,
10. 务器直接访问目
11. 地址的URL,把那个URL的响应内容读取过来,然后把这些内容再发给浏览器.浏览器根本不知道服务器发送的内容从哪里来的,所以它的地址栏还是原来的地址.redirect是服务端根据逻辑,发送一个状态码,告诉浏览器重新去请求那个地址.所以地址栏显示的是新的URL.（2）从数据共享来说 forward:转发页面和转发到的页面可以共享request里面的数据.而redirect:不能共享数据.（3）从运用地方来说 forward:一般用于用户登陆的时候,根据角色转发到相应的模块.redirect:一般用于用户注销登陆时返回主页面和跳转到其它的网站等.（4）从效率来说 forward:高.redirect:低.
12. session和cookie

注意联系项目中用到的内容。区别：

Cookie:采用的是客户端保持状态的方案.

Session:采用的是服务器端保持状态的方案，这种方式也需要保存一个sessionId所以必须要也要利用cookie存储一个sessionID

1. http请求以及状态码

1 http请求的动作，get和post;get用于信息获取，使用URL传递参数，对发送的信息数量限制，一般在2000个字符，一般用于查询数据；post一般用于服务器上的资源，对所发送信息的数量无限制。2 请求的url3 请求头，包含客户端环境信息，身份验证消息4 请求体，请求正文，包含要提交的信息

一个Http响应：1 一个数字和文字组成的状态码，表示是否成功2 响应头，表示服务器的信息3 相应体

A description...

1. ajax 创建交互式网页应用的网页开发技术

原理：AJAX的原理简单来说通过XMLHttpRequest对象来向服务器发异步请求，从

务器获得数据，然后用JavaScript来操作DOM而更新页面。这其中最关键的一步就是从

务器获得请求数据。要清楚这个过程和原理，我们必须对 XMLHttpRequest有所了解。XMLHttpRequest是AJAX的核心机制，它是在IE5中首先引入的，是一种支持异步请求的技术。简单的说，也就是JavaScript可以及时向服务器提出请求和处理响应，而不阻塞用户。达到无刷新的效果。

步骤：

function sendReq(url, query) {

var req = new XMLHttpRequest();

req.open("POST", url, false);

req.setRequestHeader("Content-type","application/json");

var get\_response = jQuery.parseJSON(req.responseText).output;//jQuery.parseJSON（）类似eval（）把json转成js对象

return get\_response;

}Ajax应用程序的优势在于：

1. 通过异步模式，提升了用户体验

2. 优化了浏览器和服务器之间的传输，减少不必要的数据往返，减少了带宽占用

3. Ajax引擎在客户端运行，承担了一部分本来由服务器承担的工作，从而减少了大用户量下的服务器负载。

AJAX应用和传统WEB应用有什么不同？

在传统的Javascript编程中，如果想得到服务器端数据库或文件上的信息，或者发送客户端信息到服务器，需要建立一个HTML form然后GET或者POST数据到服务器端。用户需要点击”Submit”按钮来发送或者接受数据信息，然后等待服务器响应请求，页面重新加载。 因为

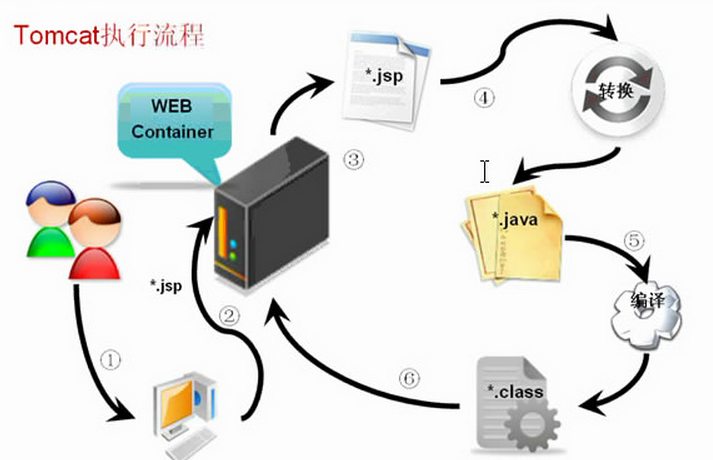
务器每次都会返回一个新的页面， 所以传统的web应用有可能很慢而且用户交互不友好。使用AJAX技术， 就可以使Javascript通过XMLHttpRequest对象直接与

务器进行交互。 通过HTTP Request， 一个web页面可以发送一个请求到web

务器并且接受web

务器返回的信息(不用重新加载页面)，展示给用户的还是通一个页面，用户感觉页面刷新，也看不到到Javascript后台进行的发送请求和接受响应。

1. tomcat执行流程以jsp为例



第一次执行：

1. 客户端通过电脑连接服务器，因为是请求是动态的，所以所有的请求交给WEB容器来处理
2. 在容器中找到需要执行的\*.jsp文件
3. 之后\*.jsp文件通过转换变为\*.java文件
4. \*.java文件经过编译后，形成\*.class文件
5. 最终
6. 务器要执行形成的\*.class文件

第二次执行：

1. 因为已经存在了\*.class文件，所以不在需要转换和编译的过程

修改后执行：

 1.源文件已经被修改过了，所以需要重新转换，重新编译。

Jsp和servlet都可以开发动态web资源，但是由于技术的特点，servlet只是负责相应请求数据，并且数据通过转发给jsp，数据显示用jsp去做。有了jsp，在MVC项目中servlet不再负责动态生成页面，转而去负责控制程序逻辑的作用，控制jsp与javabean

**二、面向移动终端流媒体系统的研究**

1. linux 相关
2. 交叉编译

由于通常宿主机系统和目标系统的 CPU 不同，直接编译生成的可执行文件格式是不同

的，无法直接在目标机上运行。因此需要在宿主机上根据目标机的执行环境建立相应的交

叉编译环境，不同的目标机设备它们的交叉编译环境也是不同的。

1. 命令相关

查找：

Find：查找指定目录或文件的命令。Find <指定目录> <指定条件> <指定动作>

find .-name “my” 搜索当前目录含子目录下，所有文件名是my开头的。

Whereis：查找指定的文件源和二进制文件和手册等。Whereis grep

Which：用于查询命令或别名的位置。是在PATH变量指定的路径下，搜索某个系统命

令的位置

Locate：快速查找系统数据库中指定的内容。和find –name 差不多的，只不过速度更快因为是在数据库中查找，

Grep：在指定的文件或标准输出，标准输入内，查找满足条件的内容。

grep –n “str” –r ./ //查找当前目录下以及下属子目录下所有包含str字符串的文件,会列出文件名.以及该行的内容.以及行号

文件权限：

chmod abc file 其中abc分别表示user、group和other的 权限，其中r（读权限）=4、w（写权限）=2、x（执行权限）=1；如果是可读和可写可执行并且，user group 和other都可则4+2+1=7，则chmod 777 file1

网络有关：

netstat:打印linux中网络状态信息，netstat –s 显示所有端口统计信息 netstat –st 显示tcp端口的统计信息， netstat –su显示udp端口的统计信息；

进程有关：ps –ef | grep java 查看所有进程中是java的进程信息 kill -9 PID //-9表示强迫进程立即停止。

前后台切换：

Ctrl+Z 暂停前台命令的执行，将该进程放入后台，回到SHELL

jobs 查看当前在后台执行的命令，可查看命令进程号码

&运行命令时，在命令末尾加上&可让命令在后台执行

fg N 将命令进程号码为N的命令进程放到前台执行，同%N

bg N c将命令进程号码为N的命令进程放到后台执行

1. 配置网络

1 sudo  gedit /etc/network/interfaces (有的是把gedit 换成vim命令，用vim命令是最后修改后如果要保存，用Esc 然后:wq,如果是：q则是不保存直接退出)

前提是桥接网络方式进行上网。

打开后添加：

auto lo

iface lo inet loopback   
auto eth0   
#ifac eth0 inet dhcp   
iface eth1 inet static   
address 192.168.1.8   
netmask 255.255.255.0    
geteway 192.168.1.1

dns-nameservers 8.8.8.8

2 sudo gedit /etc/resolv.conf

打开后添加：

nameserver 8.8.8.8

(如果这部分不添加的话，重启后会失效)

3 /etc/init.d/networking restart

开发步骤：

1 usb串口进入终端，配置地址：ifconfig eth0 192.168.1.1 route add default gw 192.168.1.1

2另外在一个本机终端中启动挂载:sudo /etc/init.d/nfs-kernel-server start

3 nfs挂载，在minicom中mount -t nfs -o nolock 192.168.1.104:/home/frank/hi3518/nfs/ /home

无线网络的时候:

无线网络采用sta模式,文件系统已经更改.

ifconfig eth0 down //先关闭有线网络

ifconfig ra0 up //打开wifi设置网络

iwlist ra0 scanning //扫描无线路由器的节点信息，会将周边无线路由器的节点信息输出

连接某个路由器，以本人的路由器为例子，执行的步骤如下：

ifconfig eth0 down

ifconfig ra0 up

iwpriv ra0 set NetworkType=Infra

iwpriv ra0 set AuthMode=WPA2PSK

iwpriv ra0 set EncrypType=AES

iwpriv ra0 set SSID="NVSHEN"

iwpriv ra0 set WPAPSK="202202202" //无线路由器的密码

iwpriv ra0 set SSID="NVSHEN"

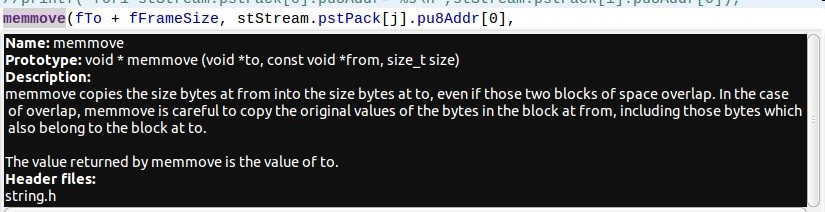
连接成功后配置ra0的IP地址

1. Live555介绍（代码）

live555有一个虚函数，然后我写的类中去继承，然后重写这个函数就可以了。

1. 本课题难点

核心代码如下，刚开始的问题是：因为循环这个地方是一只是执行一次循环，因为认为只有一个数据包，所以vlc看到是那种情况，但是仔细发现，视频中每一帧数据都是一个nalu单元（sps pps除外），而视频刚开始传输的时候H.264的SPS和PPS串，包含了初始化H.264解码器所需要的信息参数，这些信息没有传输过来。还有一个地方是就是memmove函数（如果源和目的参数有重叠, memmove() 提供有保证的行为。而 memcpy()  则不能提供这样的保证, 因此可以实现得更加有效率。如果有疑问, 最好使用 memmove()。）to这个指针干刚开始就是fto没动，然后也没改，没想到移动指针。



1. 实现在c++中调用c的代码

**extern** "C" {

//sample\_venc.h是包含sample\_venc.c程序的头文件

**#include** "sample\_venc.h"

}

1. Android部分介绍
2. NDK
3. JNI
4. FFmpeg移植