|  |  |
| --- | --- |
| 分类号:TP39 |  |
| 10710-2010124055 |



硕 士 学 位 论 文

基于 Android 平台的智能家居客户端系统设计与实现

崔 荔

|  |  |  |  |
| --- | --- | --- | --- |
| 导师姓名职称 |  | 张卫钢 教授 |  |
| 申请学位级别 | 硕 士 | 学科专业名称 | 交通信息工程及控制 |
| 论文提交日期 | 2013 年 5 月 20 日 | 论文答辩日期 | 2013 年 6 月 2 日 |
| 学位授予单位 |  | 长安大学 |  |

**Design and implementation based on the Android platform of smart home client system**

A Dissertation Submitted for the Degree of Master

**Candidate：Cui Li Supervisor：Prof. Zhang Weigang**

Chang’an University, Xi’an, China



摘 要

智能家居在国内经过数十年的发展，逐渐走向普通百姓家庭，它利用先进的计算机技术、网络通讯技术、总线技术，将与家居生活息息相关的安防、灯光控制、窗帘控制等有机的结合在一起，通过智能终端统一进行控制和管理。“一键式控制”的设计理念使得人们的生活更加智能、方便、快捷。

Android平台作为新一代操作系统的代表，凭借特有的开放性平台优势，使其已成为当今最重要的移动终端设备系统之一，基于该平台开发的互联网应用更是快速增长。

Android系统的迅速发展和技术的进一步成熟，为智能家居客户端的选型带来崭新的思路及广阔的应用前景。因此，本论文提出了一种基于Android平台的智能家居客户端系统的解决方案。

首先，研究分析Android平台以及软件开发中所涉及到的XML解析技术、网络通信技术、存储技术、多线程模型和Android系统的界面布局技术。

其次，根据智能家居系统的整体框架结构，提出一种创新的内网、外网相结合的控制方式，综合软件的需求分析和业务流程对系统的功能模块进行划分，通过模块划分图宏观说明各个模块之间的关系，然后对每个模块分别进行实现，并截图展示最终的实现效果。

最后，对软件的功能和性能分别进行测试。测试结果表明，基于Android平台的智能家居客户端系统运行正确、稳定，界面简洁美观，达到了预期的实现效果，具有较高的实用和参考价值。

**关键词：**计算机应用； 智能家居； 客户端； Android； 多线程

**Abstract**

Smart home in our country after decades of development, gradually moving towards the family of ordinary people, it uses advanced computer technology, network communication technology, bus technology, combination of family-related security, lighting control, curtain control, unified control and management through intelligent terminal." One-click" design concept makes people's lives more intelligent, convenient and fast.

Android platform, as the representative of the new generation operating system, with characteristic of open platform advantage, make it has become one of the most important mobile devices today, Internet applications based on the platform development is the rapid growth. The rapid development of the Android system and the technology to mature, bring new ideas and broad application prospects for the selection of the smart home client. Therefore, this design proposes a smart home based on Android platform client system solutions.

First of all, the thesis research and analysis technology that mentioned in the Android platform and software development, such as XML parsing technology, network communication technology, storage technology, multi-thread model and interface layout of the Android system.

Secondly, according to the overall frame structure of smart home system, puts forward an innovative combination of internal network and external network control method, and combined with the software requirements analysis and business process function to divide the modules of the system, macroscopic description of the relationship between each module by module partition map, then to implement of each module, and through the screenshots show the display effect.

Finally, the system function and performance was tested respectively. Test results show that based on the Android platform of smart home client system ran properly and steadily, the realization of the interface is simple and beautiful, achieved the desired effect, and has high practical and reference value.

**Key words:** Computer application; Smart home; Client; Android; Multithreading

目 录

[摘 要](#_Toc686410331) 2

**[Abstract](#_Toc686410332)** 3

[第一章 绪论](#_Toc686410333) 6

[1.1 课题背景及选题意义](#_Toc686410334) 6

[1.2 国内外研究现状](#_Toc686410335) 7

[1.3 客户端选型方案](#_Toc686410336) 7

[1.4 论文的主要工作](#_Toc686410337) 8

[第二章 相关技术研究](#_Toc686410338) 9

[2.1 Android系统](#_Toc686410339) 9

[2.1.1 Android基本架构](#_Toc686410340) 9

[2.1.2 Android开发环境搭建](#_Toc686410341) 10

[2.2 XML技术](#_Toc686410342) 10

[2.2.1 XML文档结构](#_Toc686410343) 10

[2.2.2 XML解析技术](#_Toc686410344) 11

[2.3 通信技术](#_Toc686410345) 12

[2.3.1 Http通信](#_Toc686410346) 12

[2.3.2 Socket通信](#_Toc686410347) 12

[2.4 存储技术](#_Toc686410348) 14

[2.4.1 SharedPreferences存储方式](#_Toc686410349) 14

[2.4.2 文件存储方式](#_Toc686410350) 14

[2.4.3 SQLite数据库存储方式](#_Toc686410351) 14

[2.5 多线程模型](#_Toc686410352) 15

[2.5.1 模型优势](#_Toc686410353) 15

[2.5.2 消息机制](#_Toc686410354) 15

[2.6 编码转换机制](#_Toc686410355) 16

[2.7 布局技术](#_Toc686410356) 16

[2.8 本章小结](#_Toc686410357) 17

[第三章 客户端软件设计](#_Toc686410358) 17

[3.1 智能家居体系结构](#_Toc686410359) 17

[3.1.1 总体框架](#_Toc686410360) 17

[3.1.2 控制方式](#_Toc686410361) 18

[3.2 需求分析](#_Toc686410362) 19

[3.2.1 功能性需求](#_Toc686410363) 19

[3.2.2 非功能性需求](#_Toc686410364) 19

[3.2.3 部分功能需求详细描述](#_Toc686410365) 20

[3.3 业务流程](#_Toc686410366) 23

[3.3.1 程序初始化流程](#_Toc686410367) 23

[3.3.2 场景模式设置流程](#_Toc686410368) 23

[3.3.3 房间布局设置流程](#_Toc686410369) 23

[3.3.4 设备位置定义流程](#_Toc686410370) 23

[3.3.5 设备控制操作流程](#_Toc686410371) 24

[3.4 功能模块划分](#_Toc686410372) 24

[3.4.1 用户登录](#_Toc686410373) 25

[3.4.2 软件更新](#_Toc686410374) 25

[3.4.3 网络通信](#_Toc686410375) 25

[3.4.4 界面操作](#_Toc686410376) 26

[3.4.5 数据存储](#_Toc686410377) 26

[3.5 通信协议设计](#_Toc686410378) 26

[3.6 本章小结](#_Toc686410379) 27

[第四章 客户端软件实现](#_Toc686410380) 27

[4.1 用户登录模块](#_Toc686410381) 27

[4.1.1 类关系图](#_Toc686410382) 27

[4.1.2 界面实现](#_Toc686410383) 28

[4.1.3 快捷登录](#_Toc686410384) 28

[4.2 软件更新模块](#_Toc686410385) 28

[4.2.1 类关系图](#_Toc686410386) 28

[4.2.2 Web服务端](#_Toc686410387) 28

[4.2.3 客户端XML文档解析](#_Toc686410388) 29

[4.3 网络通信模块](#_Toc686410389) 29

[4.3.1 类关系图](#_Toc686410390) 29

[4.3.2 通信接口](#_Toc686410391) 29

[4.4 界面操作模块](#_Toc686410392) 30

[4.4.1 类关系图](#_Toc686410393) 30

[4.4.2 界面实现](#_Toc686410394) 30

[4.5 数据存储模块](#_Toc686410395) 31

[4.5.1 类关系图](#_Toc686410396) 31

[4.5.2 表的基本结构](#_Toc686410397) 31

[4.6 本章小结](#_Toc686410398) 35

[第五章 客户端软件测试](#_Toc686410399) 35

[5.1 测试环境](#_Toc686410400) 35

[5.2 测试方法及结果分析](#_Toc686410401) 36

[5.2.1 功能测试](#_Toc686410402) 36

[5.2.2 性能测试](#_Toc686410403) 40

[5.3 所示。](#_Toc686410404) 41

[5.3 本章小结](#_Toc686410405) 41

[第六章 总结与展望](#_Toc686410406) 41

[6.1 总结](#_Toc686410407) 41

[6.2 展望](#_Toc686410408) 41

[参考文献](#_Toc686410409) 41

表格目录

表**1.1** **IDC 2012**年**3**季度发布的移动设备占有率调查数据 7

表**3.1** 各模块的基本功能 24

表**3.2** 设备状态控制指令协议 26

表**4.1** 用户信息表（**USER**） 31

表**4.2** **IP**地址表（**ADDRESS\_IP**） 32

表**4.3** 设备表（**DEVICES**） 33

表**4.4** 房间表（**ROOMS**） 34

表**5.1** 测试所需的软件环境 35

表**5.2** 测试所需的硬件环境 35

表**5.3** 登录模块主要测试用例 36

表**5.4** 软件更新模块主要测试用例 37

表**5.5** 网络通信模块主要测试用例 38

表**5.6** 界面操作模块主要测试用例 38

表**5.7** 数据存储模块主要测试用例 39

|  |  |  |
| --- | --- | --- |
|  | [3.1.2 控制方式 ...................................................](#_bookmark28) | [26](#_bookmark28) |
| [3.2](#_bookmark29) | [需求分析.........................................................](#_bookmark29) | [27](#_bookmark29) |
|  | [3.2.1 功能性需求 .................................................](#_bookmark30) | [27](#_bookmark30) |
|  | [3.2.2 非功能性需求 ...............................................](#_bookmark31) | [29](#_bookmark31) |
|  | [3.2.3 部分功能需求详细描述 .......................................](#_bookmark32) | [30](#_bookmark32) |
| [3.3](#_bookmark33) | [业务流程.........................................................](#_bookmark33) | [33](#_bookmark33) |
|  | [3.3.1 程序初始化流程 .............................................](#_bookmark34) | [33](#_bookmark34) |
|  | [3.3.2 场景模式设置流程 ...........................................](#_bookmark35) | [34](#_bookmark35) |
|  | [3.3.3 房间布局设置流程 ...........................................](#_bookmark36) | [34](#_bookmark36) |
|  | [3.3.4 设备位置定义流程 ...........................................](#_bookmark37) | [35](#_bookmark37) |
|  | [3.3.5 设备控制操作流程 ...........................................](#_bookmark38) | [35](#_bookmark38) |
| [3.4](#_bookmark39) | [功能模块划分.....................................................](#_bookmark39) | [36](#_bookmark39) |
|  | [3.4.1 用户登录 ...................................................](#_bookmark40) | [37](#_bookmark40) |
|  | [3.4.2 软件更新 ...................................................](#_bookmark41) | [38](#_bookmark41) |
|  | [3.4.3 网络通信 ...................................................](#_bookmark42) | [39](#_bookmark42) |
|  | [3.4.4 界面操作 ...................................................](#_bookmark43) | [40](#_bookmark43) |
|  | [3.4.5 数据存储 ...................................................](#_bookmark44) | [41](#_bookmark44) |
| [3.5](#_bookmark45) | [通信协议设计.....................................................](#_bookmark45) | [42](#_bookmark45) |
| [3.6](#_bookmark46) | [本章小结.........................................................](#_bookmark46) | [44](#_bookmark46) |
| [**第四章**](#_bookmark47) | [**客户端软件实现 ..................................................**](#_bookmark47) | [**45**](#_bookmark47) |
| [4.1](#_bookmark48) | [用户登录模块.....................................................](#_bookmark48) | [45](#_bookmark48) |
|  | [4.1.1 类关系图 ...................................................](#_bookmark49) | [45](#_bookmark49) |
|  | [4.1.2 界面实现 ...................................................](#_bookmark50) | [46](#_bookmark50) |
|  | [4.1.3 快捷登录 ...................................................](#_bookmark51) | [48](#_bookmark51) |
| [4.2](#_bookmark52) | [软件更新模块.....................................................](#_bookmark52) | [49](#_bookmark52) |
|  | [4.2.1 类关系图 ...................................................](#_bookmark53) | [49](#_bookmark53) |
|  | [4.2.2 Web 服务端..................................................](#_bookmark54) | [50](#_bookmark54) |
|  | [4.2.3 客户端 XML 文档解析 .........................................](#_bookmark55) | [50](#_bookmark55) |
| [4.3](#_bookmark56) | [网络通信模块.....................................................](#_bookmark56) | [54](#_bookmark56) |
|  | [4.3.1 类关系图 ...................................................](#_bookmark57) | [54](#_bookmark57) |

# 第一章 绪论

## 1.1 课题背景及选题意义

近年来，在电视、报纸、互联网等新闻媒体上，人们经常可以看到“智能家居”这一新的、时尚科技词汇。所谓的智能家居，是以住宅为平台，利用计算机技术、有线或无线组网技术、网络通信技术，融合用户需求，集与家居生活有关的安防、灯光、窗帘、电器等设备的控制于一体，通过网络进行智能管理，达到“以人为本”的全新家居生活体验[1]。智能家居相比传统家居而言，不仅具有传统的居住功能，更可以提供舒适、安全、宜人的家庭生活条件，而且利用全方位的信息交互功能，可以帮助家庭和外部信息保持及时有效的沟通，这样不仅可以优化人们的生活方式，帮助人们更加合理的安排时间，而且能有效提高家庭生活的安全性。

智能家居近几年来一直是社会的热门话题之一，已经有越来越多的机构和个人开始了对智能家居的研究。因为智能化和远程控制是智能家居的两大特征，所以，智能家居实现的目标就是无论用户距离家庭的距离远近，只要身处有网络的地方，就可以通过网络通信技术实现对家居设备的智能控制，使其按照用户设定的方式工作[2]。

智能家居能为生活带来许多便利，可谓“革命性的变化”。首先，安全更有保障，家庭的门和窗户都配备了门磁报警器，因此外出时不必担心家里有人闯入行窃，只要有人闯入就会发出报警信号传输到用户的智能终端上，同时视频监控也能抓拍照片。当室内发生煤气泄漏或是火情时也有感应器发出信号到终端上提醒用户。另外，使用智能家居系统也能使我们的家庭生活变得更轻松，在用户准备出门前，将系统设置为“离家模式”，系统会自动将家庭中所有的电灯和家用电器关闭；在用户回家前，也可以通过客户端控制家居设备，提前让家里的电器开始工作，比如设置一个舒适的空调温度、自动打开窗户等[3]。

智能家居是一项系统性的工程，它涉及到很多现代技术和人们生活中的很多方面。智能家居控制系统的最终目标是一种理念，若要想得到很好的发展，它的研发机构就必须本着便利、实用、性价比高、适合市场的发展战略，悉心研究人们的生活习惯和精神文化需求，并把这些需求当成最终目标，想方设法运用各种技术手段去实现它[4]。随着科技的发展，智能家居系统的发展也已进入了一个数字化、网络化的阶段。用户通过互联网功能，就可以在世界的任何一个地方控制家中的设备。随着手机和3G网络的普及，

用户不需要亲自在家操作，便可以通过外观小巧、方便携带的智能手机在任何时间、地点对家中所有的家居设备完成远程控制和集中管理，将原本一座很普通的住宅转变为集安全、便捷、舒适以及低碳环保为一体的高智能住宅，继而给人们的生活带来前所未有的便捷[5]。

可以想象一下，在不远的将来，智能家居系统将会广泛的应用于千家万户。在家时，仅仅通过一个小小的智能终端，就能控制家中的电器；早晨起床时，通过预先的时间设定，卧室的窗帘自动拉开，厨房的电饭煲开始自动工作；下班回家前，通过远程联网控制，房间的空调系统打开，使室内达到舒适的温度[6]。这一切的家居智能化，都能使人们的生活更加的舒适和便捷。

## 1.2 国内外研究现状

智能家居这一概念在国外发展起步较早，早在1984年美国康乃迪克州就出现了世界上第一幢智能化建筑City Place Building，由于当时的科技并不是很发达，所谓的智能化也只是在大楼改造时加入了计算机系统的应用，通过该系统可以集中监控大楼内的空调和电梯等设备，并提供一些语音、邮件信息之类的交互式通信服务[7]。自此，许多发达国家已陆陆续续对有关智能家居的解决方案进行研究，到目前为止，美国、德国、新加坡等发达国家已较广泛地应用智能家居，并在这个领域的研究已经有了不小的成果。目前，智能家居行业在国际上比较具有代表性的产品有：

(1)美国的X-10系统。它主要依托美国电力系统网络实施开发，继而对家用电器设备、照明设备以及安全防护设备实现集中控制。其销售已超过1亿个，仅在美国便有超过400万个家庭在使用，设计户型多为旧房改造的单体别墅。但由于该系统是依照美国本土的电力条件以及西方人的日常消费习惯设计，所以具有较强针对性，因而很难满足国内消费群体的要求[8]。

(2)德国的EIB系统。为较好实现住宅家居设备智能化，强化其控制功能，采用预埋总线以及中央控制的方式。这种系统当前在国内的住宅小区采用较少，主要原因在于其工程标准要求严格，实施过程较为复杂，且需要预埋线路，所以此系统仅适用于新规划建设的住宅小区[9]。

我国的智能家居产业是从20世纪90年代中后期如雨后春笋般兴起。1999年，建设部对全国住宅小区全面实施智能化技术示范工程，这对智能化住宅小区迈入快速发展阶段具有里程碑意义[10]。2000年，由海尔集团提出的“e家庭”的概念，与微软公司的

windowMe技术以及其他网络家电技术相结合。其中，中央控制器选用海尔电脑，无线通讯数字控制中心亦选用海尔移动电话，同时以其他网络家电作为终端设备，从而成为了具有真正意义上中国家居智能化的典范[11]。然而，由于我国智能家居发展历史较为短暂，所以至今还没有形成较为全面、统一的技术标准和产业标准，也未能实现大规模智能家居的应用。

智能家居系统在我国作为一个新的产业，目前正处于一个导入期与成长期的交叉点，虽然市场消费观念还未形成，但随着智能家居市场进一步的推广普及，培养消费者的使用习惯，那么智能家居市场的消费潜力必然是巨大的，产业前景光明。国内很多厂商都加入了研发和生产智能家居产品的行业中，部分厂商取得了不错的成果，比如，清华同方的e-Home数字家园、海尔的U-Home家电、海信D Net-Home数字家庭等[12]。同时，国家也通过颁布相关的政策法规来促进智能家居产业的发展，将研究通信网

络技术和相关产品的开发列为重点发展方向，并通过各种媒体渠道进行广泛宣传。迅速普及的网络技术和智能化小区对智能家居的发展起到了积极的推动作用。现在已有一部分居民接受了智能家居这个概念，很多小区的开发商也都或多或少考虑加入智能化设施，并将其作为一种亮点进行宣传推广[13]。由此看来，智能家居正在从各个方面融入人们的日常生活。

## 1.3 客户端选型方案

随着3G通信网络的广泛应用，移动设备这一概念所包含的设备品种越来越多，现在市场上广泛流行的便携移动设备如智能手机、掌上电脑（PDA）、平板电脑等都具有无线通信的功能，具备了实现移动智能家居系统客户端的硬件需求[14]。因此，本论文的智能家居系统客户端的设计范围并没有局限在手机终端的范围，而是定义在了一个更加广泛的便携移动设备范畴中。

目前，移动设备终端正处于一个高速发展的阶段，用户对移动设备的功能要求也越来越高，因此，对于移动设备的应用软件设计和开发就显得十分重要。厂商所开发的应用软件是否能够满足用户的功能需求，业务流程是否合理，软件的界面是否简洁美观，这些都是决定一款应用软件会不会被大众所能够接受的关键因素。

基于移动设备终端设计相关的第三方应用软件，首先需要考虑的是移动设备的操作系统在市场上的使用情况，当前流行较为广泛的移动操作系统主要有美国苹果公司

（Apple）的iOS系统、美国谷歌公司（Google）的Android系统、美国微软公司（Microsoft）

的Windows phone系统、加拿大Research In Motion（RIM）公司的Blackberry系统以及芬兰诺基亚公司（Nokia）的Symbian系统[15]。在2008年之前，Symbian系统占据了智能手机市场的大部分份额，但随后Google公司和Apple公司推出Android系统和iOS系统后，Symbian系统的市场份额迅速被这两种系统瓜分。

Android系统最初由Andy Rubin等人创建的Android公司开发，当时主要应用于手机领域。Google公司于2005年低调收购了成立仅22个月的高科技企业Android及其团队，并且宣布建立一个全球性的联盟组织，该组织由34家手机制造商、软件开发商、

电信运营商以及芯片制造商共同组成，并与84家硬件制造商、软件开发商及电信营运商组成[开放手机联盟](http://baike.baidu.com/view/6576231.htm)（Open Handset Alliance）来共同研发改良Android系统[16]。这一联盟将支持Google发布的手机操作系统以及应用软件，Google以Apache免费开源许可证的授权方式，发布了Android的源代码。从那时起，Android系统便开始扮演颠覆移动设备操作系统领域的角色。

根据国际数据公司International Data Corporation（IDC）统计，2012年的第三季度，

Google公司的Android设备在全球范围内的销量是1.36亿台，拿下全球75%的市场份额，这意味着全球1.81亿台智能手机中有四分之三运行的是Android系统[17]。统计数据见表1.1所示。Android设备持有率的迅速攀升，证明了Android系统成为了移动设备操作系统领域中毋庸置疑的领军者。

**表1.1** **IDC 2012年3季度发布的移动设备占有率调查数据**

| Operating System | 3Q12  Shipment Volumes | 3Q12  Market Share | 3Q11  Shipment Volumes | 3Q11  Market Share | Year-Over-Year Change |
| --- | --- | --- | --- | --- | --- |
| Android | 136.0 | 75.0% | 71.0 | 57.5% | 91.5% |
| iOS | 26.9 | 14.9% | 17.1 | 13.8% | 57.3% |
| BlackBerry | 7.7 | 4.3% | 11.8 | 9.5% | -34.7% |
| Symbian | 4.1 | 2.3% | 18.1 | 14.6% | -77.3% |
| Windows Phone | 3.6 | 2.0% | 1.5 | 1.2% | 140.0% |
| Linux | 2.8 | 1.5% | 4.2 | 3.4% | -31.7% |
| **Totals** | 181.1 | 100% | 123.7 | 100% | 46.4% |

Android平台的免费性和开放源码等优势使越来越多的开发者愿意加入到其相关应用开发的行列之中。Android系统的这些优势，使开发者能够充分发挥自己的想象空间，并且可以利用Google发布的源码来更新和改进自己设计的程序，使其更加符合开发者的设计思路和表现形式。除了Android系统已占据大部分市场份额以外，国内的许多IT企业还基于Android系统的内核和源码深度定制开发各具特色的新型操作系统，Android

系统的应用范围越来越大，适用的终端也将越来越多，具有广阔的应用领域和良好的发展前景。这正是本论文选择Android平台作为智能家居客户端系统的关键因素。

## 1.4 论文的主要工作

本论文主要实现了Android系统和移动互联网技术在智能家居系统中的应用，结合具体科研项目，采用JAVA作为开发语言，Eclipse、Android SDK作为开发工具，设计开发了智能家居系统的客户端应用程序。论文从系统的需求分析出发，逐步阐述了软件的设计过程及思路，并给出具体的功能实现方法，对系统的软件结构及设计模式进行了研究，最后给出软件系统的测试过程及结果。

论文具体的研究内容如下：

1. 通过查阅相关资料，熟悉Android平台的结构体系、开发模式及环境变量的配置。掌握XML文档的结构及其相关的解析技术。

2. 分析研究智能家居客户端系统中涉及到的关键技术，掌握Android中的网络通信方式，及多线程编程的方法。

3. 分析客户端软件的功能性需求和非功能性需求，然后根据功能需求提出具体的设计方案。

4. 分析设计智能家居系统客户端软件的用户界面，要求应尽量简洁美观、易操作。

5. 使用Android系统自带的SQLite数据库设计相应的数据表，实现对数据的访问和存储。

6. 利用Google提供的开发包和帮助文档进行第三方应用程序的开发，实现相应的功能，并在Android设备上进行测试。

# 第二章 相关技术研究

## 2.1 Android系统

Android一词的本义为“机器人”，是Google公司于2007年11月5日宣布推出的智能操作系统的名称，该系统基于Linux平台并且完全开源，主要适用于移动设备，如智能手机和平板电脑。

Android系统在正式推向市场之前，拥有两个以机器人名称来对其命名的内部测试版本，它们分别是阿童木（Android Beta）和发条机器人（Android 1.0）。后来由于涉及到版权问题，Google从Android 1.5发布的时候将其命名方式改为甜点命名法[18]。甜点的尺寸越来越大代表了版本的更新顺序，然后按照26个英文字母进行排序：Cupcake纸杯蛋糕（Android 1.5），Donut甜甜圈（Android 1.6），Eclair松饼（Android 2.0/2.1），

Froyo冻酸奶（Android 2.2），Gingerbread姜饼（Android 2.3），Honeycomb蜂巢（Android 3.0），Ice Cream Sandwich冰激凌三明治（Android 4.0），Jelly Bean果冻豆（Android

4.1/4.2)。

Google Play前名为Android Market，是一个由Google为Android设备开发的在线应用程序商店，该应用程序会预装在用户的Android设备中，用于让用户去浏览、下载以及购买在Google Play上的第三方应用程序。截止至2012年1月6日，Google公司的Google Play已有10万开发者推出超过40万的活跃应用，并且大多数的应用程序都是免费的。

### 2.1.1 Android基本架构

Android 系统的架构自上而下分为应用程序（Applications）、应用程序框架

（Application Framework）、系统运行库（Libraries）、系统内核（Linux Kernel）四个部分，如图2.1所示[19]。



**图2.1** **Android系统架构图**

#### 1. 应用程序（Applications）

Android平台的版本发布时会带有如短信、联系人、浏览器、地图等一部分必要的核心应用程序，开发者若想要开发相关的应用，都是基于这一层来实现的。在这一层主要通过JAVA 语言调用Android 的API接口来编程的。

#### 2. 应用程序框架（Application Framework）

Android 应用程序框架使得开发人员可以访问核心应用程序的API 框架，与

Windows平台上的应用程序概念不同，Android采用了基于组件的设计理念。该设计理念可以支持组件的重用，开发人员可以开放自己程序中的某些模块供其他程序调用，同时，也可以调用其他程序开放的模块，这种设计使得开发人员能够很方便的替换应用程序中的某些组件。应用程序框架中经常用到的组件和服务主要有以下五种[20]：

##### （1) View System：主要用于构建应用程序的UI，包括文本框（Textbox）、列表（List）、按钮（Button）、浏览器（Browser）等。

##### （2) Activity Manager：主要用于管理应用程序的生命周期，并提供常用的导航回退功能。

##### （3) Content Provider：主要用于应用程序间的数据共享。

##### （4) Notification Manager：主要用于在应用程序的状态栏显示某些状态信息。

##### （5) Resource Manager：主要用于对布局文件、图片等非代码资源进行管理。

#### 3. 系统运行库（Libraries）

Android 系统运行库包含了Android 类库和Android 运行时组件两个部分。

Android类库是一些C/C++库，通过应用程序框架为开发者提供服务。Android平台的类库包括[21]：

(1) Lib C：专门为基于Embedded Linux的设备定制的标准C系统函数库。

(2) Media Framework：一个支持各种多媒体格式的音频和视频的库，编码格式包括

MP3、AAC、MPEG4、H.264、PNG、JPG、GIF等。

(3) Surface Manager：管理显示子系统，为应用程序提供2D和3D图形的无缝融合。

(4) SGL：底层的2D图形引擎。

(5) OpenGL | ES: 该库主要用于硬件的3D加速功能，是基于OpenGL ES 1.0 APIs

实现的。

(6) FreeType：处理有关位图和矢量字体的显示。

(7) SSL: Secure Socket Layer的缩写，是为网络通信提供安全及数据完整性的一种安全协议，用于保障在Internet上数据传输的安全，利用数据加密技术，确保数据在网络传输过程中不会被截取及窃听。

(8) SQLite：一种开放的、轻量级的关系型数据库，支持Windows / Linux / Unix等操作系统。由于其占用资源少、处理速度快，一般只需占用几百K的内存空间，所以主要用于嵌入式设备中。

(9) WebKit：一种开源的Web浏览器引擎，具有高效稳定、兼容性好的优势。比如

Google的手机Android、Apple的iPhone等所使用的Browser内核引擎，都是基于WebKit。

Android运行时组件主要包括核心类库和Dalvik虚拟机。核心类库包含大部分Java编程语言所需调用的类，另外还包括android. net、android. os、android. media等Google自主开发的Android Java类库。

Dalvik虚拟机是Google公司自己设计用于Android平台的Java虚拟机，它可以支持已转换为. dex（Dalvik Executable）格式的Java应用程序的运行，. dex格式是专为Dalvik设计的一种压缩格式，适合内存和处理器速度有限的系统。

#### 4. 系统内核（Linux Kernel）

Android系统服务采用Linux 2.6内核，主要提供进程管理、网络堆栈、驱动程序模型、安全性、内存管理的核心系统服务。Linux内核是基于软件和硬件之间的一个抽

象层，是两者之间的桥梁。包含以下这些常规应用的驱动程序[22]：

(1) Display Driver

(2) Camera Driver

(3) Bluetooth Driver

(4) Flash Memory Driver

(5) Binder(IPC) Driver

(6) USB Driver

(7) Keypad Driver

(8) WIFI Driver

(9) Audio Driver

### 2.1.2 Android开发环境搭建

Android平台为开发人员提供了开放和免费的开发环境，使用其提供的工具和帮助文档，开发人员可以快速的编写应用程序。开发环境的搭建步骤如下[23]：

#### 1. 下载Android SDK

Android 软件开发工具包Software Development Kit（SDK）是Google 公司针对

Android系统推出的手机开发平台，具有强大的功能和完备的调试手段。使用浏览器登录[http: //developer. android. com/](http://developer.android.com/)，下载Windows版本的SDK包，解压后即可使用。

#### 2. 下载JDK6

JDK（Java Development Kit）是Sun Microsystems针对Java开发者的产品，由于

Android是用Java语言来开发应用程序的，所以必须包含JDK的开发包。浏览器登录

[http: //www. oracle. com/technetwork/java/javase/downloads/index. html](http://www.oracle.com/technetwork/java/javase/downloads/index.html), 下载Windows平台

JDK6版本的开发包。

#### 3. 下载Eclipse3.7.2

Java IDE（Integrated Development Environment, 集成开发环境）的产品有很多，有商业性质的，也有免费开源的，如Eclipse、JBuilder、NetBeans. 在本项目中选用Eclipse作为开发工具，Eclipse是一个基于Java语言并且开放源代码的可扩展开发平台。浏览器登录[http: //www. eclipse. org/downloads/](http://www.eclipse.org/downloads/)，下载Eclipse Classic 3.7.2版本。

#### 4. 下载Eclipse中Android ADT插件

Android ADT（Android Development Tools）是在Eclipse IDE中构建Android应用

程序的插件。ADT扩展了Eclipse的功能，使用它可以快速建立新的Android项目。通过Eclipse软件菜单栏中的update功能即可下载Android ADT插件，添加的update站点为[http: //dl-ssl. google. com/android/eclipse/](http://dl-ssl.google.com/android/eclipse/)。

完成以上四个步骤后，打开Eclipse，如果可以创建Android Application Project项目，则证明Android开发环境搭建成功。

## 2.2 XML技术

XML（eXtensible Markup Language）又叫可扩展标记语言，现推荐遵循的是W3C于2000年10月6日发布的XML1.0标准，它是标准通用标记语言SGML（Standard Generalized Markup Language）的一个简化子集[24]。XML的数据结构是以开放、自我描述的方式来定义的。作为标记语言中的一种类型，它将数据及其描述组合起来，相比关系模型来说，它拥有更灵活的和更强大的数据描述能力。

使用XML 文档最大的好处在于数据的管理，它给开发者带来许多好处。因为

XML文档具有系统无关性和跨平台性，所以XML的解析技术适用于任何系统平台。在任何程序中，读取XML数据都很简单，这一优点使XML成为数据交互的公共语言。

### 2.2.1 XML文档结构

XML文档

文档节点

XML文档的结构由序言和文档元素两大部分组成。序言必须处于文档的顶部，包括处理指令、注释、DTD/样式等一些关于该文档的信息，文档节点位于序言之后的位置，包括元素、属性、文本、CDATA、实体、注释等内容[25]。XML文档结构如图2.2所示。

|  |  |  |
| --- | --- | --- |
| 处理指令 | 注释 | DTD/样式 |

元素

属性

文本

CDATA

实体

注释

**图2.2** **XML文档结构图**

序言

### 2.2.2 XML解析技术

在各类软件的开发过程中，XML技术都被广泛使用。XML本身是一种以纯文本方式进行数据编码的格式，要得知XML文件中所编码的数据含义，就必须有一个能够识别XML文档中文本信息的解析器，用来解释XML文档并提取其中的数据。然而，根据数据提取的不同需求，又存在着多种解析方式，不同的解析方式有着各自的优缺点和适用环境。选择合适的XML解析技术能够有效提升应用系统的整体性能。因此，了解和区分各种不同的XML解析技术就显得尤为重要。

在Android应用开发中，常用的解析技术分别为SAX解析技术、DOM解析技术和

PULL解析技术[26]。

#### 1. SAX解析技术

SAX是一个解析速度快并且占用内存少的XML解析器，非常适合用于Android等移动设备[27]。SAX解析XML文件采用的是事件驱动模式，也就是说，它并不需要解析完整个文档，而是在读取XML文档的过程中生成一个事件流，并且对于每个事件通过回调事件对程序中相应的方法进行处理，比如元素开始和结束标记，元素内容，实体，语法分析错误等事件。

SAX解析包括两个部分：解析器和事件处理器。解析器就是XMLReader接口，负责读取XML文档，向事件处理器发送事件；事件处理器就是ContentHandler接口，负责对发送的事件进行响应和对XML文档进行处理。ContentHandler接口中定义了与处理XML文档本身关联的事件的方法，如StartDocument、EndDocument、StartElement、

EndElement、Characters等。

<?xml version="1.0"?>

<sample>

<title>Android</title>

<content>xml</content>

<parser>sax dom pull</parser>

</sample>

针对以下简单的XML文档，SAX解析产生的事件如图2.3所示。

<xmlversion=" 1.0">< sample> <title>

Start Document

Start Element

Start Element

Characters

End Element

…...

End Element

End Document

Android

</title>…...

</sample>

#### 2. DOM解析技术

**图2.3** **SAX解析器生成的事件**

DOM解析技术使用层次化对象模型表示XML文档，即为XML语法中元素、属性、实体、文档等概念定义对应的类，解析器在读入XML文档的时候，会建立XML语法和类之间的一一映射关系。DOM的层次化对象模型是一个树形结构，它将一个XML文档看做一棵节点树，每个节点代表一个XML文档中的元素。DOM有5个基本节点对象：

(1) Document对象：代表树的最高节点，是整个文档的操作入口；

(2) Element对象：代表文档中元素的映射；

(3) Attr对象：代表文档中元素属性的映射；

(4) Text对象：代表Element和Attr对象的子节点，是元素或元素属性的文本内容；

(5) NodeList对象：代表节点的遍历方式。

针对SAX解析技术中的XML文档，其对应的DOM节点树如图2.4所示。其中，椭圆框表示元素节点，矩形框表示文本节点。

sample

title

content

parser

Android

xml

sax dom pull

Document

Element

Text

**图2.4** **DOM节点树**

使用DOM解析XML文件时，会将XML文件的所有内容读取到内存中，建立完整的XML文档树形结构，使软件开发人员可以方便的进行遍历、增加、删除、修改等操作，具有良好的导航能力[28]。然而，因为DOM需要遍历整个XML文档，在内存中

构建树形结构，需要消耗大量内存。所以，对于Android这类内存非常有限的移动设备来说，不建议采用DOM方式来解析XML文件。

#### 3. PULL解析技术

在Android系统中，存在很多XML格式的资源文件，Android系统解析这些XML文件时，采用的就是PULL解析技术[29]。PULL解析器的运行方式与SAX解析器相似，也是基于事件驱动模型。当PULL解析器开始解析后，调用parser. next（）方法主动获取如START\_DOCUMENT、END\_DOCUMENT、START\_TAG、END\_TAG、TEXT等事

件。

PULL解析器和SAX解析器虽然相似，但也有所区别。SAX解析器的工作方式是自动将事件源推入事件处理器进行处理，因此不能主动控制处理事件的结束。而PULL解析器的工作方式是程序主动从解析器中获取事件，因此可以在得到需要的数据后不再获取事件，结束解析。

由于PULL解析方式解析速度快，简单易用，并且Android官方推荐在开发应用程序时使用PULL解析器。所以，本论文中有关XML数据的解析，采用PULL解析技术。

## 2.3 通信技术

Android与服务器/网关主要有两种通信方式：一种是Http通信，另一种是Socket通信。两种通信方式之间最大差异的在于，Http通信使用的是“请求-响应模式”，也就是在请求时建立连接通道。只有客户端向服务器发送请求后，服务器才能向客户端返回数据。而Socket通信则是双方在进行数据传输前就事先建立好通信通道，连接建立之后，直接可以在该通道进行数据的传输，并可以实现服务器主动推送数据，而不是每次都需要由客户端向服务器发起请求。

### 2.3.1 Http通信

Http（Hypertext transfer protocol, 超文本传输协议）是一种详细规定了浏览器和服务器之间相互通信的规则，通过因特网传送万维网文档的数据传送协议[30]。Http是一种“无连接”、“无状态”的通信协议。“无连接”的含义是客户端与服务器的一次连接只能处理一个请求，即客户端发出请求并收到服务器的回应后，就断开该次连接。“无状态”的含义是指协议对于连接中的事务处理没有记忆能力，如果后续处理需要前面的状态信息，那么必须通过重传的方式解决。

Http通信中的请求方式分为两种：get请求方式和post请求方式。get请求方式用于获得静态页面，也可以把参数放在URL（Uniform Resource Locator，统一资源定位符）字串的后面，传递给服务器。post请求方式的参数不是放在URL字串里面，而是放在

Http请求数据中。因此，在编程之前，应当首先明确使用的请求方法，然后再根据所使用的方式选择相应的编程方式。

在Android平台中，提供了HttpURLConnection和HttpClient这两个接口来开发使用Http协议的程序。

1. HttpURLConnection接口

HttpURLConnection是继承于URLConnection的类，它们都是抽象类，无法直接实例化对象，其对象主要通过URL的openConnection方法获得，但openConnection方法只是创建URLConnection或者HttpURLConnection的实例，并不是进行真正的连接操作，代码如下：

URL url = new URL("http://192.168.0.100:8080");

HttpURLConnection connnection = (HttpURLConnection)url.openConnection();

connection.setRequestMethod("post");// 设置请求方式为 post

connection.setConnectTimeout(10000);// 设置连接超时为 10 秒

connection.connect();// 建立实际连接

#### 2. HttpClient接口

与第一种HttpURLConnection接口相比，HttpClient对java. net中的类做了封装和抽象，更适合在Android上开发互联网应用，需要了解以下一些类：

(1) ClinetConnectionManager接口。此接口是客户端连接管理器接口，有如下抽象方法：

ClientConnectionManager，关闭所有无效、超时的连接；

CloseIdleConnections，关闭空闲的连接；

ReleaseConnection，释放一个连接；

RequestConnection，请求一个新的连接；

Shutdown，关闭管理器并释放资源。

(2) DefaultHttpClient. 是默认的一个Http客户端，可用它创建一个Http连接。

(3) HttpResponse。是对一个Http连接的响应。当客户端请求一个Http连接后，就会收到一个来自服务器的HttpResponse响应信息。

针对客户端请求一个Http连接，并通过服务器的响应分析请求是否成功的事件，操作代码如下：

HttpClient client = new DefaultHttpClient (); //创建一个 Http 连接

HttpGet request = new HttpGet("[www.baidu.com](http://www.baidu.com/)"); //设置请求方式为 Get HttpResponse response = client.execute(request); //获得服务器响应信息if(response.getStatusLine(),getStatusCode() == HttpStates.SC\_OK){

……// 请求成功操作

}else{

……// 请求失败操作

}

### 2.3.2 Socket通信

Socket又称为套接字，它在程序的内部提供与外界通信的端口，也就是端口通信。通过建立Socket连接，可以为通信双方提供数据传输的通道。Socket支持TCP/IP协议簇的内容，在该协议中主要的Socket类型为流套接字和数据报套接字。流套接字使用

TCP作为其端对端的协议，提供可信赖的字节流服务。数据报套接字使用UDP协议，提供数据打包发送服务。在Socket通信过程中，端点必须包含：使用的连接协议、本地IP地址、本地进程的协议端口、远程IP地址、远程进程的协议端口这五项信息[31]。

Socket 的基本通信模型如图 2.5 所示。

应用程序

服务器

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Socket | |  | | |
| UDP | TCP |
| IP | |  | 信道 |  |
|  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | | Socket | |
| UDP | TCP |
|  | 信道 |  | IP | |
|  |  |

1. Socket传输模式

IP

**图2.5** **Socket通信模型**

Socket的传输模式分为两种：一种是基于面向连接的通信协议，另一种是基于无连接的通信协议。

面向连接的Socket操作就像一部电话机，必须要等对方接上之后才能通话，所有

的数据到达的顺序与它出发时的顺序是一样的。面向连接的操作使用TCP协议，即此模式下必须先连接上目的地的Socket，连接上后Socket就可以使用一个流接口进行打开、读、写、关闭等操作，所有发送的信息都会在另一端以同样的顺序被接收。面向连接的Socket虽然安全性高，但是传输效率较低。TCP通信模型如图2.6所示。

客户端服务器

输出流

OutputStream

InputStream

输入流

输出流

OutputStream

InputStream

输入流

Socket ServerSocket

**图2.6** **TCP通信模型**

无连接的Socket就像是一个邮件投递，没有保证，多个邮件到达时的顺序可能与出发时的顺序不一样，无连接的操作使用UDP数据报协议，一个数据报是一个独立的单元，它包含了这次投递的所有信息。可将其想象成一个信封，这个模式下的Socket不需要连接一个目的Socket，它只是简单的投出数据报。无连接的Socket操作优点是快速和高效，缺点是传输数据的可靠性不高。UDP通信模型如图2.7所示。

客户端服务器

DatagramSocket DatagramSocket





**图2.7** **UDP通信模型**

到底选用哪种操作方式由应用程序的需要决定，智能家居客户端系统的通信需要数据的正确性和有序性，因此选择面向连接的操作方式。

#### 2. 基于TCP协议的Socket编程原理

java. net包中提供两个类Socket和ServerSocket，分别用来表示双向连接的客户端和服务器端。两个类中的构造方法如下：

Socket(InetAddress address, int port); ServerSocket(int port);

类构造方法参数的含义：

address，双向连接中另一方的IP地址；

port，双向连接中另一方的端口号。

Socket client = new Socket("192.168.1.110",54321);

ServerSocket server = new ServerSocket(54321);

例如，以下代码构建一个客户端Socket和一个服务器端ServerSocket。

在选择端口时，每一个端口对应一个服务，只有给出正确的端口，才能获得相应的服务。0~1023的端口号为系统所保留，例如Http服务的端口号为80，telnet服务的端口号为21，ftp服务的端口号为23，所以选择端口号时最好选择一个大于1023的数，如上述代码中的54321，防止发生冲突。在创建Socket 时如果发生错误，将产生

IOException异常，所以在创建Socket和ServerSocket时必须捕获或抛出异常。

Socket提供了getInputStream（）和getOutPutStream（）两个方法来得到对应的输入（输出）流以进行读（写）操作，这两个方法分别返回InputStream和OutputStream类对象。为了便于读（写）数据，可以在返回的输入、输出流对象上建立过滤流，如：

DataInputStream、DataOutPutStream、或PrintStream类对象。在Socket使用完毕后需要将其关闭，以释放资源。在关闭Socket之前，要注意关闭的顺序，应将与Socket相关的所有输入、输出流先关闭后，再关闭Socket。

通过总结得出，使用Socket实现客户端的步骤如下：

(1)通过IP地址和端口实例化Socket，请求连接服务器；

(2)获取Socket上的流以进行读写；

(3)把流包装进BufferReader/PrintWriter的实例；

(4)对Socket进行读写；

(5)关闭打开的流。

使用ServerSocket创建服务器端的步骤如下：

(1)指定端口实例化一个ServerSocket；

(2)调用ServerSocket的accept()以在等待连接期间造成阻塞；

(3)获取位于该层Socket的流以进行读写操作；

(4)将数据封装成流；

(5)对Socket进行读写；

(6)关闭打开的流。

## 2.4 存储技术

在Android应用程序开发中，系统提供了两种类型的存储方式，一种是内部存储，另外一种是外部存储[32]。其中，内部存储包括SharedPreferences存储方式、文件存储方式以及SQLite数据库存储方式；外部存储包括ContentProvider存储方式和网络存储方式。在实际开发中，主要使用的是内部存储方式。

### 2.4.1 SharedPreferences存储方式

在很多情况下，开发人员设计的软件应当向用户提供设置软件参数的功能。以腾讯公司旗下的即时通信软件QQ为例，软件应当提供用户设置是否允许陌生人添加自己为好友这一选项。在Windows平台的应用中，通常采用. ini格式的文件保存软件的配置参数；在J2SE应用中，采用properties属性文件进行保存；而在Android平台下，系统提供了一个轻量级的存储类SharedPreferences类，特别适合于保存一些软件常用的配置参数。使用SharedPreferences存储数据，本质上是基于Map数据结构，以key-value键值对的方式存储，其格式采用XML文件形式，保存于/data/data/package name/shared\_prefs目录下。

### 2.4.2 文件存储方式

Android平台下的文件存储，与J2SE的文件存储方式相同，都是使用输入输出流，即IO流。Android中的文件存储分为两种形式，一种是机身存储方式，即将文件存储于设备自带的内存中。另一种是SDCard存储方式，即将文件存储于设备扩展的SDCard中。

#### 1. 机身存储

Android平台提供了Context. openFileOutput（）方法来对文件进行写入操作，该方法默认将文件保存于机身内存，存储位置位于/data/data/package name/files/目录中，代码如下：

public FileOutputStream openFileOutput(String filename,int operate\_mode);

其中，filename参数指的是要保存的文件名称，如果该名称的文件不存在，Android系统会自动创建它。operate\_mode 参数用于指定进行文件存储时的操作模式，Android

系统提供了四种操作模式，分别为：

(1) Context. MODE\_PRIVATE模式：该模式为文件存储的默认操作模式，表示该文件属于私有数据，只能被应用程序本身访问。需要注意的是，使用该模式时，文件中新写入的内容会覆盖原文件的内容。

(2) Context. MODE\_APPEND模式：使用该模式时，在写入文件前会首先检查该文件是否存在，若存在就向该文件的末尾追加新的内容，若不存在就创建新的文件。

(3) Context. MODE\_WORLD\_READABLE模式：该模式表示当前保存的文件可以被其他应用程序读取。

(4) Context. MODE\_WORLD\_WRITEABLE模式：该模式表示当前保存的文件可以被其他应用程序写入。

#### 2. SDCard存储

由于移动设备本身所带的存储空间不是很大，只能用于存放占用空间较小的文件。而对于比较大的文件，Android官方推荐将其存储在外部SDCard中，避免因占用过多内存导致系统变慢等影响。

向SDCard写入数据前，首先要在AndroidManifest. xml配置文件中加入访问SDCard的权限，然后判断手机是否装有SDCard，是否处于可读写状态。如果可用，则按照标准JavaIO对文件进行写入操作。

### 2.4.3 SQLite数据库存储方式

在Android系统中，集成了SQLite关系型数据库，它属于轻量级的嵌入式数据库，在嵌入式设备中，占用资源少，只要几百kb的内存就足够。在Android应用程序中创建数据库后，数据库文件保存在/data/data/package name/databases/目录下。

对于SQLite数据库的操作，一般包括创建、打开、删除、关闭数据库。对于数据库中表的操作，一般包括添加、删除、修改、查询操作。

在Android中使用SQLite数据库时，需要使用以下两个类：

#### 1. SQLiteOpenHelper数据库助手类

为了在开发过程中更好的使用和管理数据库，程序一般需要封装一个继承自SQLiteOpenHelper类的数据库助手类。在SQLiteOpenHelper类的构造方法中，分别需传入上下文环境、数据库名称、索引、数据库版本号这四个参数。使用构造方法创建出SQLiteOpenHelper 对象后，并且在数据库第一次被创建的情况下，程序会首先执行

SQLiteOpenHelper中的onCreate（）回调方法，在该方法中写入一些程序中需要用到的数据表。需要注意的是，创建SQLiteOpenHelper对象后，程序并没有真正的创建数据库，而是在使用该对象调用getWritableDatabase（）方法或getReadableDatabase（）方法后，才去创建数据库，并返回SQLiteDatabase数据库操作类的对象。

#### 2. SQLiteDatabase数据库操作类

在SQLiteDatabase类中，使用最频繁的就是对数据表的增、删、改、查操作。其中，insert（）方法表示添加记录，delete（）方法表示删除记录，query（）方法表示查询记录，update（）方法表示更新记录，execSQL（）方法表示执行一条SQL语句。

## 2.5 多线程模型

### 2.5.1 模型优势

Android应用程序支持多线程，使用多线程的编程方式可以使程序充分利用系统资源，为设计耗时操作提供了解决途径。多线程编程可以带来以下优势：

#### 1. 提高用户体验，避免ANR对话框

在Android平台中，默认情况下运行的应用程序在5秒内没有响应用户输入时，系统会向用户显示一个对话框，这个对话框就称为ANR（Application Not Responding，应用程序无响应）对话框，如图2.8所示。



**图2.8** **应用程序无响应对话框**

弹出ANR对话框后，用户可以根据自己的情况选择“等待”，或是“强制关闭”该应用。在一个合理的、流畅的应用程序中，应尽量避免出现ANR，否则会导致用户体

验很差。因此，在主线程中不能执行非常耗时的操作，所有可能耗时的操作都要放到其他线程去处理。

#### 2. 异步处理

在有些情况下，用户操作完某项功能后并不需要同步阻塞去等待操作结果的返回，而是在有了结果后，再去通知用户，这就可以通过多线程来实现异步处理。比如应用程序要在服务器端获取一张图片，由于加载图片比较耗时，所以使用异步加载，用户可以在加载图片的同时进行其他操作，等图片加载完成后，再去刷新界面。

### 2.5.2 消息机制

在Android平台下进行多线程编程时，经常需要在主线程之外的一个单独的子线程

中处理某些比较耗时的操作，子线程操作完成后，再去通知主线程更新U（I UserInterface，

用户界面）显示[33]。但是，在主线程之外的线程中直接更新UI时，系统会报ERROR/Android Runtime: android. view. ViewRoot$CalledFromWrongThreadException: Only the original thread that created a view hierarchy can touch its views异常，原因是Android平台的机制设定在子线程中无法直接更新Activity中的UI，一般的做法是子线程向Activity传递消息，然后Activity根据这些消息自己来更新UI。

Android中实现主线程和其他线程间通信的方式有很多种方法，本论文中只介绍经常用到的两种方式：

#### 1. AsyncTask方式

AsyncTask是Android系统提供的异步处理辅助类，使用这个类可以为耗时程序开启一个新线程进行处理，处理完时返回结果给主线程。这样做的好处是不需要开发人员去管理线程间的通信，方便使用。

使用AsyncTask类实现异步通信时，主要用到该类内部的四个回调方法：

(1) onPreExecute（）方法：运行于程序的主线程，用于在开始异步操作前向用户作出相应提示。

(2) doInBackground（）方法：运行于程序的子线程，用于进行异步数据处理。

(3) onPostExecute（）方法：运行于程序的主线程，用于进行结束提示处理。

(4) onProgressUpdate（）方法：运行于程序的主线程，用于在进行异步操作时通过进度条向用户显示当前处理进度。

AsyncTask类的回调逻辑关系如图2.9所示。



execute()

onPreExecute()

onPostExecute()

主线程

AsynTask子线程

doInBackground()

#### 2. Handler方式

**图2.9** **AsyncTask回调逻辑关系图**

Android系统中提供了一个类叫做android. os. Handler，就是专门用来在子线程和主线程之间传递消息用的。Handler运行在主线程中，它与子线程通过Message对象传递数据。当Handler收到包含数据的Message对象后，将该对象放入主线程队列中，调用handleMessage（）方法配合主线程对UI进行更新，其消息传递逻辑关系如图2.10所示。

Handler

Message

handleMessage()

主线程

子线程

obtainMessage()

SendMessage()

**图2.10** **消息传递逻辑关系图**

## 2.6 编码转换机制

在通信过程中，发送方与接收方对于中文字符的编解码标准不一致时，收到的数

据就会产生乱码问题。要想解决Android通信中的中文乱码问题，首先需要了解字符、字符集、字符编码的概念。字符是文字与符号的总称，包括文字、图形符号、数学符号等；字符集是一组抽象字符的集合，常常和一种具体的语言文字对应起来，该文字中所有字符或者大部分字符就构成了该字符集，比如英文字符集、繁体汉字字符集、日文字符集等；字符编码是计算机要处理的各种字符，需要将字符和二进制内码对应起来。

要确定编码首先要确定字符集，并将字符集内的字符排序，然后和二进制数字对应起来。根据字符集内字符的多少，确定用几个字节来编码。ASCII编码是目前用得最广的字符集及编码；Unicode编码是计算机上使用的字符编码，UTF-8就是Unicode编码的实现方式；GB2312字集是简体字集；GBK字集是简繁字集。

网络通信中，产生乱码的原因主要是通信过程中使用了不同的编码方式，如：服务器中的编码方式、传输过程中的编码方式、传输到达终端设备的编码方式。因此，在传输过程中至少需要两次编码转换，首先从服务器编码转换为网络编码，再从网络编码转换为终端设备编码。

在智能家居客户端系统中，将通信中得到的数据流先转换为字节，然后再将字节按

InputStream is = conn.getInputStream(); BufferedInputStream bis =new.BufferedInputStream(is); byte bytearray[] = new byte[1024];

int current = -1; int i = 0;

while((current = bis.read()) != -1){ bytearray[i] = (byte) current; i++;

}

resultData = new String(bytearray,"GB2312");

GB2312的方式进行转换得到字符串，最后的字符串便可显示中文效果，代码如下：

## 2.7 布局技术

用户界面（User Interface）的布局，对于Android程序设计来说是至关重要的，因为它直接关乎用户的视觉体验，要想实现自己设计的程序界面，就必须先了解Android平台提供的View类。View类是Android平台可视化组件的基类，主要提供屏幕布局的

处理和与用户交互的方法，可视化组件正是重写了这些方法的对象，比如文本组件、按钮等[34]。

Android平台自身给开发者提供了各种各样的可视化组件，在使用的时候只需设置它们的属性即可。设置组件属性有两种方法，一种是在xml文件中设置静态属性，另一种是在Java代码中设置动态属性。常用的可视化组件有文本组件、可编辑文本组件、图像控件、按钮、单选按钮、复选按钮、图像按钮等。

设置好可视化组件的属性之后，需要将这些组件合理的安排在手机屏幕上进行显示。Android平台提供给开发者管理组件在屏幕上显示位置的是Layout类，Android SDK提供了几种布局类型，通过这些布局类型的组合嵌套，开发者可以设计出各种复杂的用户界面，布局可以在xml布局资源文件中定义，也可以用Java代码在程序中动态的定义。

Android的界面是由布局和组件协同完成的，布局好比是建筑里的框架，而组件则相当于建筑里的砖瓦。组件按照布局的要求依次排列，就组成了用户所看见的界面。Android的五大布局分别是线性布局、相对布局、绝对布局、表格布局及单帧布局，下面分别介绍这五种布局类型的特性：

#### 1. 线性布局（Linear Layout）

线性布局是Android开发者使用的最多的布局类型之一，用它来组织用户界面上控件的摆放位置。线性布局的作用就像它的名字一样，将控件组织为一个垂直或水平布局的形式。当布局方向设置为垂直时，它里面的所有子控件被组织在同一列中。当布局方向设置为水平时，所有子控件被组织在一行中。

线性布局中最重要的属性包括：

(1)方向属性（android: orientation），取值可以是vertical或horizontal；

(2)对齐属性（android: layout\_gravity），用来控制子控件在线性布局中如何排列和显示；

(3)权重属性（android: layout\_weight），指定每个子控件在线性布局中的相对重要性；

(4)宽度属性（android: layout\_width），指定该线性布局的宽度；

(5)高度属性（android: laytou\_height），指定该线性布局的高度。

#### 2. 相对布局（Relative Layout）

相对布局是Android五大布局结构中最灵活的一种布局结构，比较适合一些复杂界面的布局，它的子控件根据所设置的参照物控件和参数来确定它们在屏幕上的位置，使

用相对布局时有一点需要注意的是参照物控件必须要在参照它的控件之前定义，该参照物控件可以是父控件，也可是其他的子控件。

#### 3. 绝对布局（Absolute Layout）

绝对布局是通过设置组件的坐标来决定其所处位置的，屏幕的左上角为坐标原点

（0,0），第一个0代表横坐标，向右移动此值增大，第二个0代表纵坐标，向下移动此值增大，在绝对布局中的子元素可以相互重叠。绝对布局是根据坐标来进行布局的，这是不推荐的，因为无法预知运行程序的设备实际情况，如果定义了坐标，很可能程序是不能正常显示的，所以在Android2.3之后该布局结构就被废除了。

#### 4. 表格布局（Table Layout）

表格布局是通过行、列的形式来管理UI控件，Table Layout并不需要明确的声明包含多少行、多少列，而是通过Table Row，以及其他组件来控制表格的行数和列数。Table Row也是容器，因此可以向Table Row里面添加其他控件，每添加一个控件该表格就增加一列。如果在Table Layout里面添加控件，那么该控件就直接占用一行。在表格布局中，列的宽度由该列中最宽的单元格决定，整个表格布局的宽度取决于父容器的宽度，默认情况下是占满父容器本身。

Table Layout继承了Linear Layout，因此它完全支持Linear Layout所支持的全部xml

属性。

#### 5. 单帧布局（Frame Layout）

单帧布局是五大布局中最简单的一个布局，在这个布局中，整个界面被当成一块空白备用区域，所有的子元素都不能被指定放置的位置，它们统统放于这块区域的左上角，并且后面的子元素都直接覆盖在前面的子元素之上，将前面的子元素部分或全部遮挡。

## 2.8 本章小结

本章主要对客户端系统所涉及到的Android平台中的关键技术做了深入研究，包括Android移动平台系统，XML文档结构及解析技术，通信中的Http通信方式和Socket通信方式，分析项目中哪些功能需要使用Http实现，哪些功能需要Socket实现。对通信中可能产生中文乱码问题进行研究，防止客户端界面中乱码的产生；对比了Android系统中的三种数据存储技术，阐述各自的特点和优势；因为涉及到网络通信，所以必然会涉及到多线程编程技术，需要了解Android中的多线程机制；最后，对需要使用到的Android布局方式和界面控件进行相关介绍。

# 第三章 客户端软件设计

## 3.1 智能家居体系结构

### 3.1.1 总体框架

通过对比分析目前市场上现有的几种智能家居系统解决方案，本论文采用流行的四层架构系统[35]。系统总体上分为四个部分：客户端、服务器、家庭网关、前端控制器。客户端由用户操作，负责命令的发送；服务器作为客户端和家庭网关之间的桥梁，负责两者之间通信数据的转发；家庭网关从服务器或客户端接收命令后，转发至对应的前端控制器；前端控制器作为命令的最终执行者，对设备进行相应的控制操作。本论文的研究重点在于客户端部分，因此，系统中的其他部分在此不再详述。

系统总体框架图如图3.1所示。

**GPRS/3G** **WIFI**



客户端



Internet

服务器

服务器

Internet

**WIFI**

家庭网关

信号基站



**Android**客户端

无线访问点



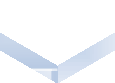
**ZigBee**

**Android**客户端

无线路由器

家庭网关

前端控制器



红外设备控制器灯光控制器墙壁开关控制器 灯光控制器

**图3.1** **系统总体框架图**

### 3.1.2 控制方式

智能家居系统的控制方式分为两种：一种是用户使用家用Wi-Fi网络连接家庭网关，

再由家庭网关控制前端设备的内网控制方式；另外一种是用户首先通过GPRS/3G/Wi-Fi网络访问服务器，服务器查找用户所对应的家庭网关，然后再由网关控制前端设备的外网控制方式。

1. 内网控制

内网控制指的是用户通过Wi-Fi连接至家庭内部的无线局域网，然后通过该内部网络与家庭网关进行通信的控制方式。内网控制的优势是在互联网不可用的情况下，仍可通过客户端软件对家居设备进行控制。内网控制的工作流程如图3.2所示[22]。

|  |  |
| --- | --- |
| Android  客户端 | Wi-Fi |
|  |

|  |  |
| --- | --- |
| 无线路由器 | LAN |
|  |

|  |  |
| --- | --- |
| 家庭网关 | ZigBee |
|  |

前端控制器

2. 外网控制

**图3.2** **内网控制工作流程图**

外网控制指的是客户端软件通过访问Internet上的服务器实现对家居设备的远程控制。客户端软件只与服务器通信，由服务器负责找寻客户端账号所对应的家庭网关。外网控制的好处是只要用户所使用的客户端具有上网功能，那么，就可以随时随地对家居设备进行控制，不再受所处地理位置的影响。外网控制的工作流程如图3.3所示。

i

|  |  |
| --- | --- |
| Android客户端 | GPRS/3G/Wi-F |
|  |

|  |  |
| --- | --- |
| 服务器 | Internet |
|  |

|  |  |
| --- | --- |
| 无线路由器 | |
|  | LAN |

前端控制器

|  |  |
| --- | --- |
| ZigBee | 家庭网关 |
|  |

**图3.3** **外网控制工作流程图**

## 3.2 需求分析

在软件开发过程中，需求分析是非常重要的一个环节。需求分析的概念是指对所要开发的应用程序进行详细研究，确定程序具有什么功能，应该达到什么样的效果等。

通过对用户的需求进行分析，将智能家居客户端系统分为功能性需求和非功能性需求两种。

### 3.2.1 功能性需求

智能家居系统中的Android客户端主要用于给用户提供一个交互的平台，用户通过客户端软件可以查看家庭中所有在线设备的工作状态，选择想要实现的功能，进行操作。

通过研究分析，总结出Android客户端需要实现的功能如下：

#### 1. 用户登录

智能家居客户端软件需要涉及用户隐私及家庭安全，因此，必须通过登录功能来确保使用者身份的合法性，保障用户的利益。

#### 2. 软件更新

当程序发布了新版本时，客户端有相应的升级提示，用户可以从服务器下载最新的应用程序进行软件的更新。

#### 3. 内/外网通信方式

在家庭宽带网络不可用的情况下，为了保障用户能够继续使用客户端软件控制家庭中的电器设备，本项目使用内/外网结合的通信方式，在室外时，通过服务器中转连接到家庭网络，在室内时，客户端的数据不再通过服务器，而是直接与家庭网关进行通信，这样设计的优势是一方面不再受宽带网络是否可用的影响，另一方面可以帮用户减少数据流量的使用。

#### 4. 自定义房间布局

智能家居产品面向的是很广泛的客户群体，使用者的居住环境有着很大的差别，因此，客户端系统采用了自定义房间布局的设计理念，即用户可以自主添加、删除房间，以适应自己房间的户型。

#### 5. 自定义场景模式

利用场景模式，用户可以实现设备的联动操作，并可以对同一区域的多路灯光同时控制，既方便操作，又能赋予同一空间各种不同的功能。系统预设几种常用的场景模式，如回家模式、离家模式、度假模式等。同时，用户也可根据自己的需求自定义场景模式。

#### 6. 自定义前端设备位置

前端设备所处位置可以通过客户端软件预先设置，适应某种固定房间布局，可以减少用户的操作复杂度，但是这种方式最大缺点是当其用在另一种户型时，功能几乎完全失效，通用性大大降低。为了弥补这种缺陷，本论文采用自定义方式指定前端设备所处位置，不再受房间布局影响，具有较高的可移植性。

#### 7. 查看设备列表

通过此功能，用户可以明确房间内所有在线的设备，从而很快的定位到某个具体的设备进行操作。

#### 8. 查看设备状态

客户端软件在每次运行后，会自动获取保存在网关的所有前端设备状态。任何一个前端设备状态发生改变时，都会向网关发送消息，然后网关再向客户端软件发送设备状态更新消息，使客户端软件时刻获知前端设备最新状态。

#### 9. 控制设备状态

用户通过客户端软件可以控制设备状态，包括开关灯、开合窗帘、开关电视、开关空调、对可调灯调光等，被控前端设备执行命令后回复是否成功执行。客户端软件的可视化交互界面，使得操作极为简便，仅需找到待控制的设备，轻触按钮，即可完成控制。

#### 10. 系统管理

通过此功能，用户可以修改账户、家庭网关的密码，查看本软件的使用说明以及版本信息等。

分析总结客户端软件的功能需求后可以得出，软件系统包含登录、软件更新、设备操作、自定义设置、系统管理五个主要用例，软件用例如图3.4所示。



**图3.4** **客户端软件用例图**

### 3.2.2 非功能性需求

软件除了需要满足功能需求外，还应具有一些非功能性需求。非功能需求从功能之外角度对软件进行约束。非功能性需求对软件质量、可靠性等方面有着重要意义，对软件的稳定，可靠，安全运行有着重要影响，是软件价值的重要组成部分。

软件的非功能性需求归纳起来主要有以下几个方面：

#### 1. 易使用性

排在第一位的是软件的易使用性，因为软件的功能再强大，要是操作起来十分复杂的话，相信是不会有很好的用户体验的。好的软件要有简洁的操作界面，符合用户的操作习惯，让用户一目了然。

#### 2. 可靠性

软件可靠性是在软件开发过程中需要关注的重要问题。软件除了能处理非法输入数据的状况，还需要能处理意料不到的操作的状况。软件的容错能力和故障恢复能力也是衡量软件的重要指标，关系到用户数据的准确性和一致性。

#### 3. 可维护性

软件除了实现功能外，还需要有较高的质量，而可维护性则是质量的重要特征之一。用户在使用软件过程中，如果发现错误或者对需求发生变化，就需要维护或开发人员对软件进行修改，而维护过程时要尽量做到不影响原有功能，且简单易行。影响可维护性的因素较多，主要包括软件开发技术、代码复杂度等因素。

#### 4. 可扩充性

软件的可扩充性也是衡量软件好坏的重要方面。可扩充性往往通过在软件开发过程中预留一些可配置的参数，达到不修改系统设计而适应新技术和需求。在系统设计初期就要充分考虑到后期可能发生的条件，做到充分准备。具体操作为将流程中业务节点划分为独立的对象，尽量做到低耦合，对象之间通过接口相互协调，通过修改个别模块即可适应新的需求。

#### 5. 兼容性

因为基于Android平台定制的手机版本众多，所以设计时应充分考虑系统不同版本之间的兼容性问题。同时，由于软件会被拥有各种分辨率的客户端用户下载使用，故要能同时在不同分辨率的客户端中正常显示。

#### 6. 资源占用率

因为手机的资源十分宝贵，所以在设计软件时要考虑软件对系统的CPU及内存的占用率，避免软件因占用过多资源而对系统造成不良后果。

### 3.2.3 部分功能需求详细描述

#### 1. 登录功能需求描述

(1)功能用例

用户登录可以分为内网登录方式和外网登录方式，软件对于服务器和网关的IP 地

址有默认的设置，如需修改，在登录界面点击菜单键即可重新设置IP地址。登录功能的用例如图3.5所示。



(2)需求描述

**图3.5** **登录功能用例图**

|  |  |
| --- | --- |
| **用例名称** | 登录功能 |
| **用例标识** | UC01 |
| **用例描述** | 在客户端软件上输入用户名、密码实现登录功能 |
| **参与者** | 客户端软件用户 |
| **前置条件** | 服务器、网关处于正常联网状态 |
| **触发条件** | 在登录界面点击登录按键 |
| **基本事件流描述** | 1: 用户启动客户端软件  2: 用户设置服务器和网关的 IP 地址（可选）  3: 用户输入用户名、密码  4: 用户点击内网或外网登录按键  5: 系统向服务器/网关发送数据  6: 系统等待数据反馈  7: 系统根据反馈结果进行页面的跳转 |
| **扩展事件流描述** | 无 |
| **异常事件流描述** | 无 |
| **后置条件** | 用户登录成功，进入软件主界面 |
| **业务规则** | 无 |
| **特殊需求** | Android 终端处于联网状态 |

#### 2. 软件更新功能需求描述

(1)功能用例

用户启动客户端软件后，软件在后台自动连接服务器获取最新版本信息，若有新版本软件发布，则向用户弹出升级提示，询问用户是否下载更新包，下载完成后，询问用

户是否覆盖安装更新包。软件更新功能用例如图3.6所示。



(2)需求描述

**图3.6** **软件更新功能用例图**

|  |  |
| --- | --- |
| **用例名称** | 软件更新功能 |
| **用例标识** | UC02 |
| **用例描述** | 后台自动获取最新版本信息，由用户选择是否进行更新 |
| **参与者** | 客户端软件用户 |
| **前置条件** | 服务器处于活动状态 |
| **触发条件** | 启动客户端软件 |
| **基本事件流描述** | 1: 用户启动客户端软件  2: 系统后台获取最新版本信息  3: 若有新版本，询问用户是否选择升级  4: 下载完成后，询问用户是否覆盖安装 |
| **扩展事件流描述** | 无 |
| **异常事件流描述** | 连接服务器失败 |
| **后置条件** | 软件更新成功 |
| **业务规则** | 无 |
| **特殊需求** | Android 终端处于联网状态 |

#### 3. 系统管理功能需求描述

(1)功能用例

用户登录成功后，可以在系统管理界面进行修改账户密码、修改网关密码、查看操作手册、查看版本信息操作。系统管理功能用例如图3.7所示。



(2)需求描述

**图3.7** **系统管理功能用例图**

|  |  |
| --- | --- |
| **用例名称** | 系统管理功能 |
| **用例标识** | UC03 |
| **用例描述** | 查看软件信息，修改账户信息等 |
| **参与者** | 客户端软件用户 |
| **前置条件** | 用户登录成功 |
| **触发条件** | 无 |
| **基本事件流描述** | 1: 用户由主界面进入系统管理界面  2: 用户选择系统管理界面中的某个功能选项 |
| **扩展事件流描述** | 无 |
| **异常事件流描述** | 无 |
| **后置条件** | 密码修改成功 |
| **业务规则** | 无 |
| **特殊需求** | Android 终端处于联网状态 |

## 3.3 业务流程

现如今，软件系统的业务流程设计在软件设计过程中占得比重越来越大，一个合理的、简便的软件使用流程，可以帮助开发人员更有条理的进行软件架构的设计。同时，一个业务流程设计合理的软件拥有着更好的用户体验，且能够吸引越来越多的使用者。因此，在软件设计过程中应当对软件的业务流程设计给予足够的重视。

### 3.3.1 程序初始化流程

一般的Android应用软件都有欢迎界面，在程序启动时可以向用户展示软件的相关信息动画，使程序显得更加绚丽，又可以在播放欢迎动画的期间做一些软件的初始化操

作。本论文的初始化设置流程主要是指智能家居客户端软件显示欢迎界面与显示登录界面之间所做的相关初始化操作。具体流程场景如下：

(1)用户点击智能家居客户端软件，显示欢迎动画界面。

(2)程序在后台访问服务器，得到服务器端程序版本号，与本地客户端软件版本号进行对比，若需要更新，则向用户弹出更新提示，程序停留在欢迎界面。

(3)若程序不需要更新，则访问运行该应用的Android平台SQLite数据库，从数据库中取出用户表和地址表的数据，将地址表的数据设为服务器和网关的默认IP地址。

(4)程序跳转至登录界面，若用户以前选中过记住密码选项，则将上一步取出来的用户数据添加到用户名列表及密码文本框中，方便用户再次登录。至此，软件初始化设置流程结束。

### 3.3.2 场景模式设置流程

本论文中场景模式指的是用户通过设置可以实现家居设备的联动操作，对一系列设备的控制操作一次性完成，而不必一个一个单独进行操作，可使用户更加方便快捷的控制家庭电器设备。系统本身预设一部分常用场景，用户也可自定义添加场景。具体流程场景如下：

(1)用户成功登录智能家居客户端系统后，进入程序的主界面。

(2)单击场景模式，从服务器/网关处读取系统预设和用户自定义的场景模式数据并以列表的形式显示，选中列表中的某一项进行开启或关闭操作。若收到“操作成功”的反馈信息，则表明该场景模式已生效。

(3)按下场景模式并保持两秒钟以上，进入场景模式的编辑界面。点击添加场景，可自定义一个新的场景模式并添加该场景模式中需关联的设备。

(4)在场景模式的编辑界面，按下列表中的某个已有场景两秒钟以上，可以修改或删除该场景模式。

### 3.3.3 房间布局设置流程

因为智能家居产品面向的客户群体拥有各式各样的房型结构，所以，客户端系统的设计采用了用户自主设置房间的方式。系统设置的默认房间结构是三室两厅一厨两卫

（客厅、餐厅、主卧、次卧、客卧、厨房、主卫、客卫），用户可以自行调整以适应自己的房间结构。具体流程场景如下：

(1)用户成功登录智能家居客户端系统后，进入程序的主界面。

(2)单击“我的房间”按键，从服务器/网关处读取用户的房间数据并以网格视图的形式显示，从视图中选择某个房间并单击，进入该房间的设备列表界面。

(3)按下“我的房间”按键并保持两秒钟以上，进入“我的房间”编辑界面。单击

“添加房间”按键，输入房间名称并选择房间类型后，点击确认即可完成房间的添加。

(4)在“我的房间”编辑界面，按下列表中的某个房间并保持两秒钟以上，可进行修改或删除该房间的操作。

### 3.3.4 设备位置定义流程

由于在客户端中将某个设备固定在某个房间中的设计方式通用性较低，只能适应某种特定的房间布局，因此，为了弥补这种缺陷，本论文采用自定义方式指定前端设备的位置，哪个设备所处哪个位置完全由用户定义，不再受房间布局的影响。具体流程场景如下：

(1)用户成功登录智能家居客户端系统后，进入程序的主界面。

(2)单击“设备列表”按键，从服务器/网关处读取已有位置设备的数据并以列表的形式显示，每一行数据代表一个设备，并显示其名称及所在位置。选中列表中的某一项单击后，进入其设备状态的控制界面。

(3)按下“设备列表”按键并保持两秒钟以上，进入“设备列表”编辑界面。单击

“添加设备”按键，得到位置为空的设备列表，选中其中的某一项可进行设置设备位置或者删除设备操作。

### 3.3.5 设备控制操作流程

智能家居客户端系统的主要功能是控制家庭电器设备的状态，也是用户使用最频繁的一项功能，因此，设备控制的操作必须便捷、灵活，才能拥有更佳的用户体验。本论文中设备控制功能有两种途径实现：

1. 通过房间查找设备。当用户欲控制某个房间里的若干设备时，采用此种方式最为便捷。具体流程场景如下：

##### （1) 用户成功登录智能家居客户端系统后，进入程序的主界面。

##### （2) 单击“我的房间”按键，界面跳转到“我的房间”列表。

(3)假设用户要控制的是客厅的窗帘设备，则单击“客厅”按键，进入客厅界面，并显示客厅里的所有设备。

(4)点击客厅中的“窗帘”按键，进入其状态操作界面，控制状态后，返回到客厅

设备列表界面。

2. 通过设备列表实现对设备状态的控制。当用户欲单独控制某个设备时，使用这种方式可以很快的定位到该设备进行操作。具体流程场景如下：

(1)用户成功登录智能家居客户端系统后，进入程序的主界面。

(2)单击“设备列表”按键，跳转至设备列表界面。

(3)假设用户要控制的是客厅的窗帘设备，则在“设备列表”中查找到该设备，单击后即可进入其状态操作界面。

## 3.4 功能模块划分

智能家居客户端系统集成了如登录、更新、设备操作、设备设置、系统管理等诸多功能，在进行设计的时候将整个软件分成各个功能模块，有利于开发。下面将首先从软件的总体设计来看智能家居系统客户端的模块化构造，然后针对模块中的各个子模块进行概要设计说明。

根据智能家居客户端系统的功能需求和实现目标，整个软件的总体模块划分及关系如图3.8所示。



**图3.8** **客户端系统模块划分图**

根据图3.8可知，客户端系统被细分为八个模块，各个模块的具体功能如表3.1所示。

**表3.1** **各模块的基本功能**

| 模块名称 | 基本功能 |
| --- | --- |
| 登录模块 | 负责处理用户的登录逻辑，通过调用网络通信模块连接服务器/网关  进行登录操作 |
| 软件更新模块 | 软件启动后通过网络通信模块自动连接服务器进行检查，当发现软件有新版本时提示用户进行更新，若用户选择更新，通过网络通信  模块从服务器下载更新文件，完成更新操作 |
| 控制操作模块 | 用户使用客户端软件进行查看设备状态、控制设备状态、自定义房  间布局、自定义前端设备位置等操作 |
| 网络通信模块 | 主要是在登录模块、软件更新模块、控制操作模块需要联网和服务器/网关进行交互时，调用协议组装模块组装数据，并调用发送模块将数据发送到服务器/网关，并且通过接收模块接收服务器/网关的反  馈数据，然后调用协议解析模块对数据进行解析 |
| 数据发送模块 | 通过 TCP/IP 协议将数据发送到服务器/网关 |
| 数据接收模块 | 从服务器/网关接收反馈数据 |
| 协议组装、解析模块 | 1. 根据各个模块的请求，组装不同的数据  2. 负责解析服务器反馈的数据 |
| 数据存储模块 | 将登录模块和控制操作模块中需要持久化的数据存入 Android 平台  自带的 SQLite 数据库中 |

### 3.4.1 用户登录

登录模块主要是指用户在登录页面输入用户名、密码并点击登录按钮之后的一系列过程。因为本论文采用了内网和外网这两种控制方式，所以用户登录时需选择内网登录方式或外网登录方式。

#### 1. 内网登录

客户端软件将用户的登录信息发送到系统的专有服务器，服务器将验证数据转发至用户所对应的家庭网关，网关对用户信息进行验证后返回结果给客户端。若用户身份合法，则进入软件的主界面；若身份不合法，则返回登录界面。

#### 2. 外网登录

客户端软件将用户的登录信息发送到家庭网关，网关对用户信息进行验证后返回结

果给客户端。若用户身份合法，则进入软件的主界面；若身份不合法，则返回登录界面。登录模块活动图如图3.9所示。



**图3.9** **登录模块活动图**

### 3.4.2 软件更新

软件更新是对旧版本软件的一种升级方式，一般包括新增功能模块，或对已有功能模块的增强和完善。通过软件更新，可以修复产品中存在的错误和BUG。

用户运行客户端软件后，后台访问智能家居的专有服务器，读取存储在服务器端的版本信息文件，解析该文件后得到服务器端保存的软件版本号及新版本的下载地址，通过比对服务器端和本地的版本号确定正在使用的软件是否最新版本。若是最新版本，则继续运行当前版本，进入登录界面；若不是最新版本，提示用户升级软件，升级完成后自动安装，安装成功后重启运行更新版本软件。

软件更新模块活动图如图3.10所示。



**图3.10** **软件更新模块活动图**

### 3.4.3 网络通信

网络通信模块主要用于处理各个功能模块发起的网络请求，通过TCP/IP协议和约定好的通信协议与服务器/网关进行通信并负责协议报文的组装和解析。

网络通信是指客户端软件与服务器或者家庭网关之间的数据交换过程。客户端软件可以主动向服务器或网关发送数据，也可以被动接收来自它们的数据。以控制前端电器

设备的状态为例，网络的通信过程可以分为两部分，一部分是客户端向前端电器设备发送控制命令的过程，另一部分是前端电器设备对客户端控制命令的反馈过程。

为了考虑用户的使用体验，本论文采用异步通信方式，发送模块和接收模块分别是程序的两个子线程，专门用于与服务器/网关之间的数据交互。当用户需要发送命令时，协议组装模块生成请求报文，并将该报文填入发送线程中的数据区，然后由发送线程向服务器/网关发送报文，发送完数据后，回到等待用户指令状态。当服务器/网关向客户端发送指令时，客户端的接收线程监测到数据的到来，调用协议解析模块对报文进行解析，并将解析结果发送给数据的请求模块，最后返回到等待服务器/网关指令状态。

网络通信模块活动图如图3.11所示。



**图3.11 网络通信模块活动图**

### 3.4.4 界面操作

整个项目的界面操作分为两种方式，一种是单击操作，另一种是按下某个按键并保持两秒钟以上，统称“长按”操作。界面中对按键的单击操作是进入下一级的控制界面，可以进行控制操作；对按键的长按操作是进入编辑界面，可以进行添加、删除、修改操作。操作后，根据反馈信息跳转至相应的界面。

界面操作模块活动图如图3.12所示。



**图3.12** **界面操作模块活动图**

### 3.4.5 数据存储

数据存储也叫做数据持久化，就是将内存中的数据模型转换为存储模型，或者将存储模型转换为内存中的数据模型的统称。本论文采用数据库的方式保存数据，使用数据库的好处是维护数据简便，可以直接调用Android系统提供的SQLite数据库接口对数据进行访问和操作。

当其他模块产生数据持久化请求时，调用数据持久化模块，若是第一次操作，就创建数据库，反之则打开数据库，然后按照主键查询数据表中是否有该条数据记录，若可以查询到，说明该条请求为修改操作，若查询不到，说明该条请求为插入操作。数据库进行相应操作后，通知请求模块该条请求的操作结果，最后关闭数据库。

数据存储模块活动图如图3.13所示。



**图3.13** **数据存储模块活动图**

## 3.5 通信协议设计

通信协议是客户端和网关/服务器沟通交流的语言，协议的设计应当简单明了，每个字段的长度既要能表达清楚含义，又不能冗余，因为字段过长对于用户来说增加了更多流量的消耗。

由于本论文采用了内、外网这两种通信方式，因此通信协议的设计也分为手机-网关的通信协议和手机-服务器的通信协议。在此以手机-网关的通信中控制设备状态的命令为例说明协议的设计及其字段的含义，如表3.2所示。

**表3.2** **设备状态控制指令协议**

| 说明 | 代码 | 长度 | 备注 |
| --- | --- | --- | --- |
| 帧头 | 0x56 | 1Byte | 协议开始的标识字段 |
| 控制码 | 0x0A | 1Byte | 指令类型的标识字段 |
| 设备 ID | …… | 1Byte | 查找前端设备的标识字段 |
| 设备状态 | …… | 1Byte | 前端设备状态的标识字段 |
| 校验码 | …… | 1Byte | 和校验结果的标识字段 |
| 帧尾 | 0x7F | 1Byte | 协议结束的标识字段 |

说明：

(1)设备ID字段：指前端设备唯一的ID号，由网关生成，通过此ID号可找到其对应的唯一设备。

(2)设备状态字段：表明前端设备所处的状态。若该设备为灯，则包括开、关两种状态；若该设备为窗帘，则状态为当前窗帘开合度的百分比值。

为了降低程序的耦合性，本论文将通信协议封装为一个类，对外提供一个接口，需要使用某条协议时，直接调用该类中的方法即可。以控制设备状态的协议为例，介绍通信协议类的封装方法，代码如下：

// @param ID 设备 ID, @param state 设备状态

public static byte[] deviceControl (byte ID, byte state){ int tag = 0;

byte[] buffer = new byte[6]; buffer[0] = 0x56; // 帧头buffer[1] = 0x0A; // 控制码buffer[2] = ID; // 设 备 ID buffer[3] = state; // 设备状态

for (int i = 0; i < buffer.length - 2; i++) {//得到校验码之前所有字段的和tag += buffer[i];

}

buffer[4] = (byte) (tag + 0x7F);// 校验码

buffer[5] = 0x7F;// 帧尾

return buffer;

}

## 3.6 本章小结

本章主要阐述了智能家居系统客户端软件部分的设计工作。首先，介绍了智能家居系统的整体框架结构，并提出一种创新的内网、外网相结合的控制方式。然后，结合软件的需求分析和业务流程对系统的功能模块进行划分，通过模块划分图宏观说明各个模块之间的关系，然后再设计各模块具体的运行流程。最后，对通信中所使用的数据格式进行了规定。

# 第四章 客户端软件实现

## 4.1 用户登录模块

### 4.1.1 类关系图

登录功能的实现需要依赖数据存储模块和网络通信模块中的部分功能，使用数据存储模块可以对用户信息进行增加、删除、修改、查询操作，对网关和服务器的IP地址进行修改操作。使用网络通信模块将用户的登录请求组装成一定格式的报文后，调用发送线程把数据发送出去，接收线程收到反馈后利用消息机制通知系统主线程进行更改

UI操作。

登录模块类关系图如图4.1所示。



**图4.1** **登录模块类关系图**

类关系图及各个类的含义解释如下：

##### （1) LoginActivity 为登录界面类，它与发送线程SendThread 类是聚合关系，

SendThread类不依附于LoginActivity类而存在；与LoginHandler是组合关系，一旦

LoginActivity类销毁，那么LoginHandler类也随之被销毁。

##### （2) SendThread为发送线程类，作为系统的一个子线程而存在。首先使用ProtocolUtil

协议工具类生成具体的通信数据格式，然后调用SocketUtil 通信工具类中的Socket，

OutputStream属性和write(byte[] data)方法进行数据写出操作。

##### （3) User 为用户实体类，封装了用户的账号、密码、账号状态属性。该类在

LoginActivity类和DBOperateUtil类之间作为对象的引用而存在。比如需要保存用户信息时，将包含账号密码信息的User对象作为参数传入DBOperateUtil类中存储用户信息的方法中，然后在该方法中取出User对象的属性值存入数据库。

(4) LoginHandler为主线程和子线程之间传递消息的类，该类的对象绑定在主线程中，也就是LoginActivity类所处的线程。负责接收数据的ReceiveThread子线程收到服务器/网关的反馈数据后，通过Message 对象将反馈的数据传回至LoginHandler 中的

handleMessage()方法，对数据处理后，通知LoginActivity类进行更改UI操作。

(5) DBOperateUtil为数据库操作工具类，是对操作方法的封装。在登录模块用到的主要方法是存储用户、查询用户、查询内网/外网IP地址、设置内网/外网IP地址。

### 4.1.2 界面实现

登录界面中的控件主要有账号输入框、密码输入框、记住密码复选框、内网登录按键、外网登录按键。界面整体上采用的是线性布局结构，在线性布局中又嵌套有子线性布局，界面上所有的控件全部由布局文件定义[23]。

界面布局资源文件名为activity\_login. xml，位于res/layout目录下，布局文件中包含的控件类型有：文本控件TextView，可编辑文本控件EditText，复选框控件CheckBox，普通按钮控件Button。写好布局文件后，在LoginActivity类的onCreate（）函数中设置该布局文件，代码为：

@Override

protected void onCreate(Bundle savedInstanceState) {// 重写 onCreate()方法super.onCreate(savedInstanceState); // 调用父类的 onCreate()方法setContentView(R.layout.activity\_login); // 设置登录界面的布局文件

}

登录界面最终的显示效果如图4.2所示。



**图4.2** **登录界面**

EditText userNameEdit = (EditText) findViewById(R.id.userNameEdit);

String username = usernameEdit.getText().toString();

要想获得用户输入的账号和密码内容，首先需要从布局文件中通过id查找该控件对象，然后调用控件中的方法即可获取用户输入的信息。例如，从登录界面获得账号控件usernameEdit中的内容，最终的字符串username就是用户输入的账号，代码如下：

界面中的登录按键用来完成用户的登录操作，点击后，向服务器/网关发送登录信息，要想实现这个功能，需要为Button按键添加事件监听器，这里添加的是单击事件监听器，并设定相应的监听函数。例如，给内网登录按键lan\_Button添加监听器和监听函数，代码如下：

Button lan\_Button= (Button) findViewById(R.id. lan\_Button); lan\_Button.setOnClickListener(new OnClickListener(){

public void onClick(View v){

……// 监听函数

}

});

### 4.1.3 快捷登录

为了使软件具有良好的用户体验，应该在用户登录界面提供记住密码这项功能，当用户选中这项功能时，如果该账号密码通过验证并登录成功，则程序应保存该账号和密码，以便用户下次使用时不必再次输入，使软件操作更加便捷。

由于存在一个用户有多个账号的可能性，在软件设计时就必须保存每一个登录成功的账号和密码，当用户想切换账号登录时，只需在用户名列表中选中某一个账号即可，不必再次输入。当用户保存了多于一个的账号信息时，软件退出后再次启动会自动填充最近一次登录成功的账号和密码，并将其设为活动账号。

要想实现上述功能，就需要用到Android平台自带的SQLite轻量级数据库，将登录成功的信息存入数据库的表中，再次运行软件时从数据库的表中将登录信息读取出来填充到对应的位置，即可实现记住密码的功能。

保存账户信息时，首先用当前输入的账户名称在数据库中进行查询，若没有，则插入账号密码；若有，则用本次输入的密码替换原来的密码。读取账户信息时，循环取出所有的账户，当某一条记录的state属性为1时，将该条记录设置为活动账户，并显示在用户的登录界面中。代码如下：

boolean flag = DBOperateUtil.saveUser(user);// 存储用户信息

if(flag){

……// 账户保存成功

}else{

……// 账户保存失败

}

ArrayList<User> users = DBOperateUtil.findUser();// 取出用户集合

for (int i = 0; i < users.size(); i++) {

……// 遍历用户集合

if(users.get(i).getState == 1){

……..// 该账号设置为活动账户，并在界面显示

}

}

## 4.2 软件更新模块

### 4.2.1 类关系图

软件的版本更新功能在软件开发过程中是必不可少的，因为无法保证开发出的应用程序是完美无缺的，需要在后期版本中不断的进行修复与改进。

软件版本检测事件发生在系统启动后，当检测到有新版本时，弹出对话框询问用户是否选择升级，若选择升级，则进行下载更新包的操作，并在界面实时显示下载进度状态，下载完成后进行覆盖安装。

软件更新模块类关系图如图4.3所示。



**图4.3** **软件更新模块类关系图**

类关系图及各个类的含义解释如下：

(1) LogoActivity为系统启动后的第一个界面，在这个界面对系统做一些初始化的工作。

(2) CheckVersionTask为系统的一个子线程，访问远程服务器并获得存储版本信息的xml文件，然后调用UpdateInfoParser对xml文件进行解析，将解析结果返回主界面，并自动销毁该线程。

(3) DownLoadApk是管理下载更新包的类，主要负责界面上下载进度的显示，并创建DownLoadUpdateManager子线程下载文件。

(4) InstallApk是下载完成后安装更新包的类。

软件更新功能实现的原理是客户端软件访问远程服务器中的Web容器，读取存放在该容器中有关软件版本信息的xml文件，通过解析xml文件，得到服务器端保存的软件版本号及新版本的下载地址。若服务器的版本号大于本地的版本号，则提示用户进行软件的更新，用xml文件提供的地址进行软件的下载；若服务器版本号小于或等于本地版本号，则说明不需要更新，继续运行当前版本。

由此可见，软件更新模块的实现分为客户端和服务端两个部分，服务端提供给客户端一个获取软件版本信息的端口，客户端通过该端口读取并解析版本信息的xml文件。

### 4.2.2 Web服务端

服务端的Web容器是一种服务程序，运行于服务器上，一个端口对应一个提供相[应服务的程序，这个程序负责处理客户端](http://baike.baidu.com/view/930.htm)发出的请求，例如JAVA中的Tomcat容器，

ASP的IIS或PWS都是这样的容器。本论文采用Tomcat容器，因为Tomcat是一个技术先进、性能稳定，而且免费的轻量级应用服务器[24]。

在服务器上安装并配置好Tomcat容器后，在Tomcat的webapps目录下新建一个smarthome文件夹，将有关版本信息的smarthome\_update. xml文件和后缀名为. apk的最新版本安装包放入该文件夹，启动Tomcat容器即完成服务器端的配置。

假设服务器的IP地址为192.168.0.1，Tomcat运行的端口号为8080，那么客户端的访问地址就为http: //192.168.0.1:8080/smarthome/smarthome\_update. xml。

<?xml version="1.0" encoding="utf-8"?>

<App>

<version>0.2</version>

<url>http://192.168.0.1:8080/smarthome/SmartHomeV0.2.apk</url>

<contentText>优化了用户界面</ contentText >

</App>

smarthome\_update. xml文件的代码如下：

其中，App为文档的根节点，其子节点version代表服务器端软件的版本号，url代表新版本的下载地址，contentText代表新版本软件的更新内容描述。

### 4.2.3 客户端XML文档解析

客户端软件启动后，在后台向服务器发出Http连接请求，设置连接超时为5秒，若5秒内连接不上服务器，提示用户连接服务器失败。连接成功后，读取服务器中Tomcat

容器里的xml文件，解析xml文件后，得到服务器端软件的版本号，通过对比软件的版本号，提示用户是否需要进行升级操作。实现代码如下：

// 设置访问的路径

URL url = new URL("http://192.168.0.1:8080/smarthome/smarthome\_updat e.xml");

// 获得 Http 连接

HttpURLConnection conn = (HttpURLConnection) url.openConnection(); InputStream is = conn.getInputStream();// 获得 xml 文件的输入流UpdateInfo info = UpdateInfoParser.getUpdataInfo(is); // 解析 xml 文件String server\_versionName = info.getVersion();// 得到服务器端版本号

// 得到客户端版本号

String client\_versionName = VersionUtil.getVersionName(context); float clientVersion = Float.parseFloat(client\_versionName);

float serverVersion = Float.parseFloat(server\_versionName); if(serverVersion <= clientVersion){ // 无需升级

……// 继续运行原有程序

}else {// 有新版本

……// 弹出对话框提示用户进行升级

}

从服务器端得到版本xml文件后，需要对其进行解析，本论文采用Pull解析器对文件进行解析，通过循环判断输入流的开始标记，即可得到xml文件中包含的版本号、更新包的下载地址、更新内容这三种信息，具体实现代码如下：

XmlPullParser parser = Xml.newPullParser(); // 定义 Pull 解析方式

parser.setInput(inputStream, "utf-8"); // 设置解析的数据源

int type = parser.getEventType();

UpdateInfo info = new UpdateInfo();// 封装版本信息的实体

while (type != XmlPullParser.END\_DOCUMENT) {// 进行循环解析

switch (type) {

case XmlPullParser.START\_TAG:

if ("version".equals(parser.getName())) {// 得到版本号

info.setVersion(parser.nextText());

}else if ("url".equals(parser.getName())) {// 得到更新包的下载地址

info.setUrl(parser.nextText());

}else if ("content".equals(parser.getName())) {// 得到更新内容

info.setContent(parser.nextText());

}

break;

}

type = parser.next();

}

当客户端检测到有新版本时，弹出对话框提示用户，这里的对话框使用的是Android中的AlertDialog控件，这个控件依附于LogoActivity类而存在。对话框中可以设置标题、内容、确定或取消按键，提示用户版本升级的界面显示效果如图4.4所示。



**图4.4** **提示用户版本升级界面**

若用户在上述界面中点击取消按键，则跳过升级模块，进入软件的登录界面；若用户选择确定按键，则开始从服务器下载新版本文件，界面显示效果如图4.5所示。



**图4.5** **下载新版本文件界面**

新版本软件下载完成后，询问用户是否进行覆盖安装，并且旧版本中的用户数据将被保存，提示安装新版本界面显示效果如图4.6所示。



**图4.6** **提示新版本安装界面**

## 4.3 网络通信模块

### 4.3.1 类关系图

网络通信模块是客户端软件中各个模块与网关/服务器进行交流沟通的接口，它负责处理各个模块的联网事务，比如接收其他模块的数据，组装成一定的协议格式后发送给网关/服务器，或者从网关/服务器接收数据进行解析后向其他模块发送动作指令。

网络通信模块类关系图如图4.7所示。



**图4.7** **网络通信模块类关系图**

类关系图及各个类的含义解释如下：

(1) SocketUtil为封装输入输出流的工具类，当有数据要写出或写入时，就需要调用该类中的write（）和read（）方法。

(2) ClientSocket为与服务器/网关建立Socket连接的类，使用IP地址和PORT端口号进行连接。

(3) ProtocolUtil为封装通信协议的类，需要使用某条协议时直接调用即可。

(4) SendThread和ReveiveThread为发送子线程和接收子线程，专门负责用户指令的发送和反馈数据的接收，与ClientSocket类和ProtocolUtil类属于依赖关系，调用被依赖类中的某些方法而得以完成这个类的一些职责。

### 4.3.2 通信接口

Android系统在网络编程方面提供了非常好的支持，针对TCP的传输方式，Android提供了Socket类和ServerSocket类，针对UDP传输方式，Android提供了DatagramSocket类和DatagramPackage类。Socket是比较底层的网络编程方式，是建立其他应用协议的基础。通过创建Socket对象，就可以打开输入输出流来进行通信。

#### 1. 建立连接

本论文的网络通信模块采用基于TCP（面向连接）的Socket编程，分为客户端和服务器端两个部分。Android应用软件属于客户端部分，服务器端可以是系统的专有服务器或家庭网关。由于TCP是面向连接的通信方式，两者之间需要建立一个Socket连接，因此，客户端创建一个Socket并向服务器发出连接请求后，通过此连接即可与服务器端进行通信，代码如下：

String IP = "192.168.0.110"; // 服务器的 IP 地址int PORT = 54321; // 服务器侦听的端口号Socket socket = new Socket();

SocketAddress socketAddress = new InetSocketAddress(IP, PORT);

socket.connect(socketAddress, 5000); // 建立连接，超时为 5 秒

#### 2. 输入输出流的封装

Socket连接建立成功后，打开输入输出流即可进行通信，为了降低程序的耦合性，将输入输出流的方法封装在一个类中，使用时生成类的对象调用方法即可，代码如下：

public void write(byte[] data) throws IOException {// 输出流

os = socket.getOutputStream(); os.write(data);

os.flush();

}

public byte[] read() throws IOException {// 输入流

is = socket.getInputStream();// 得到输入流

byte[] buffer = new byte[1024]; // 创建字节缓冲数组

int n = is.read(buffer); // 读入缓冲区的字节总数

if (n != -1) {

byte[] data = new byte[n]; System.arraycopy(buffer, 0, data, 0, n); return data;

}

}

#### 3. 发送和接收子线程

发送线程和接收线程在用户登录成功后创建，线程内部通过标志位来控制线程的结束。也就是说，在应用软件退出之前，程序中一直存在三个线程：一个主线程，一个发送线程和一个接收线程。

主线程的作用主要是响应用户的操作，为了提高用户体验，将比较耗时的网络操作放入其他子线程中，这样就不会阻塞主线程，不然在网络环境较差的情况下，程序会给用户造成“假死”的状态，降低程序的友好使用性。而且在Android4.0以后，在主线程执行耗时的网络操作时，程序会报出android. os. NetworkOnMain ThreadException异常。用户在使用客户端软件时，需要联网查询的数据很多，并且次数较为频繁，因此，

为了避免程序不断的创建和销毁线程浪费系统资源，在子线程创建后使用循环的方式来防止线程自动结束，这样，在接收和发送数据时就能重复利用该子线程。当需要结束子线程时，将其循环的标志位设置为false，线程即可销毁。

因为发送线程在整个程序运行过程中只存在一个实例，所以创建发送线程的对象时，要使用设计模式中的单例模式。发送线程在每发完一次数据后就进入wait（）状态，程序需要再次发送数据时，使用notify（）唤醒该发送线程，然后再进行数据的发送。线程在等待状态下不排队占用系统的CPU时间片，可以有效的减少系统资源的浪费。

private SendThread() {}// 构造方法私有化

public static SendThread getInstance() {// 单例模式

if (sendThread == null) {

sendThread = new SendThread(); sendThread.start();

}

return sendThread;

由于发送线程与接收线程的设计思路类似，故只介绍发送线程，代码如下：

}

@Override

public void run() { while (isRun){

try {

wait();//等待状态if(isRun){// 发送数据

socketUtil.write(ProtocolUtil…….);

}

}catch(Exception e){

// 发送数据异常时的处理

……

}

}

}

## 4.4 界面操作模块

### 4.4.1 类关系图

界面操作模块是直接面向用户的，显示内容所见即所得，界面操作使用逐级递进的方式，也就是下一级界面是建立在上一级界面的基础之上的，不能跨越上一级界面而直接进入下一级界面。在用户登录成功后，首先显示的是系统的主界面，然后根据主界面的按键选项进入不同的操作界面。

由于设计采用了长按和单击两种操作方式，故界面显示的时候会分为编辑界面和控制界面。在编辑界面只能进行添加、删除、修改操作，在控制界面只能进行控制操作。

界面操作模块类关系图如图4.8所示。



**图4.8** **界面操作模块类关系图**

类关系图及各个类的含义解释如下：

(1) MainActivity为系统的主界面显示类，该界面监听单击和长按事件，且与下一级界面有强烈的包含关系，组合类负责被组合类的生命周期，即主界面销毁后，各个子界面也将随之销毁。

(2) RoomListActivity为“我的房间”控制界面，由主界面单击“我的房间”按键跳转。该界面只监听用户的单击事件，选中房间单击后跳转至具体的房间界面

RoomActivity，显示本房间中的所有设备，然后单击某个设备进入其控制操作界面

DeviceActivity。

(3) RoomListEditActivity为“我的房间”编辑界面，由主界面长按“我的房间”按键跳转。该界面只监听用户的长按事件，在这里可以添加、删除、修改某个具体房间，选中房间长按后进入该房间的设备编辑界面RoomEditActivity，用于修改房间中的设备。

(4) DeviceListActivity为“设备列表”界面，由主界面单击“设备列表”按键跳转。该界面只监听用户的单击事件，显示整个房间中的所有设备及其所处位置，单击后可进入相应设备的控制界面DeviceActivity。

(5) SystemManageActivity为“系统管理”界面，由主界面单击“系统管理”按键跳转。该界面只监听用户的单击事件，包含四个子界面，UpdateUserActivity 为“修改

账户密码“界面，UpdateGatewayActivity 为”修改家庭网关密码“的界面，

OperationManualActivity为查看“操作手册”界面，VersionInformationActivity为查看“版本信息”界面。

### 4.4.2 界面实现

用户登录成功后，进入系统的主界面，该界面分为我的房间、我的场景、设备列表、安防监控、系统管理五大部分。房间界面的设计采用线性布局嵌套网格视图（GridView）的形式，设定每行显示两个网格，网格中的数据由Java代码填充。

主界面的布局资源文件设置方式与登录界面一致，区别在于登录界面显示的数据是由布局文件中的组件添加的，即xml添加组件的方式。而房间界面的布局文件中只指定了GridView组件的显示形式，没有指定具体的显示数据，这些数据是在MainActivity类中使用Java代码动态添加的。

为了使界面更加直观、生动，GridView组件中的每一个网格都显示一组图片和文字。要想达到这种显示效果，需要生成一个HashMap对象，以键值对（Key-Value）的形式添加图像资源和文字资源，然后将该HashMap对象存入一个ArrayList集合中去，

ArrayList集合中的一个元素即代表着一个网格中的数据，最后用SimpleAdapter将集合显示出来。代码如下：

ArrayList <HashMap<String, Object>> arrayList = new ArrayList <HashMap <String, Object>> (); // 新建一个装有 HashMap 键值对的集合对象

HashMap<String, Object> map = new HashMap<String, Object> (); map.put("ItemImage", R.drawable.picture); // 添加图像资源map.put("ItemText", " 客 厅 "); // 添 加 文 字 资 源arrayList.add(map); // 将 map 对象添加到集合中

SimpleAdapter adapter = new SimpleAdapter (RoomListActivity.this, arrayList, R.layout.activity\_gridview\_item, new String [] {"ItemImage", "ItemText"}, new int[]

{ R.id. ItemImage, R.id.ItemText });

gridview.setAdapter(adapter); // 将适配器添加到网格组件中并显示

gridview.setOnItemClickListener(new ItemClickListener());// 添加单击消息处理

“我的房间”界面与主界面显示方式类似，系统主界面和“我的房间”界面最终的显示效果如图4.9所示。



**图4.9 系统主界面和“我的房间”界面**

if (state == 0x64) {// 如 果 设 备 状 态 值 为 100 iButton.setImageResource(R.drawable.ds\_lighton); //设置图像按钮为灯亮图片

}

else if (state == 0x01) {// 如 果 设 备 状 态 值 为 1 iButton.setImageResource(R.drawable.ds\_lightoff); //设置图像按钮为灯灭图片

}

当用户想改变家庭中某个设备的状态时，需要进入到其相关的控制界面类DeviceActivity进行操作。控制界面的显示采用线性布局嵌套图像按钮的形式，用户点击图像即可完成改变设备状态的功能。在xml布局文件中，只定义了图像按钮的位置，没有添加图片，因为设备的状态需要根据网关发来的数据确定，所以，图像按钮的图片在程序中由Java代码动态添加。代码如下：

控制界面最终的显示效果如图4.10所示。



**图4.10** **控制界面效果图**

## 4.5 数据存储模块

### 4.5.1 类关系图

智能家居客户端系统中需要存储的信息量较小，不需要大的数据库，所以本应用程序采用Android平台内置的SQLite数据库，它可移植性好、易使用，高效且可靠性高。

数据存储模块主要用于对程序中的数据进行增加、删除、修改、查询操作，数据存储模块类关系图如图4.11所示。



**图4.11 数据存储模块类关系图**

类关系图及各个类的含义解释如下：

(1) SQLiteOpenHelper类为Android平台提供的数据库助手类，在该类中有两个方法：onCreate()和onUpgrade()，用于对SQLite数据库进行创建和更新操作，提供了两个方法getWritableDatabase()和getReadableDatabase()来获得数据库的操作对象。

(2) DatabaseHelper类继承自SQLiteOpenHelper类，重写了SQLiteOpenHelper类中的onCreate()和onUpgrade()方法。同时，它通过调用getReadableDatabase()方法得到的返回值依赖于SQLiteDatabase类，获得数据库的操作对象。

(3) SQLiteDatabase类是Android平台提供的数据库操作类，提供一些方法，可以创建、删除、执行SQL指令、完成常规的数据库管理任务，如execSQL（）可以执行一般的SQL语句，insert（）可以给数据表中插入表项，update（）可以更新表项等。

(4) DBService类为操作数据库的业务逻辑类，用于与数据库直接进行交互，该类与SQLiteDatabase类和DatabaseHelper类为单向关联关系。

### 4.5.2 表的基本结构

数据库中需要存储用户信息表、IP地址表、设备表、房间表。用户信息表用于保存用户登录时的账号和密码，表结构如表4.1所示。

**表4.1** **用户信息表（USER）**

| 序号 | 字段 | 字段含义 | 类型 | 长度 | 为空 | 主键 | 说明 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | 编号 | INTEGER | 1 | N | 是 | 自动增长 |
| 2 | username | 账号 | VARCHAR | 32 | N |  |  |
| 3 | password | 密码 | VARCHAR | 32 | N |  |  |
| 4 | state | 状态标志 | INTEGER | 1 | N |  | 1: 活动账号 |

IP地址表用于保存登录时的内网IP地址和外网IP地址，表结构如表4.2所示。

**表4.2** **IP地址表（ADDRESS\_IP）**

| 序号 | 字段 | 字段含义 | 类型 | 长度 | 为空 | 主键 | 说明 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | 编号 | INTEGER | 1 | N | 是 | 自动增长 |
| 2 | ip | IP 地址 | VARCHAR | 32 | N |  |  |
| 3 | type | 接入方式 | VARCHAR | 32 | N |  | LAN/WAN |

设备表用于保存智能家居设备的信息，表结构如表4.3所示。

**表4.3** **设备表（DEVICES）**

| 序号 | 字段 | 字段含义 | 类型 | 长度 | 为空 | 主键 | 说明 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | 设备编号 | INTEGER | 16 | N | 是 | 由网关指定 |
| 2 | name | 设备名称 | VARCHAR | 32 | N |  |  |
| 3 | location | 设备位置 | INTEGER | 16 | Y |  |  |
| 4 | status | 设备状态 | INTEGER | 16 | Y |  |  |

房间表用于保存用户房间的结构，表结构如表4.4所示。

**表4.4** **房间表（ROOMS）**

| 序号 | 字段 | 字段含义 | 类型 | 长度 | 为空 | 主键 | 说明 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | id | 房间编号 | INTEGER | 16 | N | 是 | 由网关指定 |
| 2 | name | 房间名称 | VARCHAR | 32 | N |  |  |
| 3 | type | 房间类型 | INTEGER | 16 | N |  |  |

## 4.6 本章小结

本章主要介绍了智能家居客户端系统在Android平台的编程实现方法。首先，在每个功能模块开始编写代码实现之前，阐述了模块中各个类的关系及类的含义；接着，对于有界面的模块，介绍了界面层的实现方法，并展示代码实现后的最终效果截图；最后，对每个模块需要实现的功能进行代码实现。

# 第五章 客户端软件测试

本章主要进行系统验证和测试。首先介绍系统测试环境的搭建，然后对系统的功能模块进行验证。通过设计丰富的测试用例，对系统的正确性、稳定性和健壮性进行验证。

## 5.1 测试环境

搭建测试环境是实施测试过程中非常重要的一个阶段，测试环境搭建的是否合适在很大程度上可以决定测试结果的准确性和真实性。测试环境包括软件环境和硬件环境，软件环境包含测试软件运行的操作系统，开发工具的版本号等；硬件环境包括测试所必须有的计算机、Android平台的手机、配套数据线等设备。

本系统测试所需的软件环境如表5.1所示，硬件环境如表5.2所示。

**表5.1** **测试所需的软件环境**

| 设备名称 | 软件环境 |
| --- | --- |
| 计算机 | Windows 7 操作系统，JDK1.6, Android SDK2.3.3, Eclipse SDK3.7.2,  Tomcat6.0 |
| 手机 | Android OS 2.3.3(480 × 800)，Android OS 4.2.2(720 × 1280) |

**表5.2** **测试所需的硬件环境**

| 设备名称 | CPU | 内存 | 硬盘 |
| --- | --- | --- | --- |
| 计算机 | 2GHz | 2GB | 160G |
| 手机 | 1GHz | 1GB | 16GB |

## 5.2 测试方法及结果分析

目前，智能家居客户端系统已进入测试和修复阶段。本系统的测试部分主要分为功能测试和性能测试。功能测试主要为了验证系统各个功能部分是否完备；性能测试主要为了评估软件在Android平台中运行时的性能状况。

### 5.2.1 功能测试

智能家居客户端系统主要涉及到登录、更新、网络通信、控制、数据存储这五大功能模块。针对以上各个功能点设计主要的测试用例，对系统进行测试。

#### 1. 登录功能测试

登录模块是用户使用本软件的先决条件，只有通过身份验证的用户，才能正常使用软件中的各个功能。登录验证时，软件向服务器/网关提交用户输入的账号和密码，然后从服务器/网关处获得验证结果。测试过程中可以把用户的请求报文和回应报文输出到控制台，查看报文是否正确。主要测试用例如表5.3所示。

**表5.3** **登录模块主要测试用例**

| 功能描述 | 测试步骤 | 预期结果 | 测试结果 |
| --- | --- | --- | --- |
| 账号或密码为空 | 点击登录 | 弹出消息提示账户或密码不能  为空 | 通过 |
| 输入错误账号或密码 | 点击登录 | 弹出消息提示账户或密码错误 | 通过 |
| 输入正确账号和密码 | 点击登录 | 弹出消息提示用户登录成功 | 通过 |
| 记住密码 | 用户选择记住密码且  至少成功登录过一次 | 账号密码自动填充 | 通过 |

#### 2. 软件更新功能测试

软件更新模块主要测试的是网络通信模块是否能正常的获取服务器的版本信息文件，用户选择更新时是否能从服务器正确的下载更新包文件，以及下载完成后是否能正确的安装。测试的时候服务器配置的版本号应该高于客户端的版本号，然后在客户端启动软件自动请求更新，从服务器下载文件后进行安装。主要测试用例如表5.4所示。

**表5.4 软件更新模块主要测试用例**

| 功能描述 | 测试步骤 | 预期结果 | 测试结果 |
| --- | --- | --- | --- |
| Tomcat 服务未启动 | 运行软件 | 弹出消息提示连接服务器失败 | 通过 |
| 服务器版本号高于客  户端版本号 | Tomcat 服务已启动 | 弹出对话框询问用户是否升级 | 通过 |
| 下载更新包 | 用户选择升级 | 从服务器下载文件并显示进度  条 | 通过 |
| 安装更新包 | 用户选择安装 | 软件覆盖安装成功，并且用户  以前的数据还存在 | 通过 |
| 服务器版本号低于或  等于客户端版本号 | Tomcat 服务已启动 | 没有任何提示，直接进入登录  界面 | 通过 |

#### 3. 网络通信功能测试

网络通信模块在整个系统中处于非常关键的地位，如果通信模块不能正常工作，那么整个系统都会出现问题，因此，网络通信模块的测试显得尤为重要。

网络通信模块的测试主要包括通信报文是否能正确的生成、解析，发送和接收模块能否正确的发送和接收消息。测试方法是将生成的报文和接收的报文在控制台打印出来，查看各个字段的内容是否正确。主要测试用例如表5.5所示。

**表5.5 网络通信模块主要测试用例**

| 功能描述 | 测试步骤 | 预期结果 | 测试结果 |
| --- | --- | --- | --- |
| 请求报文生成 | 控制台打印生成的请  求报文 | 报文中各个字段均正确 | 通过 |
| 回应报文解析 | 控制台打印收到的回  应报文 | 报文中各个字段均正确 | 通过 |
| 数据发送 | 发送请求报文 | 服务器/网关收到正确的报文 | 通过 |
| 数据接收 | 接收回应报文 | 客户端收到正确的报文 | 通过 |

#### 4. 界面操作功能测试

界面操作模块主要验证的是用户发送的指令是否能正确的执行，客户端收到回应后界面是否有相应的变化。主要测试用例如表5.6所示。

**表5.6 界面操作模块主要测试用例**

| 功能描述 | 测试步骤 | 预期结果 | 测试结果 |
| --- | --- | --- | --- |
| 添加房间 | 选择房间类型，填写房  间名称 | 添加房间成功，界面动态刷新 | 通过 |
| 删除房间 | 长按房间名称弹出确  定删除提示 | 删除房间成功，界面动态刷新 | 通过 |
| 修改房间 | 重新填写房间名称 | 修改房间成功，界面动态刷新 | 通过 |
| 添加设备 | 将设备设置到某个房  间中 | 添加设备成功，在对应的房间  有该设备显示 | 通过 |
| 删除设备 | 长按设备名称弹出确  认删除提示 | 删除设备成功，列表中该设备  消失 | 通过 |
| 修改设备 | 重新填写设备名称，重  新设置设备位置 | 修改设备成功，界面动态刷新 | 通过 |

#### 5. 数据存储功能测试

数据存储模块主要验证的是需要持久化的信息是否能正确的保存在系统自带的

SQLite数据库中，测试方法是将要存储数据的指令执行一遍，然后查看数据库中是否有对应的记录被添加。主要测试用例如表5.7所示。

**表5.7 数据存储模块主要测试用例**

| 功能描述 | 测试步骤 | 预期结果 | 测试结果 |
| --- | --- | --- | --- |
| 用户信息表 | 某账号第一次登录并  选择记住密码 | 用户信息表中新增一条记录 | 通过 |
| IP 地址表 | 用户修改网关或服务  器的 IP 地址 | 修改 IP 地址表中的记录 | 通过 |
| 设备表 | 添加一个设备，并设置  其名称、位置 | 设备表中新增一条记录 | 通过 |
| 房间表 | 新建一个房间，并设置  其类型、名称 | 房间表中新增一条记录 | 通过 |

在对软件的功能测试中，对系统主要涉及到的登录、更新、网络通信、控制、数据存储这五大功能模块进行了测试，设计的测试用例全部通过，证明客户端软件已实现所有设计的功能。

### 5.2.2 性能测试

系统性能测试主要目标是测试客户端软件运行时在CPU占用率和物理内存占用方面的性能。

由于CPU和内存占用是一个实时变化的状态，而市面上还没有具体的哪款Android应用能做到实时监控CPU和内存占用并使用LOG日志保存，因此，考虑到Android的底层框架是基于Linux的平台，所以，可以通过Linux的资源监控命令来实现对Android平台的资源实时监控。

使用这种测试方法需要具备以下两个前提条件：

(1)获取被测试客户端的root权限。因为涉及到底层的Linux命令，所以需要读取或执行相应的文件。

(2)获取应用程序的安装包名。因为需要监控的是具体的某个应用资源占用的情况，所以需要知道这个应用对应的“安装的包名”是什么，这就需要使用Android开发环境

中DDMS模块中的Devices中查看，当启动被测软件时，其主应用程序的安装包名一般就会在列表的最下端出现，如图5.1所示。



**图5.1** **模拟器的应用程序列表**

通过上述截图可以看到，智能家居系统客户端应用软件的主程序包名为：

wlx. smarthome. activity。

具备以上两个前提条件后，再使用Linux命令进行监控，操作流程如下：

#### 1. 进入Linux系统的底层，命令：adb shell；

2. 清除系统缓存，命令：echo 3> /proc/sys/vm/drop\_caches；

#### 3. 每3秒打印一次资源利用情况，命令：top -d 3 | grep wlx. smarthome. activity，如图5.2所示。



**图5.2** **程序运行占用资源情况**

在Linux下，top命令经常用于分析程序的性能，使用该命令能实时显示系统中各进程占用资源的情况，其功能类似于[Windows](http://www.baike.com/wiki/Windows)平台中的[任务管理器](http://www.baike.com/wiki/%E4%BB%BB%E5%8A%A1%E7%AE%A1%E7%90%86%E5%99%A8)。由于上图没有显

示每一列对应的标题，因此可以单独通过top命令了解到每列数据所代表的含义，如图

## 5.3 所示。



**图5.3** **top指令参数含义**

在这里，只解释一下要用到的两个参数：

|--CPU%: CPU占用率

|--RSS：实际占用的物理内存数，单位KB

在实际测试时，可以针对不同的业务，打印出不同的标签，用于区别现在从事的那个业务，并为后期分析各业务模块中CPU和内存的占用以及对比使用。比如，用户选择更新时，客户端从服务器下载更新包的资源利用情况如图5.4所示。



**图5.4** **下载业务占用资源情况**

分析以上数据可知，系统在开始下载更新包之后，CPU的占用率维持在50%左右，实际占用的物理内存数维持在29M左右，属于正常范围内，通过这些数据说明软件达到了当初设计时的性能要求。

以上就是对智能家居客户端应用的评测数据的获取，然后将日志结果复制到excel

的表格中，并画出相应的数据表。最后，可以根据得到的图表进行性能分析。

## 5.3 本章小结

本章主要介绍了系统测试的方法。首先，阐述了系统测试所需要的测试环境；然后，

将测试分为功能测试和性能测试两部分。功能测试通过设计测试用例的方式完成，性能测试主要测试的是软件的CPU和内存占用率；最后，对测试的结果进行了分析总结。

# 第六章 总结与展望

## 6.1 总结

智能家居这一概念进入中国已经有数十年的时间了，目前，智能家居系统的发展也已进入了一个数字化、网络化的阶段。随着Android系统和移动互联网技术的普及，用户不需要亲自在家操作，便可以通过外观小巧、方便携带的智能移动设备在任何时间、任何地点对家中所有的家居设备完成远程控制和集中管理。通过将普通住宅转变为智能化住宅，可以给用户的生活带来前所未有的便捷。

本论文以Android平台作为研究对象，结合具体科研项目和实际的功能需求，提出了一个以Android作为控制终端系统的智能家居客户端应用软件设计，并对其中关键的通信技术、存储技术和多线程技术做了深入研究。经反复测试，应用程序可以通过家庭内部的无线网络实现对家居设备的控制，也可以通过连接Internet上的服务器实现对家居设备的远程控制。同时，采用此种控制方式的软件具有通用性，易于移植，能在带有此类操作系统的智能平台上安装和使用，市场应用价值高，易于推广。

本论文的主要工作体现在以下几个方面：

#### 1. 研究当今几种主流的智能操作系统，设计并实现了一种基于当今最流行的

Android操作系统的智能家居终端通用控制程序。

#### 2. 对客户端软件中所涉及到的通信技术、数据存储技术、XML解析技术、多线程模型、界面布局技术进行研究，分析了每种技术中几种不同实现方式的特点，并根据这些特点决定软件实现时所使用的方式。

3. 介绍了整个智能家居系统的总体框架和控制方式，智能家居系统由客户端、服务器、家庭网关、前端控制器四大部分组成。客户端的控制方式采用了创新的内网、外网两者相结合的方式，使用户家庭在没有接入互联网的情况下也可以控制智能家居设备成为了可能。

4. 通过客户端软件的需求分析，确定程序应该划分的功能模块，并采用模块化的设计思想，使系统条理清楚，结构分明，并为后期的维护和扩展带来很大的便利。

5. 通过绘制模块中的类关系图，从宏观上弄清楚各个类之间的关系，再从单个类入手，完成具体功能的设计实现，并展示最终显示效果。

6. 通过设计测试用例对整个系统的功能进行测试，并对软件的资源占用情况做了

实时监视，最终表明该软件的设计方案合理实用，达到了预期的效果。

总之，本论文实现了Android平台在智能家居客户端系统中的应用，从理论和实践两个方面论述了该方案的可行性及应用价值。

## 6.2 展望

本系统虽然在测试过程中未发现问题，但还是要经受住家居实际应用的检验。而且智能家居系统还有很多相关技术需要去研究实现，使整个系统变得更加完善。接下来需要在以下方面做进一步的研究：

#### 1. 对软件的界面进行进一步的优化，使用户操作更加简便。

#### 2. 选择适当的网络安全协议和数据加密算法，对以太网中的数据和SQLite数据库中的数据进行加密处理。

#### 3. 在客户端软件中加入安防和视频监控功能。

参考文献

[1] 牛勇. 浅谈建筑智能化与智能家居[J]. 建材技术与应用, 2007(9): 45-46.

[2] 唐荣霞. 智能家居控制系统设计与实现[D]. ft东: ft东大学, 2009.

[3] 蔡沁蓉. 智能家居让生活更美好[J]. INTELLIGENT BUILDING, 2010(2): 41-42.

[4] 龙帆. 基于ZigBee的智能家居网关系统的研究与实现[D]. 湖南: 湖南大学, 2010. [5] [王朝华](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e7%8e%8b%e6%9c%9d%e5%8d%8e&amp;code=27667285%3B11230473%3B06749608%3B), [陈德艳](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e9%99%88%e5%be%b7%e8%89%b3&amp;code=27667285%3B11230473%3B06749608%3B), [黄国宏](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e9%bb%84%e5%9b%bd%e5%ae%8f&amp;code=27667285%3B11230473%3B06749608%3B). 基于Android的智能家居系统的研究与实现[J]. 计算机技术与发展, 2012(06): 225-229.

[6] 梁晓梅. 智能家居系统设计研究[J]. 科技风2012(18): 150.

[7] 祁志强. 智能家居的现状及发展趋势[J]. INTERLLIGENT BUILDING 2008(12): 43-44.

[8] [张云勇](http://search.cnki.net/Search.aspx?q=author%3A%e5%bc%a0%e4%ba%91%e5%8b%87), [房秉毅](http://search.cnki.net/Search.aspx?q=author%3A%e6%88%bf%e7%a7%89%e6%af%85). [基于物联网的智能家居技术标准化现状及发展建议](http://epub.cnki.net/grid2008/brief/detailj.aspx?filename=YDTX201015008&amp;dbname=CJFD0910)[J]. 移动通信, 2010(15): 25-29.

[9] 冯凯, 童世华. [智能家居的由来及其发展趋势](http://www.cnki.net/kcms/detail/detail.aspx?filename=XPJX201006007&amp;dbcode=CJFQ&amp;dbname=CJFD2010)[J]. 中国新技术新产品, 2010(06): 7-8.

[10] 戚振兴. 浅议我国智能家居发展[J]. 广西轻工业机械与电气, 2009(10): 63-64. [11] [张胜英](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e5%bc%a0%e8%83%9c%e8%8b%b1&amp;code=24517650%3B08387122%3B), [王烟军](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e7%8e%8b%e7%83%9f%e5%86%9b&amp;code=24517650%3B08387122%3B). 我国智能家居的现状和未来[J]. 现代商业, 2012(16): 60-62. [12] 李苗, 王伟东, 王帆. [国内智能家居现状和发展](http://www.cnki.net/kcms/detail/detail.aspx?filename=KJCB201110060&amp;dbcode=CJFQ&amp;dbname=CJFD2011)[J]. 科技传播, 2011(10): 27-28. [13] [张永刚](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e5%bc%a0%e6%b0%b8%e5%88%9a&amp;code=22648934%3B). 我国智能家居现状与问题[J]. 智能建筑与城市信息, 2012(12): 26-27. [14] 况莉莉, 李振, 衡友跃. 基于Android智能家居的探讨[J]. 淮北职业技术学院学报, 2012(03): 85-87.

[15] 李惠, 丁革建. [智能手机操作系统概述](http://www.cnki.net/kcms/detail/detail.aspx?filename=GZDN200903042&amp;dbcode=CJFQ&amp;dbname=CJFD2009)[J]. 电脑与电信, 2009(03): 67-69. [16] [王会林](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e7%8e%8b%e4%bc%9a%e6%9e%97&amp;code=07040424%3B). 嵌入式iOS和Android应用开发探讨[J]. 信息与电脑, 2012(11): 104-105. [17] Android Marks Fourth Anniversary Since Launch with 75.0% Market Share in ThirdQuarter, According to IDC. http: //www. idc. com/getdoc. jspcontainerId=prUS23771812[OL]. IDCWorldwideMobilePhoneTracker, November1, 2012. [18] 李宁. Android开发权威指南[M]. 人民邮电出版社, 2011.9.

[19] Android. htttp: //[www. android. com/](http://www.android.com/) [OL]. [20] 汪永松. Android平台开发之旅[M]. 机械工业出版社, 2010.8. [21] E2EColud工作室. 深入浅出Google Android[M]. 人民邮电出版社, 2009.8.

[22] Reto Meier. Professional Android 2 Application Development[M]. 清华大学出版社, 2010.12.

[23] Android Developers. [http: //developer. android. com/index. html](http://developer.android.com/index.html) [OL].

[24] Elliotte Rusty Harold[著], 马云, 钟萍等[译]. XML宝典[M]. 电子工业出版社, 2001.1. [25] Eclipse. org. XML Schema Infoset Model[OL]. [http: //www. eclipse. org/modeling/mdt/.](http://www.eclipse.org/modeling/mdt/) [26] 冯进, 丁博, 史殿习. XML解析技术研究[J]. 计算机工程与科学, 2009(2): 120-125.

[27] 张迪, 朱敏, 张凌立. 基于SAX的XML解析与应用[J]. 计算机与数字工程, 2008(07): 103: 107.

[28] 蔚晓娟, 冉静, 李爱华. 基于DOM的XML解析与应用[J]. 计算机技术与发展, 2007(04): 33-35.

[29] 栾咏红. 基于Android的XML解析器的分析与比较[J]. 南京晓庄学院学报, 2011(06): 98-101. [30] 林汝泽, 徐媛媛, 方凯. 基于HTTP协议的Android手机数据同步实现[J]. 信息通信, 2013(01): 96-97. [31] [孙剑](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e5%ad%99%e5%89%91&amp;code=27922996%3B27922995%3B08665607%3B27922997%3B), [董超](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e8%91%a3%e8%b6%85&amp;code=27922996%3B27922995%3B08665607%3B27922997%3B), [夏继媛](http://www.cnki.net/kcms/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20search.aspx?dbcode=CJFQ&amp;sfield=au&amp;skey=%e5%a4%8f%e7%bb%a7%e5%aa%9b&amp;code=27922996%3B27922995%3B08665607%3B27922997%3B). 基于Android的Socket(TCP/IP)在LED灯光控制系统中的应用[J]. 深圳职业技术学院学报, 2012(05): 19-24.

[32] 倪红军. 基于Android系统的数据存储访问机制研究[J]. 计算机技术与发展, 2013(06): 27-30.

[33] 闫伟, 叶建栲. 多线程技术在android手机开发中的应用[J]. 信息通信, 2012(01): 46-48.

[34] 赵亮, 张维. 基于Android技术的界面设计与研究[J]. 电脑知识与技术, 2009(29): 83-86. [35] 胡晨光. 小区智能家居系统的设计与实现[D]. 四川: 电子科技大学. 2012. [36] 孟亚辉. 浅谈软件项目开发过程中的需求分析[J]. 科技信息, 2009(11): 70-72. [37] 卢如挺. Android平台通讯管理系统网络模块的设计与实现[D]. 北京: 北京邮电大学, 2012.

攻读硕士学位期间参与的科研项目及发表的论文

发表论文：

[1]崔荔, 邵春晖, 崔磊. J2EE基于Struts和Hibernate框架的新闻发布系统分析与设计[J]. 电子设计工程, 2012(15): 44-47.

[2]邵春晖, 崔荔. 基于ARM 的六相位交通信号机设计[J]. 现代电子技术,

2012(9): 187-190.

参与项目：

1.“长安大学信息工程学院毕业设计管理系统”. 2010年10月-2011年4月.

2.“长安大学信息工程学院网络工程系新闻发布系统”. 2011年12月-2012年4月.

3."智能家居系统Android客户端开发“. 2012年7月- 2013年3月.

致 谢

岁月如歌，光阴似箭，三年的研究生生活即将结束。经历了找工作的喧嚣与坎坷，我深深体会到了写论文时的那份宁静与思考。回首三年的求学历程，对那些引导我、帮助我、激励我的人，我心中充满了感激。

首先要感谢导师张卫钢教授。三年来，我时刻体会着张老师严肃的科学态度，严谨的治学精神，精益求精的工作作风。同时，张老师还犹如慈父般对我们关怀备至，让我们感念至深。从专业课学习、论文开题以及论文撰写整个过程，都倾注了张老师大量的心血。张老师不仅在学习上对我严格要求，在日常工作和生活中也给予我非常多的帮助和意见，给予我很多的宽容和理解。我深深受益于张老师的关心、爱护和谆谆教导，让我三年来如沐春风。能师从张老师，我为自己感到庆幸。在此谨向张老师表示我最诚挚的敬意和感谢！

其次，感谢一直关心与支持我的朋友们。感谢我同门的所有兄弟姐妹，正是有你们的并肩作战，才使我们的集体更加团结富有朝气。感谢我寝室的姐妹张珊、邵欣茹，三年来，我们朝夕相处，共同进步，感谢你们给予我生活和学习中所有的关心和帮助。同窗之谊，我将永生难忘！

再次，需要特别感谢的是我的父母。父母的养育之恩无以为报，他们是我二十多年求学路上的坚强后盾，在我面临人生选择的迷茫之际，为我排忧解难，他们对我无私的爱与照顾是我不断前进的动力。

最后，感谢在百忙中抽出宝贵时间评阅本文的各位专家。

在此，向所有帮助我的人致以崇高的敬意和衷心的感谢，愿他们身体健康、工作顺利、万事如意！