

分类号\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 密级\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

UDC \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 学号 2014002061



毕业设计（论文）

|  |  |
| --- | --- |
| 论文题目 | 局域网设备控制管理系统 |
| Thesis Topic | Device Controlling and Management System  in Local Area Network |

|  |  |
| --- | --- |
| 学生姓名 | 周宇豪 |
| 学 号 | 2014002061 |
| 所 在 院 系 | 信息与计算机学院 |
| 专 业 班 级 | 物联网工程1402班 |
| 导师姓名职称 | 李爱萍 讲师 |
| 完成日期 | 2018年6月4日 |

2018年6 月20 日

# 

太　原　理　工　大　学

毕业设计（论文）任务书

第1页

|  |
| --- |
| 毕业设计（论文）题目：  局域网设备控制管理系统  毕业设计（论文）要求及原始数据（资料）：  1．综述国内外局域网设备控制研究现状；  2．深入了解HTTP服务器的相关技术及应用。  3．熟练掌握Go,Java,HTML/JavaScript/CSS的开发语句；  4．设计并实现局域网文件传输，键盘控制，摄像头直播的功能；  5．深入了解计算机程序并发，线程等技术；  6．训练检索文献资料和利用文献资料的能力；  7．训练撰写技术文档与学位论文的能力； |

第2页

|  |
| --- |
| 毕业设计（论文）主要内容：   1. 综述本地服务器客户端开发现状； 2. 分析相关需求，总体设计； 3. 局域网文件传输，摄像头直播，键盘控制的应用和实现；   4．了解并发模型，HTTP服务器架构知识；  5．Android与桌面GUI程序的设计和实现；  6．多个客户端之间相互发现功能的实现；  7.系统测试；   1. 实验结果分析和总结。   学生应交出的设计文件（论文）：  1．内容完整、层次清晰、叙述流畅、排版规范的毕业设计论文；  2．包括毕业设计论文、源程序等内容在内的毕业设计电子文档及其它相关材料。 |

第3页

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 主要参考文献（资料）：  [1]祝翔,董启文,郁可人.基于WebSocket的PK答题的设计与实现[J].华东师范大学学报(自然科学版),2018(02):89-100.  [2]林路智.Go语言搭建网站解析[J].电脑知识与技术,2015,11(29):60-61.  [3].Google Go语言将加速对Android平台的支持[J].电脑编程技巧与维护,2010(17):4.  [4]何兴鹏,刘钊远,陶琛嵘.Linux程序向Android平台移植的研究[J].计算机测量与控制,2018(05):112-115.  [5]江存.基于Linux的2种HTTP服务器实现与对比分析[J].现代计算机(专业版),2017(24):58-61+76.  [6]王伯槐,张烨.基于Go语言的消息推送平台的设计与实现[J].数码设计,2017,6(02):33-36.   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | 专业班级 | 物联网1402班 | | | | | 学生 | 周宇豪 | | 要求设计（论文）工作起止日期 | | | | | 2018年3月18日~2018年6月1日 | | | | 指导教师签字 | |  | | | | 日期 | 2018年3月18日 | | 教研室主任审查签字 | | | |  | | 日期 |  | | 系主任批准签字 | | |  | | | 日期 |  | |

# 

# 1.绪论

## 1.1项目开发背景

起初做这个系统只是为了解决自己日常生活中的不方便，比如说：手机和电脑传文件的时候还需要数据线，我会觉得实在是太麻烦了！于是自己便随手做了一个Android APP，这个App启动后会开启一个HTTP 服务器，服务器能用HTTP协议来传文件，发布到应用市场上之后，没想到尽然收到了大量的好评。当然，评论之中，也不乏批评我做的不好的地方，但绝大多数的评论都是在给我提供一些建设性的意见，哪些地方做的不好，哪些地方可以这样这样改进，是时受宠若惊。于是我，带着不能辜负大众的一片热情的心情，开始花大量的时间去优化，完善这个系统。

时至今日（2018年6月），此系统在酷安应用市场上已经有6万次下载（应用名称为：局域网精灵），6778个关注者，540条评论。同时，我已经把这个App的Android版如今已经上架到各大应用商店了：小米，百度，腾讯。其他应用商店下载量虽然不如酷安平台多，但也算是给使用其他应用平台的用户多一种下载渠道吧，就像我们互联网行业经常对用户说的一样：我可以不用，但是你不能没有，哈哈。虽然这样的一个成绩对于真正的开发者来说并不算多，但是，于我个人而言，是对我自身专业能力的莫大的鼓励。

要知道，在此之前，我尽历了大三下学期的求职阶段。以一个计算机相关专业的本科生的身份来说，大学四年的经历，与全国本科求职者比较，我的简历很普通。难以同其他来自全国各地的本科计算机学生争求同一职位。

我个人的性格是不喜欢与他人竞争，在自己的大学四年里没有参加过任何全国性的竞赛，也没有获得过任何奖学金，成绩也不过及格而已，毫无闪光点。要说我大学四年有何收获，我想应该是想明白了一些人或事，找到了自己生活的理由，给自己的的人生定义了价值取向。我想我以后不会再为“人一辈子活着有什么意义”之类的问题所困扰了。私以为这才是我在大学四年得到的最大收获。毛主席也说过：看待事物要优先抓住主要矛盾。相比漫无目的的奋斗，先找到正确的方向才是我大学阶段亟待解决的主要矛盾。

当然我在大学四年也不是什么都没干，相比课堂和书本上的学习，我更喜欢自己动手在互联网上学习。从高中开始我就有了这样一个念头，如果没有老师教我了，我该怎么学习？这在高中的同学眼中是一个看起来杞人忧天的想法，现状也证明了我当时的想法的前瞻性。从大一开始逼迫自己去自学，锻炼独立思考问题的能力，独立寻找解决办法的能力。授人以鱼不如授人以渔。我认为一个真正的好老师，应该把学生教到不需要老师的程度才行！

本项目便是在我的自学过程之中诞生的一个点子！待此项目完善至终，也正好到了毕业设计的年份了。选了一个适合自己的题目，便顺势将其作为自己大学本科的毕业设计了。

## 1.2项目简介

本文设计的局域网设备管理和控制系统，是基于HTTP协议的一个集文件传输、摄像头直播和输入设备控制为一体的多功能系统。

其功能主要包括：局域网设备之间的文件上传和下载，局域网设备之间的输入设备控制（目前只支持手机端控制电脑端），其中的输入设备包括键盘和鼠标，也就是说，你可以使用手机来控制电脑的键盘和鼠标。

除此之外，本系统还有一个额外的功能，那就是摄像头直播，这里所说的摄像头直播是指：手机将自己的摄像头画面实时直播到局域网上，其他的同一个局域网下的设备都可以查看这台设备的摄像头直播的画面。

## 1.3国内外现状

处于在这个系统做出来之前，市面上并没有一个特别统一的，大众认可的局域网文件传输软件，也就是说，在这个领域，相对来说还是一个空白。

那个时候大家是怎么在电脑和手机之间传输文件的呢？

普通的用户一般是通过QQ，来将电脑上的文件发送到手机上。这种方式有很大的缺点，就是传输速度受到带宽的限制，以我自己的网络为例。使用这种方式的传输速度最高也不过2MB/s，但是如果使用我做的这个系统来传输文件，传输速度可以达到6MB/s甚至更高。

还有一种方式就是通过数据线将手机连接电脑，然后传输文件。这种方法虽然速度快，但是效率低下，当传输一些小文件的时候，大部分时间都花在了找数据线上了。很多用户对此抱怨很深。

高级一点的用户会选择ftp等入门难度高的传输协议。这些协议对于普通用户来说很难理解，更加难以使用。最大的缺点是，在传输之前，你还需要给电脑安装一个ftp服务器。然而我这个系统在传输的时候，接受者是不需要安装任何客户端的，只要有浏览器就能进行所有的操作。况且浏览器是每一个操作系统都内置的软件。

可以说我这个系统做出来以后，受到大家欢迎的主要原因是解决的一部分网友日常生活中的痛点，抓住了痛点之后，大家自然喜欢。

## 1.4应用场景

PPT遥控器。平时大家在做PPT演讲的时候，都是用电脑插上投影仪，然后使用鼠标来切换PPT的下一页。这样很麻烦，因为演讲者不能离开电脑，必须在切换下一页的时候，回到电脑旁边，用鼠标点击才能切换。那有了本系统之后，用户就可以实现用手机遥控电脑，只要手机在手里，就可以不需要回到电脑旁边，一直演讲下去，对演讲者的站位会更加自由。

摄像头直播。因为本系统可以实现局域网内的摄像头直播，所以，只要让手机和电脑通过WiFi连接在同一个局域网下，然后将手机的摄像头画面直播到局域网，电脑上就可以看到摄像头的画面。因为电脑本身携带不方便，很难移动，这样做可以实现用电脑观看隔壁房间的摄像头画面。因为是WiFi连接，所以不需要任何线连接就可以实现手机变成移动摄像头这一梦幻功能。

文件传输。在我们的日常生活中，手机和电脑传文件一直是非常不方便的。有了我们做的这个系统之后，手机和电脑可以轻松的通过WiFi来传输文件，而且稳定高效，不用担心数据线中途断开等问题。

另外，剪切板在电脑和手机之间的传输也是我们日常生活中的一大难题。如果电脑不安装QQ等软件很难传输，所以我们完全实现了在Web端的剪切板共享，从此，如果你在手机上看到什么网址，文字之类的，在手机上复制了之后，立马可以在电脑上接收到了，十分方便。

## 1.5本章小结

本章主要介绍了本系统的主要的三大功能。以前经常在互联网上阅读一些关于产品经理的思维方式的文章，颇有感触。就拿这些市面上的面向消费者的应用程序来说吧，我们以前是不叫应用的，我们以前都叫软件，或者程序。但是自从移动互联网到来之后，我们开始把软件和程序称之为应用了，这个词是来源于国外的单词Application。虽然是直接从字面意思翻译过来的，但是如果我们以我们以往的认知来理解应用这个词还是感觉与真正我们要指代的意思是有失偏颇的，但是我们中国人对于一些新兴的词汇接受的速度是非常非常快的，这也是为什么移动互联网能够把像我们父母那一批以前在桌面互联网时代对于互联网一窍不通的人群能够带入进我们中国大批的网民大军里面来，因为当你开始重新创造一个词汇或者一个旧的词汇，但是我们给他赋予一个全新的意思（相对我们以往的认知来说），这个时候，人们对于该词汇的反应其实是和面对一个英文单词的反应是一模一样的，就好比把他当成了一个全新的单词开始认识。婴儿为什么学语言学的比大人快呢？因为他们每天都是在接触新的东西，所以就算是老人，当把他们放到一个全新的环境里面去，里面全都是他们有生之年没有遇到过的新事物，他们的学习速度也会非常快的。所以说年龄不是我们学习速度慢的接口，只是环境所至。

随着移动互联网的兴起，不但中国网民的数量越来越多了，每个人家里平均的电子设备数量也越来越多了，大家的家里现在大多数都不止一台手机，也许有两台手机，或者一台手机，一台电脑，甚至家里每个人都有一台手机，每个人都有一台电脑。当百姓家里的电子设备多了之后，对于电子设备的管理和控制就成了我们的新问题。如何才能高效的给家里的其他电子设备传输自己刚才下载的一个文件？如果快速的分享自己看到的一段文字到其他的设备上？这些都成为了我们每天都会遇到的小问题。所以，本文所介绍的系统，就是用来解决这些用户看似很小，实际上能够大大提高百姓日常生活中的工作效率的痛点的。

所以我认为本文所介绍的系统对于大部分家里有多台电子设备的人们是非常有意义的。

# 2.需求分析

## 2.1目标分析

随着互联网时代的到来，人们开始花越来越多的时间泡在网上。过去我们其实对于互联网是非常负面的态度。你比如说，过去，我们会把那些整天泡在网络上的人称为“网虫”，因为虫子移动速度慢，就好像他们每天坐在电脑面前，身体瘦弱，每天只有吃饭和上厕所的时候会离开电脑，开始走动，而且走动的时候都是慢吞吞的，迈着疲惫的步伐，就和毛毛虫一样，所以大家会叫这一类人“网虫”。然而今天，处在互联网盛世的我们，如今再回过头来看这一类人，其实会觉得心里很不是滋味。当初我们那么鄙视这一类人，他们每天面对的是家长的指责，还有朋友的嘲笑，如今这些最早接触互联网的人已经赚的盆满钵满了，那些没有赶上互联网潮流的人们到了现在也只能叹息。其实，仔细想一想，你会发现，每一次人类的科技革命都是发生在一些普罗大众之外的边缘群体，这些边缘群体都有一个共同的特征，他们喜欢接触新的事物，不会被世俗的眼光所困惑，不会被世俗的价值观所禁锢，而拥有这些特征的人群，基本上都是年轻人。这也是为什么马云走访各国演讲，每次都会强调要大家重视年轻人，正是因为年轻人没有包袱，再加上一腔热血，想象力比老人丰富，未来永远是属于年轻人的。

在21世纪这样一个数字时代，我们每个人都离不开互联网，就好像我们现在每家每户如今都离不开电一样。每个家庭至少有两部电子设备，这两部电子设备，或许是电脑，或许是手机。他们都或多或少地承载了当下人们每天的生活，工作，娱乐，和办公。我们每一个人，似乎都开始变成“网虫”了。当然，一旦家里的电子设备开始多了起来之后，我们对于电子设备的管理就成了新的问题。如何让局域网下的各个设备更加灵活的连接起来，是我们要解决的问题。

## 2.2功能分析

### 2.2.1功能简述

用户可以自由地分享和接受其他同一局域网下的设备的上传过来的文件。同时，其他同一局域网下的设备可以自由的访问分享者设备当前剪切板的具体内容。

当电脑客户端点击开启了受控端选项之后，同一局域网下的其他移动设备就可以发现，并向开启了受控端选项的电脑设备发送键盘指令，也就是我们通常所说的键盘控制。

用户开启应用之后，访问摄像头直播的Web页面，点击直播，即可向所有同一局域网下的其他设备广播自己摄像头所拍摄到的画面。

### 2.2.2主要功能

1. 文件传输。用户打开App之后，选择用户想要分享的文件，这个时候App会在后台启动一个HTTP服务器，其他在同一个局域网下的其他设备只需要输入分享者的局域网IP地址，就可以访问用户刚才分享出去的文件，可以下载，还可以上传文件到分享者的手机上面去，电脑同理。
2. 键盘控制。这一个功能需要电脑客户端先开启受控端选项，因为我们必须在受控者允许的情况下，才能让他接受其他设备发送过来的键盘指令。在受控端开启了受控选项之后，同一局域网下的其他设备就会自动发现有新的受控端设备上线了，然后其他设备就可以直接向该受控端发送键盘指令了。
3. 摄像头直播。这一个功能是基于Web的，也就是说，只要局域网下有一个客户端开启了Web服务，其他的设备通过访问这个Web服务器，就同时都拥有了直播者和观看者两个功能了。在其他设备进入摄像头直播的Web页面之后，可以选择到底是要成为直播者，还是观看者。

### 2.2.3主要技术

本文设计到的技术比较多，而且很复杂，下面我们分别详细介绍一下各个技术。

1. HTTP协议

HTTP协议其实是一个非常古老的技术了，过分一点的说，这个其实是上个世纪的技术了，但是不能说因为这个技术太过古老了，我们就不应该用它。电还是上上个世纪的产物呢，我们现在21世纪不也是照样还在用电呢？

如今Google等各大互联网公司一直在向大家建议使用一种新的HTTPS协议，其目的是为了让互联网更加安全，因为HTTPS是在HTTP的基础之上加了一个SSL加密层。以我个人的体验来说，HTTPS虽然安全性提高了，但是有一个致命的缺点，那就是传输速度[1]。由于HTTPS使用到了非对称加密方式，众所周知，非对称加密方式的加密和解密的速度会大大降低，同时，传输速度还随着加密等级成指数上涨。所以说HTTPS是不适合我们这个系统的，原因有二：其一，HTTPS传输速度慢，会影响文件传输模块的传输速度；其二：HTTPS是需要一个共有证书的，大部分证书都需要花钱买，虽然有免费的证书吧，但是致命缺点是每过几个月这个免费的证书就会失效，这显然是不适合我们这种局域网下的Web项目的。既然都在一个局域网了，HTTPS那种安全等级其实就没有必要了，因为局域网和局域网之间是有一个网关进行隔离的，如果有黑客想要对局域网下的设备进行攻击的话，是需要先把局域网的网关攻破才能有资格去进行下一层设备的黑客攻击。所以说局域网下其实对安全性的要求并不是很高。而且HTTPS的证书也很麻烦。

如今很多追求网速的网站依然使用的是HTTP协议，不是因为他有多么好，而是因为他兼容性好，简单，快速。作为一种基于TCP/IP协议的应用层协议，HTTP对于数据的封装是经历过了时间的检验的。以往我们如果想在两个设备之间发送数据的话，是直接建立Socket连接，也就是普通的TCP连接。因为TCP协议是基于连接的，所以你在建立连接之后可以间断性的发送数据和接受数据。但是问题来了，如果你要间断性的发送数据，你如何检测两个数据之间的间隔呢？如何区分接收端收到的数据到底是一个数据还是两个数据呢？HTTP就能解决这种数据包封装的问题，HTTP协议支持很多种数据编码封装格式。比如说我们最常用的application/x-www-form-urlencoded编码格式，这个编码格式主要是用于对表单数据进行编码封装。他会在不同的数据之间加入&符号，用来将多个表单数据分隔开来。那玩意表单的内容就包含&怎么办？这个时候，HTTP协议会将表单里的具体内容用url编码方式进行编码，这种编码方式可以保证不会出现&等特殊字符，如果数据内容里面已经包含了特殊字符的话，他会将这些特殊字符转换成百分号开头的标识符，这样就实现了表单数据的编码与封装。

1. JavaScript和HTML5标准

JavaScript做为一门动态技术，一开始其实是符合Web行业的要求的。但是随着Web技术日新月异的发展速度，大家发现JavaScript越来越不适应当前Web行业的需求了，一个是因为JavaScript的性能问题越来越突出，另外一个是因为JavaScript的语法还不够灵活。但是后来Google的Chrome团队推出的V8引擎，拯救了JavaScript。不仅让他的运行速度大大提升，同时也大大推动了新的Web技术的普及，因为这个技术的出现，chrome浏览器的市场占有率一下子成为了第一名。

Google作为HTML5技术的推动者，我们非常庆幸chrome浏览器是由这个公司开发的，因为没有Google对Web技术的大力推动，也许我们今天依然使用的是HTML4标准的浏览器，也许我们今天依然需要在网页上下载一个Flash插件才能看网页视频。Google作为一个全球性的科技巨头公司，对于人类科技的发展的贡献无疑是巨大的。

当年推动HTML5标准普及的另外一个重要的原因，是来自全球各地各大技术黑客接二连三地向媒体曝光出来的Flash漏洞，这无疑为Flash离开历史舞台敲响了警钟。当人们开始抱怨网页Flash看视频的时候，CPU风扇一直转个不停，当人们开始担心自己的电脑可能因为开启了Flash插件而遭到木马病毒的入侵的时候，HTML5标准开始悄然诞生。HTML5的标准的诞生，不仅仅是为了解决Flash领域的各种问题，同样也是为了不让互联网这一21世纪最伟大的发明就此堕落。要知道一个技术的发明，可以带动多少产业的更替，可以创造多少就业，虽然也会杀死很多行业，但这是历史的进程，不可改变。

HTML5标准给Web行业带来了生机！我们终于可以不安装任何插件，就能在网页上面看视频了。我们终于可以不安装任何插件就可以在网上玩游戏了。HTML5标准的到来，为互联网的媒体娱乐提供了很好的技术基础。与此同时，HTML5也给JavaScript续了一命，给JavaScript带来了很多新的API和函数，让JavaScript在成为无所不能的编程语言的路上又前进了一步。

HTML5标准中新增的摄像头等多媒体接口就是我们本文所做的系统实现的必要基础，没有了这个接口，我也不可能做出这样的系统来。

1. WebSocket技术

WebSocket同样是随着HTML5标准一起来到大众的视野中的。但是WebSockey走的更远，他没有局限的Web浏览器上面运行，而是成为了一种数据通信的新兴标准。在目前的后端领域，人们在写Web API的时候，出来使用基于HTTP的Restful API之外，人们多了一个选择，那就是WebSocket。

初看WebSocket，大家可能会觉得这就是一个让JavaScript创建TCP连接的小玩意，为什么会如此流行呢？对，没错，WebSocket底层协议实现确实不复杂，和Socket连接相似，但是他却不止如此。WebSocket在Socket连接的基础之上增加了对数据的封装过程，也就是WebSocket中的Message机制。这个机制是必要的，他成功的解决了Socket连接中数据的编码封装问题。而且WebSocket相比HTTP Restful API的最大的优势就是简单，速度快。

1. Go语言

作为一门Google开发的编程语言，他不火是没有道理的。自古以来，那些能够火起来的编程语言，都有一个共同的特点，那就是他们有一个比较硬的后台撑腰。比如说Java就有Sun公司在背后撑腰，Swift就有苹果公司在他的后面撑腰，C#就有微软在后面撑腰。对比和Go语言同一时期发布的全新编程语言Rust，就能看出有一个强硬的后台是多么重要。同样式新出的编程语言，如今Go语言已经稳定在了编程语言排行榜的前20名了，然而Rust依然在50名开外。

当然，一门语言的流行肯定不止上面说的那一个原因，语言本身的特性和风格也是一门语言能否流行起来的因素之一。Go语言也不例外。当人们谈论Go语言的时候，他们在谈论什么？性能？并发？跨平台？静态编译？都不是，其实真正让Go语言流行起来的原因是：简单。

简单这个词虽然只有两个字，但是，对一门编程语言来说，他却包含了非常复杂的内容。我以前有参加过一场微软的Lang.Next大会，会议的主题是关于各大编程语言的未来，和一些新增的特性。像JavaScript、C#、Java等都在大会上有提到过。这些语言在谈论自己新增的特性的时候，都会特别骄傲地说这让我们的编程语言更加地通用，更加强大了。但是我看到的却是，每一门语言都在不断的将其他语言的特性吸收过来，然后加到自己的语言之中，他们称之为语言新版本。这看似进步，然而我却开始担忧了。试想一下，在未来，如果每一个编程语言都这样做，所有的编程语言都开始趋向同质化，是多么恐怖的一件事情啊。当你想要学习一门新的编程语言来拓宽自己的思维，改变自己的思考方式的时候，发现所有的编程语言都是一样的套路，这一定程度上回减少编程界的多样性，也会一定程度的限制计算机编程领域的发展。

在所有的编程语言都在不停地给自己增加特性的时候，Go语言却在给自己作减法。往一个项目里面加东西很容易，然而，要想作减法很难。你必须知道那些是必要的，那些是不必要的。Go语言的设计者在谈论自己设计这门语言的设计哲学的时候，无疑都会提到一个词：大道至简。让一门大众使用的编程语言变得越来越简单，易学，同时保持灵活性，我想这也是Go语言的新用户越来越多的原因，因为学习门槛低，又能够事半功倍，何乐而不为。

Go语言是谷歌开发的一款全新的编程语言，可以在不损失应用程序性能的同事，大大降低代码的复杂度。

他的宗旨是让开发者更加容易地开发出简单，稳定，高效的软件！

为什么选择Go语言？1.性能(并发)，2.跨平台，3.网络库丰富，4.语法简单。

性能(并发)。Go语言针对多处理器系统应用程序专门进行了优化，在使用了Go编译之后，程序可以媲美C或C++代码的速度，而且更加安全，而且还支持并行进程[2]。

编译之后的Go语言代码的运行速度和C语言非常接近，而且编译速度非常快，就好像在使用一个动态的交互式编程语言一样。

目前，现有的各大流行的编程语言都没有对多核处理器进行专门的优化。而Go语言就是为了解决这一问题而诞生的。

并发编程。时至今日，并发编程已经成为了程序员的基本技能了，在各大技术社区都可以看到诸多与之相关的讨论主题。到底那种方式是最佳的并发编程体验？或许会一直争论下去。但是Go语言却一反常态，从底层就将一切都并发化了！运行时使用了Goroutine运行所有的一切，当然也包括main.main入口函数。

可以说，Goroutine已经成为了Go语言的标志性特征了。他使用类协程的方式来灵活地处理并发单元，同时却又在运行时层面做了更加深度的优化处理。这样，使得Go语言在语法上的并发编程变得极为简单！无需处理回调，无需关注执行时的切换，一切仅仅需要一个go关键字，简单而又自然！

搭配Go语言的channel，可以让新手也能轻松实现CSP并发模型。将并发单元间的数据耦合拆解开来，各司其职。这对所有纠结于内存共享、锁粒度的开发人员来说都是一个可以期盼的大解脱！如果真的要说有什么不足，那就应该是要有一个更大的计划，将通信从进程内拓展到进程之外，实现真正意义上的分布式[3]！

Goroutine和Channel使得编写高并发的服务器软件变得相当容易，很多情况下完全不需要考虑锁的机制和由此带来的一切问题。比如，单个的Go应用也能有效的利用多个CPU核心，并且执行的性能特别好。这个和Python相比有天壤之别！多线程的Python程序其实并不能有效地利用多核的优势，只能用多进程的方式来部署。如果使用标准库里面的multiprocessing包又会对监控和管理造成很多不必要的挑战。所以，部署一个python语言的时候，通常是每个CPU核心来部署一个应用，这样就会造成很多不必要的资源浪费。比如说假设某个Python应用启动之后需要占用100MB内存，而服务器有32个CPU核心，那么留下一个核心给系统，而运行31个应用副本需要浪费3GB的内存资源，太糟糕了！

1. Android Studio开发

在Android刚开始兴起的时候，还是比较贫穷的。没有官方的IDE，那个时候大家要想开发Android程序的话，需要使用Eclipse安装插件来开发。在当时看来，Eclipse的确是当时最好的Java IDE了，直到有一个叫Jetbrain的公司出现了。随着Android手机逐渐开始侵占iPhone的市场，Google也开始愈来愈重视自己的这个操作系统了。于是基于Jetbrain公司的产品，开发了Android Studio这样一个Android开发官方IDE。其实一开始的时候，Android Studio并不稳定，那个时候，所有的Android开发教程里面还是推荐使用Eclipse来开发。随着Google对Android Studio的优化技术的成熟，Android Studio越来越开始有点官方IDE的样子了。如今，在网络上已经没有人会推荐使用Eclipse来开发Android程序了。但是，中国不一样。

在中国的高校里面，大学Android相关课程里老师们依然教学生使用Eclipse来开发Android程序。教育上面的技术落后，无疑拖慢了新老技术更替的速度。让新的技术的普及速度变慢了很多。这其中的原因，一个是因为中国互联网与外国互联网技术上的语言隔阂造成的，另一个是因为中国大学大部分教师年龄太大，接受新事物的速度变得特别慢，在教授年轻的学生知识的时候，依然使用的是老旧的工具，进一步拖慢了新技术的普及速度。

一开始在实现的时候就是在Android平台实现的，等到用户量多了起来的时候，才开始扩展到其他平台。

目前Android应用程序开发的现状就是本地化要求比较高的App就使用Native开发，也就是用Java来开发；那些以内容为主的，对本地功能要求不高的，完全可以用网页替代的App（如：百度贴吧，微博等），就是用一些Web技术来开发，也就是用HTML,CSS和JavaScript来开发。对于本系统而言，JavaScript的功能自然是不足以满足我们的要求的啦，所以肯定是Native方式[4]。但是依然存在一个问题。

因为我们是需要在本地建立一个HTTP服务器的，由于HTTP服务器本身结构复杂，Java标准库里面又没有已经实现好的HTTP服务器，所以，HTTP服务器底层实现部分我选择了使用Go语言来编写。

Go语言官方团队构建了一个GoMobile库，这个库是可以帮助你将Go语言的代码运行在Android上面的。虽然不能用Go语言来写Android的UI，但是可以将Go语言代码以第三方库的形式嵌入到你的Android项目里面，然后使用Java来调用。有了这个库，我们就可以把让Go语言来写我们的HTTP服务器变成现实啦！因为Go语言本身就是为了写服务器而生的。所以使用Go语言写起来会非常顺手。

5. 桌面GUI框架

因为本系统需要做到跨平台，所以在系统实现的时候需要寻找合适的桌面GUI库。桌面GUI开发有很多选择，主要分为跨平台的GUI框架和非跨平台的GUI框架。其中我们重点关注跨平台的GUI框架。Qt。比较知名的跨平台框架，但是因为导出的安装包体积太大，所以放弃[5]。Sciter。非常轻量级的GUI框架，使用HTML来写界面，导出的安装包体积也小。UI。来自github上的一个第三方GUI框架，跨平台，原生支持，Go语言。唯一的缺点是GUI组件太少。

综合考虑之后，最终选择了Sciter框架。

## 2.2 本章小结

本章主要介绍了局域网设备控制和管理系统下所使用到的一些关键性的技术，还有一些关于这些相关技术的一些闲谈。新老技术的更替是需要时间的，但是新技术代替老技术是必然的，因为这是历史的进程，不可改变。至于我们在实际生产的时候，到底是使用最新的技术，还是使用老旧的技术，我个人认为，选择相对稳定的新技术，才是最好的平衡方法。

# 3.系统总体设计

## 3.1系统总体设计

整个系统是又两个部分组成：网络核心库、UI层。UI层包括Android UI层，Linux UI层，Windows UI层，Mac UI层。应用通过将网络核心库和UI层连接起来实现。

## 整体架构

图3-1.1 系统整体架构图

为什么要这样设计呢？由于本系统的目标是要做到跨多个平台(Windows,Linux,Android,Mac等)，为了减轻开发和维护的负担，整个系统必须最大化地将通用的部分剥离出来，尽量做到一次编写，到处运行。

因为移动端(Android)和桌面端(Windows,Linux,Mac)的UI差异太大，所以无法实现UI层的共享。但是，网络核心库在各个操作系统中是通用的，可以共享。因为本系统对于UI界面没有太大的要求，所以网络核心库是整个系统编写过程中的重中之重！在完成网络核心库之后，再将网络核心库集成到各大操作系统的UI程序里面，通过一个事件回调接口(后面会在系统初始化里面详细讲解)来实现网络核心库与UI层的通信。这样就实现了系统的跨平台。

其中，网络核心库主要分为两大模块：HTTP服务器模块、UDP发现模块

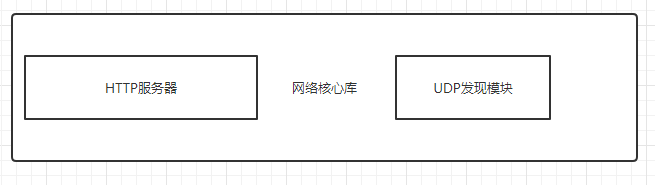


图3-1.2 网络核心库架构图

## 3.2 HTTP服务层

整个系统在启动的时候，第一步就是启动一个HTTP服务器。

这个HTTP服务器承载着文件传输、剪切板复制、摄像头直播和控制指令发送等功能。也就是说，基本上所有的模块都是基于这个HTTP服务器来传输信息和数据的。

### 3.2.1 HTTP服务器初始化

HTTP服务器启动之后第一件事情就是进行系统初始化操作。系统在初始化的时候，需要传入几个参数：事件回调接口、HTTP服务器所要监听的端口号、临时文件夹的路径还有文件接收路径。如果所传入的参数为空的话，则使用默认的值。

1. 事件回调接口(EventHandler)

事件回调接口(EventHandler)，是核心网络库向UI层通信的媒介。

为什么要设计这样一个接口？在GUI编程中，不管是桌面端，还是移动端。整个UI的操作、显示和修改，都是在同一个线程里面的，我们把这个线程叫做UI线程。在UI线程之中，是不允许进行一些耗时的操作的，比如说：Socket监听，网络请求等。否则会出现程序的图形界面卡死的情况。

所以，在本系统中，因为需要启动一个HTTP服务器，这个HTTP服务器是绝对不能在UI线程中启动的，必须要另起一个新的线程。那么问题就来了，如果这个HTTP服务器有什么状态变化的情况（比如：HTTP服务器收到一个上传过来的文件，收到控制指令等），如何去通知UI线程，并在图形界面之中显示出来呢？ 我们只能设计这样一个接口，在UI层实现接口中定义好的方法，然后当HTTP服务器出现状态变化的时候，只需要调用该接口对应的方法即可。

事件回调接口是一个接口(Interface)，定义在网络核心库之中。里面定义了一些事件的方法，但是没有实现他们。这些方法是需要由UI层来实现的。因为UI层各有不同，比如收到的消息如何显示？在Android上是显示一个AlertDialog，在Windows上则是弹出一个AlertWindow。所以这些事件回调的方法，必须是由不同的UI层自己去实现，无法共享。

事件回调接口中包含一下事件方法：

- 当收到剪切板的时候

- 当收到Web上传的文件的时候

- 当收到其他客户端上传来的文件夹的时候

- 当新的设备上线或者下线的时候

- 当收到远程控制指令的时候

- 存储键值对

- 获取键值对

为什么事件回调接口中还需要包含存储键值对、获取键值对两个方法？系统在初始化的时候需要读取配置文件，这些配置文件是以键值对的形式存储在设备之中的。然而，在不同的操作系统之中，存储配置信息的方式各有不同。比如，Windows是存储在AppData目录下的，Linux是存储在用户目录下的隐藏文件夹里面的，而在Android系统中，配置信息是存储在SharedPreferences之中的。所以，我们需要抽象出一个存储键值对方法，和一个获取键值对的方法，来让各大平台各自实现自己的存储过程即可。

(2) HTTP服务器监听的端口号

HTTP服务器要监听的端口号(Port)，是在启动HTTP服务器时的必要参数。他有自己的默认值，用户可以在设置界面里面更改端口号。但是，因为服务器一旦启动了，端口号就不能修改了，所以当用户在设置界面里面修改了端口号之后，必须重启整个系统才能使修改生效。

(3) 临时文件夹路径

临时文件夹路径(Temp Path)，该参数是为了防止Temp目录空间不足的情况出现(Linux下默认Temp目录存储空间大小为4G)，于是让用户可以自定义Temp目录位置。

(4) 文件接收路径

文件接收路径，是该参数是在启动HTTP服务器时的必要参数，他有自己的默认值，用户也可以对其进行修改。在整个系统之中，该参数是以一个共有变量的形式存储在内存之中的。也就是说，如果用户在设置界面里面修改了该参数，则不需要重启系统，即可使之生效。

## 3.3 UDP发现层

守护进程(Deamon)，是整个网络核心库的两大主要模块之一。主要的作用是进行同一局域网下的在线设备相互发现(后面会在“设备发现模型”部分详细讲解)。

### 5.3.1 守护进程初始化

守护进程初始化，是在HTTP服务器初始化之后进行的。初始化的时候，只需要一个参数即可：操作系统的标识。

操作系统的标识，主要作用不过是让用户能够快速识别其他在线设备。

### 5.3.2 设备发现模型

为什么需要“在线设备互相发现”这一功能？在同一局域网之中，当有多台设备同时开启了本系统时。用户可以进行一些更加高级的操作，比如：远程控制，向指定设备发送文件夹等。如果没有在线设备相互发现这一功能的话，用户需要自行查询每台机器的局域网IP地址，同时还要自己一个字一个字填写IP地址和端口号，况且IP地址和端口号的输入过程十分麻烦且耗时。

为了解决这一问题，为了让用户能够无需输入在线设备的IP地址，用户唯一需要做的事情就是在所有在线设备列表里面选择自己想要的设备即可。所以我们增加了UDP守护进程，主要用于同一局域网下的在线设备互相发现。

在线设备发现模型的要求

- 当设备上线的时候，要广播通知所有局域网下的已有设备

- 当设备下线的时候，所有局域网下的其他设备也要实时更新在线设备列表

- 设备需要让所以同一局域网下的其他设备知道自己的状态(如：是否开启了接收控制指令的选项)

- 当设备的状态改变时，要实时地在其他设备的在线设备列表里面更新

本系统的设备发现模型包含三个部分：广播发现、心跳连接和一个状态获取HTTP接口。

(1).广播发现

广播发现部分，是整个设备发现模型的基础，也是实现整个模型的过程的第一步。当设备启动本系统的时候，UDP守护进程初始化，同时会向所在的局域网广播一条上线提醒消息。但是，消息之中并不会包含”上线“等复杂表示，该消息仅仅只包含一个信息：那就是本机的HTTP服务器端口号。也就是说，所谓的广播发现，不过是一个向局域网下的全部设备广播一下自己的HTTP服务器端口号的简单操作罢了。

为什么只包含一个HTTP服务器端口号？

广播发现这一行为是整个设备模型的基础，但也仅仅只是基础而已，他不会包含太多的功能或者作用。在系统启动的时候，向局域网的所有设备广播自己的HTTP服务器端口号，当其他在线设备收到这一个UDP广播消息的时候，仅仅意味着：”当前局域网下的在线设备列表产生变化了，需要更新一下在线设备列表了！“。

虽然仅仅只包含一个简单的HTTP服务器端口号，但它却像一个跳板，告诉其他设备：我的状态改变了(这里的状态不仅仅包括上线、下线，还有在用户开启或关闭了”接收控制指令“选项的时候，也属于状态变化的范畴，这个时候也会向局域网下的所有设备广播自己的HTTP服务器端口号。)，快利用你们收到的HTTP服务器端口号，在我的HTTP服务器上获取我的具体状态信息。

(2).心跳连接

在设备发现模型之中，为了满足“当设备下线的时候，所有局域网下的其他设备要实时更新自己的在线设备列表”这一要求，我们必须要让每一台设备直接都建立一个心跳连接，这个心跳连接不会进行任何数据的传输，他的唯一作用就是让其他的设备知道自己是在线的。如果这个心跳连接断开的话，就说明此设备已经离线了。通过这样一种方式，就实现了“当设备下线的时候，所有局域网下的其他设备要实时更新自己的在线设备列表”。

(3).状态获取HTTP接口

每一台设备都需要设置这样的一个状态获取HTTP接口。因为HTTP协议是无状态的，这个接口的作用很简单，不管是谁来访问这个接口，都会向访问者返回本设备的实时状态信息，返回之后则立即断开连接。

每当收到在线设备列表变更的消息的时候，每台设备都会访问所有设备的状态获取HTTP接口。如果无法访问，则说明该设备已经离线，如果能访问，则将返回来的设备实时状态信息覆盖原有的设备列表中的信息。

## 3.4 文件传输模块

整个文件传输模块由两部分组成：文件部分和剪切板部分

### 3.4.1 文件部分

在文件部分的数据结构设计上，每一个文件个体包含三个信息：文件名，文件的绝对路径和文件的ID。

为什么要设计文件ID这一信息？

由于文件传输模块是建立在HTTP服务器之上的，所以，为了防止不同的文件在的URL下载地址出现冲突，URL不能以文件名作为标识，又因为文件路径包含的“/”符号与URL的分隔符冲突，所以也不能以文件路径作为文件URL的唯一表示。只能在每一个文件添加了之后，用一个随机生成的数字作为每一个分享出去的文件的唯一标识，这就是文件ID的由来。

文件部分包含四个HTTP页面或接口，分别是：

- 文件列表页面

- 文件下载接口

- 文件预览接口

- 文件上传接口

文件部分是文件传输模块的重要组成部分，其未来的发展方向还有很多，改进的方法也有很多，比如使用并行下载来提升文件传输的速度等。

### 3.4.2 剪切板部分

剪切板部分比较简单，同样是给予HTTP服务器的。主要包含两个共有变量，一个用于存储剪切板的内容，另一个用于存储剪切板开启的状态。

为什么需要剪切板部分？

剪切板是人们日常使用手机的过程中最常用到的功能之一。比如平时大家在手机上复制一段文字，复制一个网址，想要传给电脑怎么办呢？以前的办法是通过QQ或者微信发送给电脑端，然后电脑端再复制一下。这样有两个缺点：一个是电脑必须联网，另一个是电脑还必须安装QQ或者微信的客户端，如果说还有什么缺点的话，我认为应该是二次操作。所谓的二次操作是指手机复制了一次剪切板，发送到电脑上之后，电脑还得再复制一次。很不方便。所以我决定在本系统之中加入剪切板功能，就是为了方便大家在家具智能设备之间共享剪切板。

## 3.5 远程控制模块

远程控制模块包括控制者和被控制者。目前的版本，控制者只能是手机，被控制者只能是电脑。

### 3.5.1 控制者

控制者通过发送将控制指令以HTTP请求的方式发送给被控制者

### 3.5.2 被控制者

被控制者通过在HTTP服务器上新增一个接口，专门用于接收HTTP控制指令。当收到HTTP控制指令的时候，再又UI层去解析并执行控制指令

### 3.5.3 权限问题

被控制者默认是不接受任何控制指令的，这个时候如果有控制者想发送控制指令过来，会返回无效信息。只有当被控制者开启受控端选项的时候，才会接收控制指令。在“受控端”开启的时候，有两种选项：第一种是接收任何人的控制指令，第二种是接收指定设备的控制指令，第三种是接收除了指定设备以外的所有控制指令。

### 3.5.4 申请控制权限的过程

如果控制者是在受控端的可控设备列表之外的话，需要先向受控端申请控制权限，受控端同意了之后，才能进行远程控制。

## 3.6 摄像头直播模块

摄像头直播模块是基于HTTP服务器的，主要由广播，WebSocket长连接，直播者页面和观看者页面组成。

### 3.6.1 .WebSocket长连接

WebSocket长连接是将所有观看者和直播者通过HTTP服务器连接起来的媒介。之所以选择WebSocket是因为这是目前最新的Web技术————HTML5推荐的保持长连接的方法。摄像头直播时的广播就是依靠这一长连接来实现的。

### 3.6.2 广播

广播是在HTTP服务器收到直播者在Web浏览器上发过来的摄像头数据的时候，将这一数据由服务器向所有在线的观看者发送过去的行为。

没当有新的用户连接到摄像头直播页面，就会与服务器建立一个WebSocket连接，同时，将该连接加入所有已连接的数组里面。这样一来，当直播者需要广播数据的时候，只需要遍历一下这个数组，然后逐一发送即可。

### 3.6.3 直播者页面和观看者页面

直播者页面和观看者页面是由HTML5技术实现的，通过调用HTML5浏览器接口，获取摄像头数据，然后通过WebSocket发送给服务器，在观看者收到摄像头数据之后，利用Canvas画布显示出来即可。

## 3.7本章小结

在本系统中设计的这个设备发现模型，在设计模式上，完美的实现了分而治之的思想。将设备实时状态更新这一功能按照类型分给了三个部分：广播发现，心跳连接和状态获取HTTP接口。他们分别代表了三个角色：通知者，监视者和展示者。每一个角色自身的功能极其简单，但是三个角色合作起来，却又能正好完成一项复杂的工作。这也是我个人在Go语言的学习过程之中领悟到的：解耦。这样一来，就不会像其他的设备发现模型那样，出现无限的广播反弹的现象，而且又简单。正所谓：把一件简单的事情做复杂很容易，但是能把一件复杂的事情简化的人才是大师。

# 4.开发与实现

## 4.1 开发环境

表 4-1

|  |  |  |
| --- | --- | --- |
| 编号 | 名称 | 开发或运行环境 |
| 1 | 客户端 | Android,Windows,Linux,Mac |
| 2 | 开发工具 | Android Studio,Sublime Text 3,Go |
| 3 | 硬件设备 | 小米5A一台，小米笔记本Air一台 |

## 4.2 HTTP服务器

理论上来说，任何语言都可以用来写HTTP服务器，但是不同的语言都有自己的优劣之分。选择什么样的语言来写HTTP服务器，咱们先看一看目前各大互联网企业的服务器都是用什么语言写的。

Java是目前服务端使用最多的语言，对于一些大型的网站，云计算，服务器都是用Java写的。但是，缺点是太过庞大，因为我们其实做的是一个面向普通用户的软件，软件的体积必须要尽可能的小，如果你写一个服务器，要运行的话还要用户再去安装一个JRE，那用户体验肯定就太差了。再者，Java本身没有将HTTP协议写进标准库里面去，也就是说，你需要使用第三方的HTTP框架来完成，第三方框架最出名的也就是Spring框架了，这个框架也是太过庞大了，之适用于大型网站，不够轻巧。

PHP、Python、C#也都可以写服务器，但是，因为我们需要全平台支持，如果不能再手机端运行的话，就不能使用这个语言。

Go语言是我们最合适的选择，他支持全平台，简单，标准库内置HTTP协议实现，不需要第三方HTTP框架。同时运行速度还非常快[6]。

### 捕获1

图 4-1 Web文件传输页面

用户在Web文件传输界面可以点击Download下载别人分享出来的文件，也可以直接点击文件名预览图片等文件，同时下面也可以上传本机文件到手机上。

核心代码为

mf,e:=os.OpenFile(MyStoragePath+path+v.Filename, os.O\_WRONLY|os.O\_CREATE|os.O\_TRUNC, 0644)

defer mf.Close()

io.Copy(mf, file)

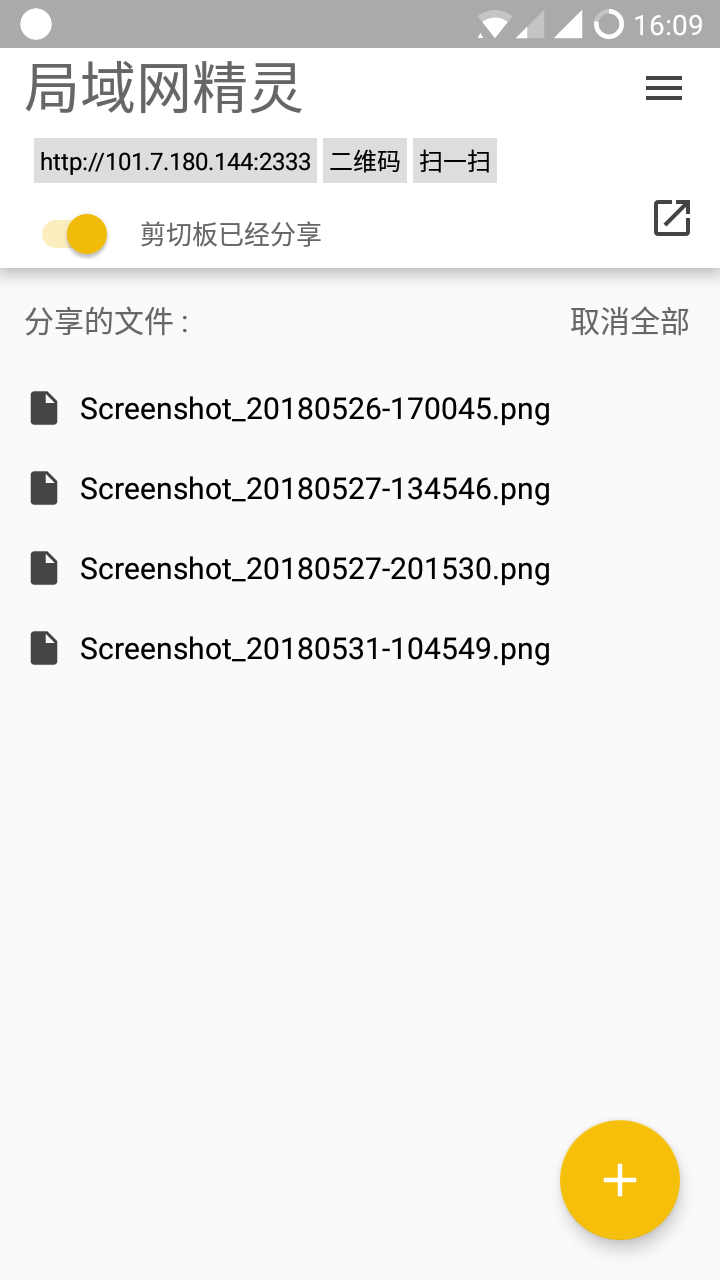


图 4-2 App文件分享界面

## 4.3 键盘控制

为了实现键盘和鼠标的控制，我们找到了一个第三方库：

https://github.com/go-vgo/robotgo

这个第三方库实现了跨平台的键盘鼠标输入控制，也就是说，可以在Go语言代码里面控制键盘和鼠标的输入。比如执行指令"enter"，就相当于按了键盘上的回车按键。另外一个，关于组合按键，他是这样解决的。例如：输入"control alt t"这样的命令，以空格来隔开，就可以实现组合按键的功能。

本系统在实现的时候也是原封不动的使用了该第三方库的指令系统，在控制端输入指令之后，通过HTTP协议发送给受控端，然后受控端执行指令即可。

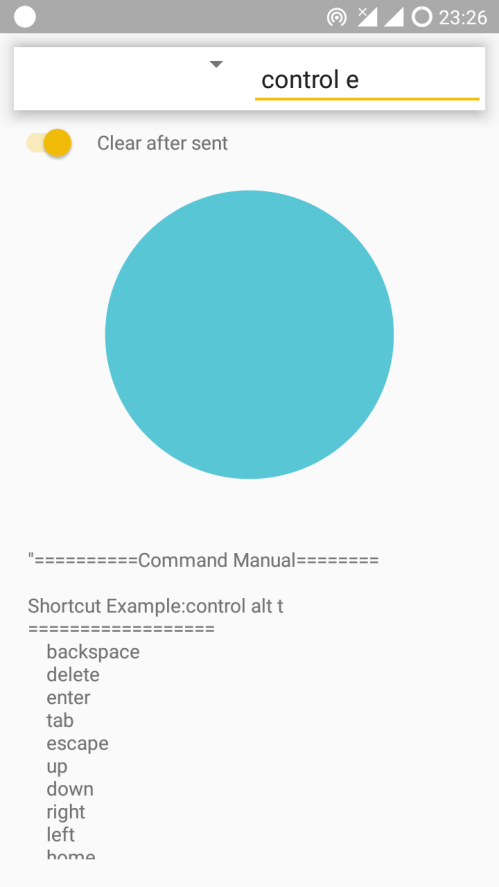


图 4-3 App键盘控制中心界面

键盘控制指令的核心代码为

b, e := ioutil.ReadAll(r.Body)

if e != nil {

fmt.Println("readAll() failed:", e)

return

}

MyEventHandler.OnRemoteControlCmdReceived(string(b))

## 4.4 摄像头直播

摄像头直播功能主要是参考了网上的一个第三方摄像头直播示例代码，然后有了灵感。

为什么在Web端实现？

整个摄像头直播功能主要是在Web端实现的，也就是说需要借助浏览器来调用系统的摄像头接口。这和大家以往想象的摄像头直播软件不同。之所以这么实现，一个是因为Go语言本身就有很多这种类型的例子；其二是因为在Web端实现的话，另一台设备不需要安装客户端了，只要有一个浏览器即可实现直播者或者观看者的角色。我觉得这样会方便很多。

首先直播者在Web端点击一个按钮，JavaScript便通过WebSocket连接到HTTP服务器，然后用JavaScript通过HTML5的摄像头API————navigator.getUserMedia()函数，来申请摄像头画面数据。拿到摄像头画面数据之后呢，把这些数据通过刚才连接上的WebSocket发送给服务器，服务器收到数据之后，则对所有的WebSocket连接进行广播。至于观看者这边，同样，也是用WebSocket与服务器进行连接，然后监听数据。如果服务器发送数据过来了，观看者这边再用JavaScript把画面显示在HTML中的Img标签里面即可。

摄像头直播的关键代码是：

func sendToAll(str string) {

for k, v := range members {

if e := websocket.Message.Send(v, str); e != nil {

members = append(members[:k], members[k+1:]...)

break

}

}

}

Web端的实现代码为：

<video id="sourcevid" controls></video>

<canvas id="output" style="display:none"></canvas>

<script>

var data,v=

document.getElementById("sourcevid"),

mcavas=document.getElementById("output"),

mcavasContext=mcavas.getContext("2d")

</script>

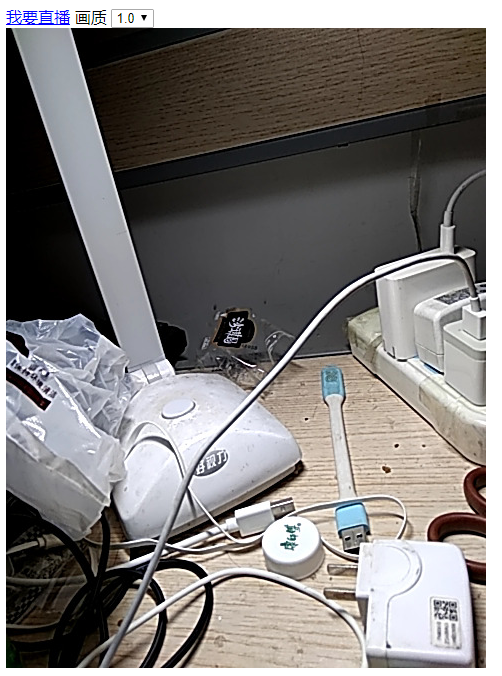


图 4-2 App文件分享界面

# 5.总结

## 5.1遇到的问题

### 5.1.1Android临时文件夹问题

在用户上传文件到手机上的时候，因为大文件在接收的时候肯定是不能存在内存里的，我们是需要做一个本地缓存文件，然后对接收到的数据进行分段合并。这个本地缓存文件在Linux上本来是存储在临时文件夹的。但是在最近的几个Android版本出来了之后，Android在操作系统级别上去掉了原来的临时文件夹功能。如果有程序试图使用临时文件夹来存储分段的缓存文件，就会出现permission denied的错误。

经过我的仔细调查之后，找到了解决办法。那就是修改Go语言运行时的临时文件夹环境变量，这样，Go语言运行时的临时文件夹就会修改成当前程序有权限的文件夹。

## 5.2感想与展望

做软件的开发是一件非常有意思的事情。因为你在编程的时候，其实是一种创造的过程，把一个东西从无到有，这个过程中每个人都是造物主。在我第一次接触编程之后，我发现自己的眼界变得开阔了，不仅仅局限在编程过程中有这种感受。在生活中，我也会习惯性地开始以创造者的角度来思考问题，在这个思考的过程中，我学到了很多很多。

关于本文所讲的系统，我个人认为以后还是会有很多提升的空间的。比如在系统上面加入P2P功能，让用户的文件传输不仅仅局限在局域网里面，让所有的用户可以在公网上进行文件传输。当然这实现起来可能会有很多困难，但是这不妨碍这个功能成为吸引人的特性。

# 参考文献

[1]祝翔,董启文,郁可人.基于WebSocket的PK答题的设计与实现[J].华东师范大学学报(自然科学版),2018(02):89-100.

[2]林路智.Go语言搭建网站解析[J].电脑知识与技术,2015,11(29):60-61.

[3].Google Go语言将加速对Android平台的支持[J].电脑编程技巧与维护,2010(17):4.

[4]何兴鹏,刘钊远,陶琛嵘.Linux程序向Android平台移植的研究[J].计算机测量与控制,2018(05):112-115.

[5]江存.基于Linux的2种HTTP服务器实现与对比分析[J].现代计算机(专业版),2017(24):58-61+76.

[6]王伯槐,张烨.基于Go语言的消息推送平台的设计与实现[J].数码设计,2017,6(02):33-36.

# 致谢

本次毕业设计，首先要感谢我的指导老师李爱萍，我的毕业设计是在李老师的指导下完成的，老师给予了我很大的帮助。并为具体开发以及论文的书写提出了许多宝贵意见，他严谨仔细的治学态度深深的影响了我，在此对李老师的帮助表示衷心的感谢！

在这里还要特别感谢和我一起工作学习的同学，通过跟他们的讨论也解决了我在设计过程中的一些疑惑，增长了我的见识，拓宽了我的视野，同时也让我发现了自身的不足，谢谢你们给予我的帮助。

最后我要感谢我的母校太原理工大学，在这里我度过了令我终身难忘的四年，四年中，我收获了很多，不仅收获了知识，还收获了一帮志同道合的师长和朋友，感谢他们一直以来的支持和鼓励。