长沙民政职业技术学院

**毕业设计说明书**

**题目**：移动电子简易交易系统设计与实现

|  |  |  |
| --- | --- | --- |
| **产品设计** | **工艺设计** | **方案设计** |
| **√** |  |  |

**类型：**

**学生姓名： \_\_\_\_\_\_\_ 刘雁鹏\_\_\_\_\_\_\_\_\_\_\_\_\_**

**学 号： \_\_\_\_ 1325113128 \_\_\_\_\_\_\_**

**学 院： \_\_\_\_\_\_\_\_\_ 软件学院\_ \_\_\_\_\_\_\_\_\_**

**专 业： \_\_软件技术（移动应用开发方向）\_\_\_**

**班 级： \_\_\_\_\_\_\_\_\_\_\_计应1331班\_\_\_\_\_\_\_\_\_\_\_**

**学校指导教师：\_\_\_\_\_\_\_\_\_\_李政仪\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**企业指导教师：\_\_\_\_\_\_\_\_\_\_陈九龙\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**2016年 月 日**

摘 要

移动电子商务是移动信息服务和电子商务融合的产物。随着3G的普及和4G网络的引入，人们的消费理念和商家的传统理念都在不断的转变，移动电子商务已经成为一种新型的商务模式，与人们的生活密切相关，其发展前景广阔，并对我国的经济产生深远的影响。

随着我国互联网和移动通信的迅猛发展，智能手机市场份额逐步提升，手机上网成为现代人们生活中一种重要的上网方式，人们正逐渐利用手机等移动智能终端设备进行网上支付、个人信息服务、网上银行业务、网络购物、手机订票、娱乐服务等，这种移动数据终端设备参与商业经营的移动电子商务正在迅速崛起。

近年来手机网民的规模不断扩大；截至2015年6月中国网民达6.68亿人，手机网民达5.94亿人，占网民总数88.9%。手机电商的发展成为必然，那么基于手机的移动交易平台将成为未来人们主要的交易方式。

**关键词：Android;移动交易系统;手机交易系统;交易系统**

目录

[1. 绪论 5](#_Toc438304732)

[1.1 选题原因及意义 5](#_Toc438304733)

[1.2 国内外应用现状 5](#_Toc438304734)

[2. 相关技术综述 7](#_Toc438304735)

[2.1 Android平台 7](#_Toc438304736)

[2.1.1 Android平台特性 7](#_Toc438304737)

[2.1.2 Android系统架构 8](#_Toc438304738)

[2.1.3 Android应用程序组件 10](#_Toc438304739)

[2.2 JSON数据格式 13](#_Toc438304740)

[2.3 SERVLET编程 15](#_Toc438304741)

[2.3.1 SERVLET是什么？ 15](#_Toc438304742)

[2.3.2 SERVLET的生命周期 16](#_Toc438304743)

[2.3.2 SERVLET工作原理 16](#_Toc438304744)

[2.3.3 JAVA SERVLET API 17](#_Toc438304745)

[2.4 JDBC技术 20](#_Toc438304746)

[3. 移动电子简易交易系统需求分析与设计 22](#_Toc438304747)

[3.1 系统需求分析 22](#_Toc438304748)

[3.1.1 用户定位 22](#_Toc438304749)

[3.1.2 系统需求详细说明 23](#_Toc438304750)

[3.2 系统功能模块设计 23](#_Toc438304751)

[3.2.1 商品交易模块设计 24](#_Toc438304752)

[3.2.2 用户管理模块设计 26](#_Toc438304753)

[3.2.2 买家卖家交流互动 27](#_Toc438304754)

[3.3 数据库设计 27](#_Toc438304755)

[3.4 系统总体部署 30](#_Toc438304756)

[4. 移动电子简易交易系统的实现与测试 32](#_Toc438304757)

[4.1 开发环境与工具 32](#_Toc438304758)

[4.2 商品交易模块的实现 33](#_Toc438304759)

[4.2.1 商品展示模块 33](#_Toc438304760)

[4.2.2 商品发布模块 38](#_Toc438304761)

[4.3 用户社交模块的实现 44](#_Toc438304762)

[4.3.1 留言板模块 44](#_Toc438304763)

[4.4 客户端与服务端通信 47](#_Toc438304764)

[4.5 系统测试 47](#_Toc438304765)

[4.5.1 测试环境 47](#_Toc438304766)

[4.5.2 功能测试 48](#_Toc438304767)

[5. 结论 52](#_Toc438304768)

[致谢 53](#_Toc438304769)

[参考文献 54](#_Toc438304770)

# 绪论

## 1.1 选题原因及意义

近年来，随着智能手机的普及、移动互联网的高速增长，人们越来越能感受到科技为生活带来的便利和翻天覆地的改变。智能手机渗透到了人们生活的方方面面，各种各样的手机应用层出不穷，涉及娱乐、社交、购物、游戏、搜索查询等等。根据中国互联网络信息中心最新发布的报告显示截至2015年6月，我国手机网民规模达5.94亿，较2014年12月增加3679万人，网民中使用手机上网的人群占比由2014年12月的85.8%提升至88.9%。而在智能手机系统里，Android操作系统的市场份额为53.54%并且呈现不断上升的趋势，在这种情况下，开发基于Android平台的手机应用，可以很好的满足人们的潜在需求、丰富人们的移动生活。

在所有的手机应用领域中，电子商务的应用开发尤为迅速。移动购物市场规模快速增长，预估到2016年，移动购物市场的规模将大大超过移动增值，成为移动互联网上第一大细分行业。并且随着目前人们生活水平的提高，人们再也不是简单的消费者了。有些人会制作手工艺作品，想通过出售自己的手工艺品赚钱，有些人们手上闲置比较多，想将手上的闲置换成现金。随着手机应用被人们普遍的接受，人们都能接受从应用市场安装一款解决实际问题的应用。

本文的移动电子简易交易系统是一个基于Android平台的手机应用。本应用希望解决上述人们想要自由交易的需求，让用户可以发布商品信息，包括转让物品，手工制品，艺术品。一方面提供简洁的发布模式，让用户随时随地都可以讲自己想要出售的物品拍照发布到应用上或者求购自己需要的商品。另一方面，区别于其他的交易市场，我们不提供资金链的支持，用户可以凭自己的需求选择现金交易或者以押金的形式或者通过第三方平台的担保交易。后面我们可以加入商家模块，也可以通过对买家交易数据的统计，定向的向用户推送一些比较合适的广告，实现营收。

## 1.2 国内外应用现状

近年来随着电脑技术和互联网的普及发展，也出现了很多的在线交易平台，很大程度的方便了老百姓的生活。包括大型的中和分类信息发布平台、综合电子商务平台、专一的二手交易品台和一些论坛、社区的交易板块等。而且，随着移动电商和O2O（Online to offline）的火热，过内外业出现了一批新发展的移动交易，消费应用，每个不同形态的产品都满足了一定用户的需求，但都有着自身的优点和局限之处。

在国内传统的综合分类信息发布平台有58同城，赶集网和百姓网等。它们都是大型的分类信息网站，基于本地同城的生活服务，在全国各地都建有子网站。交易板块是网站中比较重要的一个板块。它们有自己一定的优势，像用户量多、种类齐全、商品更新速度快。目前也有手机APP版。缺点是发布复杂、信息浏览复杂、用户体验不佳，有大量的虚假信息，广告和诈骗信息。

综合的店子商务平台如淘宝网，强大的淘宝平台，有靠谱的支付宝交易平台，有信用等级，有贴心的网购一键转卖功能。但是在淘宝上认证身份麻烦，发布分类信息相对复杂，给一些不熟悉的人带来很多的困惑或不便。

与国内对应的国外也有很多相似的交易模式，像亚马逊、eBay还有Craigslist。那些平台所提供的服务和模式大部分与国内的相差无几，不过在闲置的交易上采用的是竞价拍卖模式，这个模式在国内并不适用。

本文基于Android平台的简易交易系统，旨在打造属于大学生的校园交易平台，为大学生提供方便快捷发布物品，交易物品的空间。与其他在线平台不同的是，本系统目标用户明确，定位于在校大学生，建立良性循环和标准的校园交易平台，更具社交性和趣味性。

# 相关技术综述

## 2.1 Android平台

Android是Google公示于2007年11月5日推出的基于liunx内核的开源移动设备操作系统，广泛的应用在智能手机、智能电视、平板电脑或者其他便携设备上。Android平台作为一个开放性很强的平台，拥有大量丰富的应用程序和用户界面，开发者和用户可以完全发挥自己的想象力来使用它，不会存在阻碍移动产业创新的专利权等障碍。Android平台由于其高度的开放，良好的用户体验和低廉的成本，被大量的移动终端厂商所采用，如三星、HTC、华为、魅族、小米等。

### 2.1.1 Android平台特性

* 应用程序框架支持组件的重用与替换。这样我们可以把系统中不喜欢的应用程序删除，安装我们喜欢的应用程序。
* Dalvik虚拟机专门为移动设备进行了优化。Android应用程序将由Java编写、编译的类文件通过DX工具转换成一种后缀名为.dex的文件来执行。Dalvik虚拟机是基于寄存器的，相对于Java虚拟机速度要快很多。
* 内部集成浏览器基于开源的WebKit引擎。有了内置的浏览器，这将意味着WAP应用的时代即将结束，真正的移动互联网时代已经来临，手机就是一台“小电脑”，可以在网上随意遨游。
* 优化的图形库包括2D和3D图形库，3D图形库基于OpenGL ES 1.0。强大的图形库给游戏开发带来福音。在3G最为重要的的应用莫过于手机上网和手机游戏。
* 5. SQLite用作结构化的数据存储。
* 6. 多媒体支持包括常见的音频、视频和静态印象文件格式如MPEG4、H.264、MP3、AAC、AMR、JGP、PNG、GIF。
* 7. GSM电话（依赖于硬件）。
* 8. 蓝牙（Bluetooth）、EDGE、3G、WiFi（依赖于硬件）。
* 9. 照相机、GPS、指南针和加速度计（依赖于硬件）。
* 10. 丰富的开发环境包括设备模拟器、调试工具、内存及性能分析图表和Eclipse集成的开发环境插件。

### 2.1.2 Android系统架构

Android是基于liunx内核的移动设备开发平台和操作系统，主要分为4层，依次是应用程序层，应用框架层，系统运行库层和liunx内核层。如图2-1所示。

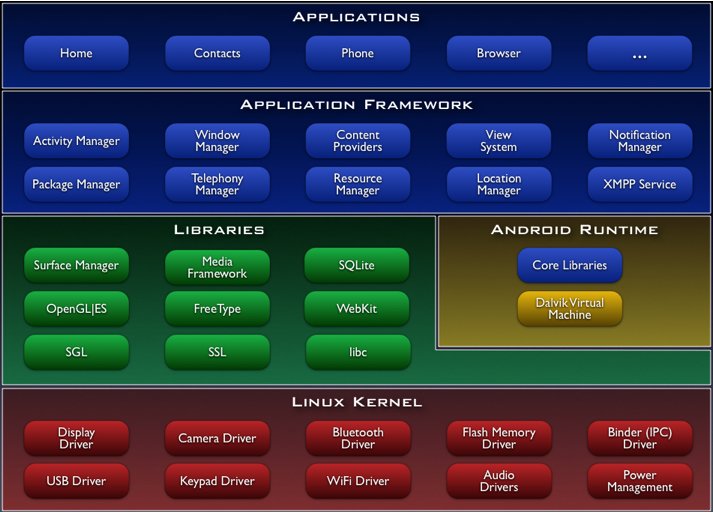


图2-1 Android系统架构图

下面分别讲解一下各个部分：

1. 程序应用

　　Android 连同一个核心应用程序包一起发布，该应用程序包包括E-mail客户端、SMS短消息程序、日历、地图、浏览器、联系人管理程序等。所有的应用程序都是用Java编写的。

2. 应用程序框架

　　开发者完全可以访问核心应用程序所使用的API框架。该应用程序框架架构用来简化组件软件的重用，任何一个应用程序都可以发布它的功能块并且任何其他的应用程序都可以使用其所发布的功能块（不过得遵循框架的安全性限制）。该应用程序重用机制使得组件可以被用户替换。

　　以下所有的应用程序都由一系列的服务和系统组成，包括：

　　1）一个可扩展的视图（Views）可以用来创建应用程序，包括列表（lists）、网络（grids）、文本框（text boxes）、按钮（buttons），甚至是一个可嵌入的Web浏览器。

　　2）内容管理器（Content Providers）使得应用程序可以访问另一个应用程序的数据（如联系人数据库），或者共享它们自己的数据。

　　3）一个资源管理器（Resource Manager）提供非代码资源的访问，如本地字符串、图形和分层文件（layout files）。

　　4）一个通知管理器（Notification Manager）使得应用程序可以在状态栏中显示客户通知信息。

　　5）一个活动类管理器（Activity Manager）用来管理应用程序生命周期并提供常用的导航回退功能。

3. Android程序库

　　Android包括一个被Android系统中各种不同组件所使用的C/C++集库。该库通过Android应用程序框架为开发者提供服务。

　　以下是一些主要的核心库：

　　1）系统C库：一个从BSD继承来的标准C系统函数库（libc），专门为基于Embedded Linux的设备定制。

　　2）媒体库：基于PacketVideo OpenCORE；该库支持录放，并且可以录制许多流行的音频视频格式，还有静态映像文件包括MPEG4、H.264、MP3、AAC、JPG、PNG。

　　3）Surface Manager：对显示子系统的管理，并且为多个应用程序提供2D和3D图层的无缝融合。

　　4）LibWebCore：一个最新的Web浏览器引擎，用来支持Android浏览器和一个可嵌入的Web视图。

　　5）SGL：一个内置的2D图形引擎。

　　6）3D libraries：基于OpenGL ES 1.0 APIs实现；该库可以使用硬件3D加速(如果可用）或者使用高度优化的3D软加速。

　　7）FreeType：位图（bitmap）和向量（vector）字体显示。

　　8）SQLite：一个对于所以应用程序可用、功能强劲的轻型关系型数据库引擎。

4. Android运行库

　　Android包括了一个核心库，该核心库提供了Java编程语言核心库的大多数功能。

　　每一个Android应用程序都在它自己的进程中运行，都拥有一个独立的Dalvik虚拟机实例。Dalvik是针对同时高效地运行多个VMs实现的。Dalvik虚拟机执行.dex的Dalvik可执行文件，该格式文件针对最小内存使用做了优化。该虚拟机是基于寄存器的，所有的类都是经由Java汇编器编译，然后通过SDK中的DX工具转化成.dex格式由虚拟机执行。

　　Dalvik虚拟机依赖于Linux的一些功能，比如线程机制和底层内存管理机制。

5. Linux内核

　　Android的核心系统服务依赖于Linux内核，如安全性、内存管理、进程管理、网络协议栈和驱动模型。Linux内核也同时作为硬件和软件栈之间的硬件抽象层。

### 2.1.3 Android应用程序组件

一个Android程序是由很多不同的组件构成的，每个组件之间使用Intent组件来进行通信。Android主要包括Activity（活动）、Broadcast（广播接收机）、Service（服务）、ContentProvider（内容提供程序）等四大组件。

下面简要的介绍一下这四大组件：

（1）Activity 活动

Activity是Android中最基本的一个构造块，基本上应用程序的流程全部会在Activity中运行，在我们的应用程序中，一个Activity就是手机中单独的一屏。每个Activity都是由很多个View控件，像Textview、Button、Checkbox等组成的用户接口，并且会对不同的事件做出响应。几乎大多数的应用都会有多个屏幕，也都是由很多个Activity组成。例如新浪微博应用有一个屏幕用来展示微博信息流，第二个屏幕用来接收消息推送，还有用户个人主页、搜索、系统设置等其他屏幕。每一个这样的屏幕就是一个Activity，这么多个Activity之间可以根据设计需求进行相应的跳转和数据传递。

每个Activity会有自己的生命周期，包括Active状态、Paused状态、Stopped状态。如图2-2所示：

图2-2 Activity生命周期图

（2）Service服务

一个Service是一段拥有长生命周期但是没有自己用户界面的程序，与很多其他操作系统的守护进程很相似，它会在后台运行，也可以跟其他的组件进行交互。例如。很典型的音乐播放器的例子，用户离开播放页面或者进入其他程序，音乐都会继续播放，这里就需要用到Service服务。播放器的Activity会使用Context.startService()来启动一个Service,让音乐一直在后台播放，直到这个Service运行结束。而且，还可以用Context.bindService()方法绑定一个Service,利用提供的借口与之通信。像音乐播放器，可以设置音乐暂停，播放等操作功能。

（3）BroadcastReceiver广播接收机

Broadcast是指广播信息，Android系统本身有很多的广播信息要发送给用户，像电量不足、系统更新等，其他程序也会有很多广播信息，像文件下载、系统通知等。Broadcast Recevier是广播接收机，它和Service一样没有用户界面，主要负责监听和接收广播信息，然后根据程序自身逻辑，调用其他组件处理广播消息。可以启动一个Activity来和用户进行复杂的交互，也可以调用NotificationManager(通知管理器)来通知用户，NotificationManager通知用户的形式很多样化，可以在通知栏显示信息，也可以是响铃、震动、闪LED灯方式。

（4）Content Provider相当于一个可共享的数据仓库接口。应用程序可以将他们的数据存储在文件中、SQLite数据库中，或其他任何有效的方式。一个ContentProvider类实现了一组很标准的方法，能够让其他的应用程序保存或读取该ContentProvider中的各种数据。

Android四大组件之间的关系如图2-3所示：

图2-3 Android各组件关系图



上面介绍了四大组件，Android使用Intent类来完成各个组件之间的调用，实现了调用者与被调用者之间的解耦，相当于一个媒介作用。Intent类描述了应用程序一次操作的动作、动作所涉及的数据等，Android通过Intent的描述就能快速找到相应的组件，完成组件的调用。

## 2.2 JSON数据格式

JSON(JavaScript Object Notation)是一种非常轻量级的文本数据交换语言，以文本为基础，很适合阅读。虽然说JSON是Javascript的一个子集，但这并不代表它不能用在别的语言中。它是一种标准，只要符合JSON的语法构造规则、任何语言都可以使用和解析JSON这种数据格式。

JSON有两种数据结构：object（对象）和array（数组）。对象就是一个“名称-值”对的集合，一个对象一“{”开始，“}”结束，名称和值之间映射冒号隔开，“名称-值”对之间使用逗号分隔。如图2-4所示：

图2-4 JSON对象结构图



数组是值（value）的有序组合。一个以“[”开始，以“]”结束，值与值之间使用逗号隔开。如图2-5所示：

图2-5 JSON数组结构图



值（value）可以是字符串（string）、数值（number）、true、false、null、对象（object）和数组（array）。并且这些结构可以嵌套使用。

JSON和XML一样都是数据交换格式，但是XML是一个完整的标记语言、JSON不是，这使得XML在程序判读方面要花费很多的时间。相比较的话，可以发现它们的异同点如下：

* 它们都有很好的可读性和丰富的解析手段。
* 在扩张性方面、XML由于本身的结构，有很好的扩展性，JSON扩张性不如XML。
* JSON编码难度和解码难度要比XML容易很多。
* JSON相对于XML而言，数据体积更小。
* JSON对数据的描述比XML要差。
* JSON的速度比XML快很多。

不同的场合适合用不同的数据格式，XML在数据存储、扩展和高级检索方面有自己的优势，但JSON由于比XML体积小很多、能够支持快速解析、很适合应用在互联网数据传输上。

Android系统提供了JSON.org包来对JSON数据进行操作，这个包主要包括以下几个类：JSONObject类是构造JSON对象的类，JSONArray类是构造JSON数组的类，根据上面说的JSON语法规则，使用两个类就可以构造JSON对象了。JSONStringer类是使用另一种方式构造一个JSON对象，JSONTokener类是对JSON数据进行解析。JSONException类封装了一些常见的异常处理。在Android系统开发中，开发者使用这些类就可以对JSON进行基本的操作了。

## 2.3 SERVLET编程

由于动态网站良好的用户交互体验，较高的安全性，轻松的后期维护，更高的效率等优点使得动态网站快速发展，其中基于Java高级编程语言的servlet编程技术得到广泛的应用。

### 2.3.1 SERVLET是什么？

Servlet是使用Java Servlet 应用程序设计接口（API）及相关类和方法的 Java 程序。除了 Java Servlet API，Servlet 还可以使用用以扩展和添加到 API 的 Java 类软件包。Servlet 在启用 Java 的 Web 服务器上或应用服务器上运行并扩展了该服务器的能力。Java servlet对于Web服务器就好象Java applet对于Web浏览器。Servlet装入Web服务器并在Web服务器内执行，而applet装入Web浏览器并在Web浏览器内执行。Java Servlet API 定义了一个servlet 和Java使能的服务器之间的一个标准接口，这使得Servlets具有跨服务器平台的特性。

Servlet 通过创建一个框架来扩展服务器的能力，以提供在 Web 上进行请求和响应服务。当客户机发送请求至服务器时，服务器可以将请求信息发送给 Servlet，并让 Servlet 建立起服务器返回给客户机的响应。 当启动 Web 服务器或客户机第一次请求服务时，可以自动装入 Servlet。装入后， Servlet 继续运行直到其它客户机发出请求。Servlet 的功能涉及范围很广。例如，Servlet 可完成如下功能：

1. 创建并返回一个包含基于客户请求性质的动态内容的完整的 HTML页面。
2. 创建可嵌入到现有 HTML 页面中的一部分 HTML 页面（HTML 片段）。
3. 与其它服务器资源（包括数据库和基于 Java 的应用程序）进行通信。
4. 用多个客户机处理连接，接收多个客户机的输入，并将结果广播到多个客户机上。例如，Servlet 可以是多参与者的游戏服务器。
5. 当允许在单连接方式下传送数据的情况下，在浏览器上打开服务器至applet的新连接，并将该连接保持在打开状态。当允许客户机和服务器简单、高效地执行会话的情况下，applet也可以启动客户浏览器和服务器之间的连接。可以通过定制协议或标准（如 IIOP）进行通信。
6. 对特殊的处理采用 MIME 类型过滤数据，例如图像转换和服务器端包括（SSI）。
7. 将定制的处理提供给所有服务器的标准例行程序。例如，Servlet 可以修改如何认证用户。

### 2.3.2 SERVLET的生命周期

Servlet 的生命周期始于将它装入 Web 服务器的内存时，并在终止或重新装入 Servlet 时结束。

1. 初始化

在下列时刻装入 Servlet：

* 如果已配置自动装入选项，则在启动服务器时自动装入。
* 在服务器启动后，客户机首次向 Servlet 发出请求时。
* 重新装入 Servlet 时装入 Servlet 后，服务器创建一个 Servlet 实例并且调用 Servlet 的 init() 方法。在初始化阶段，Servlet 初始化参数被传递给 Servlet 配置对象。

1. 请求处理

对于到达服务器的客户机请求，服务器创建特定于请求的一个“请求”对象和一个“响应”对象。服务器调用 Servlet 的 service() 方法，该方法用于传递“请求”和“响应”对象。service() 方法从“请求”对象获得请求信息、处理该请求并用“响应”对象的方法以将响应传回客户机。service() 方法可以调用其它方法来处理请求，例如 doGet()、doPost() 或其它的方法。

1. 终止

当服务器不再需要 Servlet, 或重新装入 Servlet 的新实例时，服务器会调用 Servlet 的 destroy() 方法。

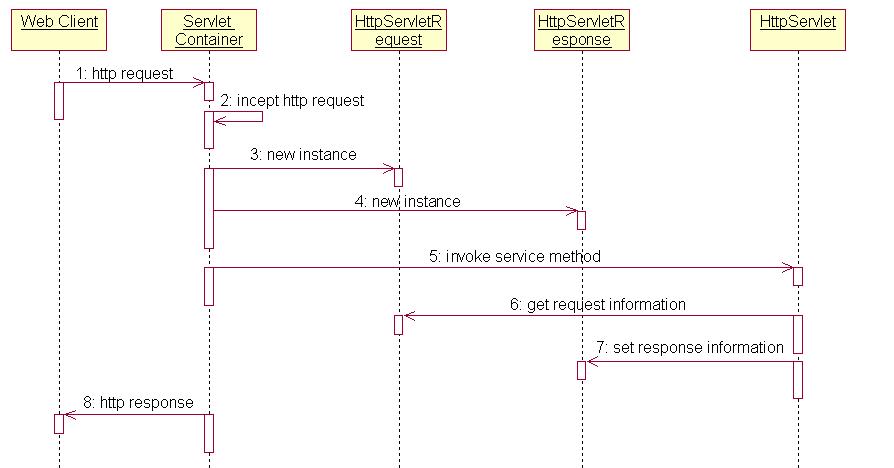
### 2.3.2 SERVLET工作原理

1、首先简单解释一下Servlet接收和响应客户请求的过程，首先客户发送一个请求，Servlet是调用service()方法对请求进行响应的，通过源代码可见，service()方法中对请求的方式进行了匹配，选择调用doGet,doPost等这些方法，然后再进入对应的方法中调用逻辑层的方法，实现对客户的响应。在Servlet接口和GenericServlet中是没有doGet（）、doPost（）等等这些方法的，HttpServlet中定义了这些方法，但是都是返回error信息，所以，我们每次定义一个Servlet的时候，都必须实现doGet或doPost等这些方法。

2、每一个自定义的Servlet都必须实现Servlet的接口，Servlet接口中定义了五个方法，其中比较重要的三个方法涉及到Servlet的生命周期，分别是上文提到的init(),service(),destroy()方法。GenericServlet是一个通用的，不特定于任何协议的Servlet,它实现了Servlet接口。而HttpServlet继承于GenericServlet，因此HttpServlet也实现了Servlet接口。所以我们定义Servlet的时候只需要继承HttpServlet即可。

3、Servlet接口和GenericServlet是不特定于任何协议的，而HttpServlet是特定于HTTP协议的类，所以HttpServlet中实现了service()方法，并将请求ServletRequest、ServletResponse 强转为HttpRequest 和 HttpResponse。

如图2-6：

图2-6 Servlet工作原理

### 2.3.3 JAVA SERVLET API

Java Servlet 开发工具（JSDK）提供了多个软件包，在编写 Servlet 时需要用到这些软件包。其中包括两个用于所有 Servlet 的基本软件包：

javax.servlet 和 javax.servlet.http。可从sun公司的Web站点下载 Java Servlet 开发工具。 下面主要介绍javax.servlet.http提供的HTTP Servlet应用编程接口。

HTTP Servlet 使用一个 HTML 表格来发送和接收数据。要创建一个 HTTP Servlet，请扩展 HttpServlet 类， 该类是用专门的方法来处理 HTML 表格的 GenericServlet 的一个子类。 HTML 表单是由 <FORM> 和 </FORM> 标记定义的。表单中典型地包含输入字段（如文本输入字段、复选框、单选按钮和选择列表）和用于提交数据的按钮。当提交信息时，它们还指定服务器应执行哪一个Servlet（或其它的程序）。 HttpServlet 类包含 init()、destroy()、service() 等方法。其中 init() 和 destroy() 方法是继承的。

1. init() 方法

在 Servlet 的生命期中，仅执行一次 init() 方法。它是在服务器装入 Servlet 时执行的。可以配置服务器，以在启动服务器或客户机首次访问

Servlet 时装入 Servlet。 无论有多少客户机访问 Servlet，都不会重复执行 init() 。

缺省的 init() 方法通常是符合要求的，但也可以用定制 init() 方法来覆盖它，典型的是管理服务器端资源。 例如，可能编写一个定制 init() 来只用于一次装入 GIF 图像，改进 Servlet 返回 GIF 图像和含有多个客户机请求的性能。另一个示例是初始化数据库连接。缺省的 init() 方法设置了 Servlet 的初始化参数，并用它的 ServletConfig 对象参数来启动配置， 因此所有覆盖 init() 方法的 Servlet 应调用 super.init() 以确保仍然执行这些任务。在调用 service() 方法之前，应确保已完成了 init() 方法。

1. service() 方法

service() 方法是 Servlet 的核心。每当一个客户请求一个HttpServlet 对象，该对象的service() 方法就要被调用，而且传递给这个方法一个“请求”（ServletRequest）对象和一个“响应”（ServletResponse）对象作为参数。在 HttpServlet 中已存在 service() 方法。缺省的服务功能是调用与 HTTP 请求的方法相应的 do 功能。例如， 如果 HTTP 请求方法为 GET，则缺省情况下就调用 doGet() 。Servlet 应该为 Servlet 支持的 HTTP 方法覆盖 do 功能。因为 HttpServlet.service() 方法会检查请求方法是否调用了适当的处理方法，不必要覆盖 service() 方法。只需覆盖相应的 do 方法就可以了。

* 当一个客户通过HTML表单发出一个HTTP POST请求时，doPost（）方法被调用。与POST请求相关的参数作为一个单独的HTTP请求从浏览器发送到服务器。当需要修改服务器端的数据时，应该使用doPost()方法。
* 当一个客户通过HTML 表单发出一个HTTP GET请求或直接请求一个URL时，doGet()方法被调用。与GET请求相关的参数添加到URL的后面，并与这个请求一起发送。当不会修改服务器端的数据时，应该使用doGet()方法。

Servlet的响应可以是下列几种类型：

* 一个输出流，浏览器根据它的内容类型（如text/HTML）进行解释。
* 一个HTTP错误响应, 重定向到另一个URL、servlet、JSP。

1. destroy() 方法

* destroy() 方法仅执行一次，即在服务器停止且卸装Servlet 时执行该方法。典型的，将 Servlet 作为服务器进程的一部分来关闭。缺省的 destroy() 方法通常是符合要求的，但也可以覆盖它，典型的是管理服务器端资源。例如，如果 Servlet 在运行时会累计统计数据，则可以编写一个 destroy() 方法，该方法用于在未装入 Servlet 时将统计数字保存在文件中。另一个示例是关闭数据库连接。
* 当服务器卸装 Servlet 时，将在所有 service() 方法调用完成后，或在指定的时间间隔过后调用 destroy() 方法。一个Servlet 在运行service() 方法时可能会产生其它的线程，因此请确认在调用 destroy() 方法时，这些线程已终止或完成。

1. GetServletConfig（）方法

GetServletConfig（）方法返回一个 ServletConfig 对象，该对象用来返回初始化参数和 ServletContext。ServletContext 接口提供有关servlet 的环境信息。

1. GetServletInfo（）方法

GetServletInfo（）方法是一个可选的方法，它提供有关servlet 的信息，如作者、版本、版权。

当服务器调用sevlet 的Service（）、doGet（）和doPost（）这三个方法时，均需要 “请求”和“响应”对象作为参数。“请求”对象提供有关请求的信息，而“响应”对象提供了一个将响应信息返回给浏览器的一个通信途径。javax.servlet 软件包中的相关类为ServletResponse和ServletRequest，而javax.servlet.http 软件包中的相关类HttpServletRequest和HttpServletResponse。Servlet 通过这些对象与服务器通信并最终与客户机通信。Servlet 能通过调用“请求”对象的方法获知客户机环境，服务器环境的信息和所有由客户机提供的信息。Servlet 可以调用“响应”对象的方法发送响应，该响应是准备发回客户机的。

## 2.4 JDBC技术

JDBC是一种可用于执行SQL语句的JavaAPI（ApplicationProgrammingInterface，应用程序设计接口）。它由一些Java语言写的类、界面组成。JDBC给数据库应用开发人员、数据库前台工具开发人员提供了一种标准的应用程序设计接口，使开发人员可以用纯Java语言编写完整的数据库应用程序。

通过使用JDBC，开发人员可以很方便地将SQL语句传送给几乎任何一种数据库。也就是说，开发人员可以不必写一个程序访问Sybase，写另一个程序访问Oracle，再写一个程序访问Microsoft的SQLServer。用JDBC写的程序能够自动地将SQL语句传送给相应的数据库管理系统（DBMS）。不但如此，使用Java编写的应用程序可以在任何支持Java的平台上运行，不必在不同的平台上编写不同的应用。Java和JDBC的结合可以让开发人员在开发数据库应用时真正实现“WriteOnce，RunEverywhere！”

Java具有健壮、安全、易用等特性，而且支持自动网上下载，本质上是一种很好的数据库应用的编程语言。它所需要的是Java应用如何同各种各样的数据库连接，JDBC正是实现这种连接的关键。

JDBC扩展了Java的能力，如使用Java和JDBCAPI就可以公布一个Web页，页中带有能访问远端数据库的Ap?plet。或者企业可以通过JDBC让全部的职工（他们可以使用不同的操作系统，如Windwos，Machintosh和UNIX）在In?tranet上连接到几个全球数据库上，而这几个全球数据库可以是不相同的。随着越来越多的程序开发人员使用Java语言，对Java访问数据库易操作性的需求越来越强烈。

MIS管理人员喜欢Java和JDBC，因为这样可以更容易经济地公布信息。各种已经安装在数据库中的事务处理都将继续正常运行，甚至这些事务处理是存储在不同的数据库管理系统中；而对新的数据库应用来说，开发时间将缩短，安装和版本升级将大大简化。程序员可以编写或改写一个程序，然后将它放在服务器上，而每个用户都可以访问服务器得到最新的版本。对于信息服务行业，Java和JDBC提供了一种很好的向外界用户更新信息的方法。

简单地说，JDBC能完成下列三件事：

* 同一个数据库建立连接。
* 向数据库发送SQL语句。
* 处理数据库返回的结果。

JDBC架构:

JDBC API支持两层和三层的处理模式对数据库的访问，但一般JDBC体系结构由两层组成：

* JDBC API: 这提供了应用程序到JDBC管理器连接。
* JDBC Driver API: 这支持JDBC管理器 - 驱动器连接。

JDBC API使用一个驱动程序管理器和数据库特定的驱动程序提供透明的异构数据库的连接。确保使用正确的驱动程序来访问每个数据源的JDBC驱动程序管理器。驱动程序管理器能够支持多个并发连接到多个异构数据库的驱动程序。

JDBC架构图如图2-7，它显示的JDBC驱动程序和Java应用程序与驱动程序管理器的位置：

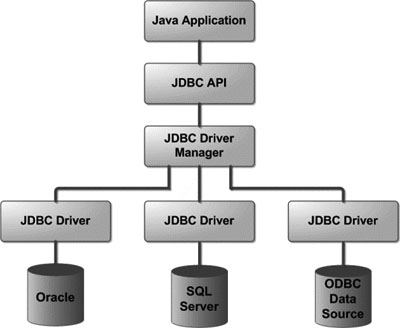


图2-7 JDBC架构图

JDBC API提供了以下接口和类：

* DriverManager: 这个类管理数据库驱动程序的列表。从Java应用程序的连接请求匹配的合适的数据库驱动程序，使用通讯子协议。第一个JDBC驱动程序识别某个子协议将被用来建立一个数据库连接。
* Driver: 此接口处理与数据库服务器的通信。将直接与驱动程序对象很少。相反，您可以使用DriverManager隔离对象，这种类型的管理对象。它也抽象与驱动程序对象与工作相关的细节。
* Connection : 此接口与用于接触一个数据库的所有方法。连接对象通信的情况下，即，所有的通信是只通过与数据库连接对象。
* Statement : 使用接口提交到数据库的SQL语句创建的对象。一些派生的接口接受，除了执行存储过程的参数。
* ResultSet: 这些对象保存后，使用Statement对象执行SQL查询从数据库中检索数据。它作为一个迭代器，让您可以通过它的数据移动。
* SQLException: 这个类处理的数据库应用程序中发生的任何错误。

# 移动电子简易交易系统需求分析与设计

## 3.1 系统需求分析

需求分析是做软件开发中不可或缺的部分，主要分析解决产品是“做什么”的问题。通过对用户的理解，对行业的理解以及各种调研等方法来明确产品的目标用户，来确定产品“满足了用户哪方面的需求”。将得到的需求分析细化，可以很好的为后面的系统设计与开发提供相应的指导。

### 3.1.1 用户定位

本移动电子简易交易系统主要将目标用户定位在大学生校园用户，在校大学生之间基本上有很多相同的兴趣爱好和消费趋向，它们交易流通的物品主要集中在书籍、日常用品、自行车、电子产品等等上面。而且在校大学生一般都有智能手机，也很热衷于用网上交易，他们比起社会上的人来说，比较单纯，性用度良好。并且一个校园各个区域在地理位置上很近，更方便用户见面交易取货。这些方面都为建立移动端的交易市场提供了良好的条件。

### 3.1.2 系统需求详细说明

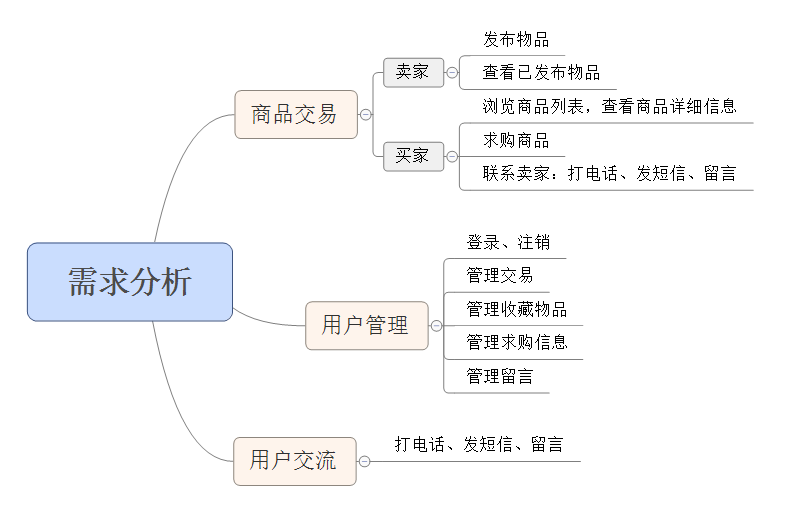
我们详细分析了用户在交易过程中必不可少的部分：商品交易、用户管理、用户之间的交流。本系统的详细需求分析如图3-1所示：

图3-1 详细需求分析图

商品交易是系统的核心需求，本系统的每一个用户既可以是卖家，又可以是买家，两种角色集于一身。但是从需求的角度来说，对卖家和买家又进行了细分。对于卖家来说，他们可以发布自己需要出售的商品，包括全新的或者自己闲置的物品。对于买家来说，有目标的用户可以在商品列表里面寻找，如果没有找到可以发布求购商品信息；没有目标的用户可以查看其他用户发布的商品信息，并且收藏喜欢商品；如果有购买意向可以联系卖家，通过打电话，发短信，或者留言的方式。然后由线上引导到线下，然后双方当面交易。

系统围绕着个人对个人的交易来进行构造，是一个典型的C2C电子商务系统。所以用户的管理非常重要，首先用户需要登录和退出系统，需要有个人中心，以便查看和管理自己的交易信息。

## 3.2 系统功能模块设计

通过上一节对用户的需求分析，本节将这些需求转化成具体的功能模块设计。由于是移动端的应用，在设计的时候要充分考虑到移动端移动化，碎片化，以及用户使用移动智能手机的场景，力求让系统坐到简洁、清晰、易用。基于以上思路，本系统主要分为商品交易、用户管理、交流互动三大模块，系统的各个功能模块框架如图3-2所示：

图3-2 功能模块图



### 3.2.1 商品交易模块设计

商品交易是本系统最为重要的核心模块，主要为用户创造简洁方便的交易流程和透明标准化的商品交易平台。由于交易范围是在同校，就是离用户很近的距离，所以交易流程与淘宝、亚马逊等传统电商的流程很不一样，在本系统上，用户是在线上发布商品供需信息，在线下当面交易。系统提供发布商品信息、浏览商品、发布求购商品、买家卖家交流等功能，不提供支付和物流等功能。

用户可以是卖家也可以是买家，但队友商品交易而言，这是两个不同的角色，本系统商品交易的流程如图3-3所示：



图3-2 系统商品交易流程图

系统在设计商品交易流程时力求做到简单、直接，不让用户感到难用。

对于卖家来说，要想发布商品，需要先编辑商品的信息，包括商品的图片、描述，价格、联系方式等，这些商品信息都会保存在系统的数据库里，然后就等待有意向的买家来联系。对于买卖双方的联系交流，系统设计了打电话、发短信、留言等多种方式，针对不同的用户场景。

对于买家来说，有目标的用户可以直接浏览商品列表寻找自己需要的商品，没目标的用户可以浏览商品信息列表，如果对商品感兴趣的话，可以查看商品的详细信息，就是卖家提供的商品图片、价格、描述等。有购买意向的话，可以通过系统提供的多种联系方式联系卖家。如果没有找到自己想要的商品，还可以发布求购信息。

随后基于线上的联系。买卖双方可以在线下自己当面交易。系统不负责线下交易，所以在商品的交易环节，主要包括发布商品，商品展示两个部分。

1. 发布商品模块

用户要出售商品，首先就需要发布商品信息。系统提供两个发布入口，分别是商品发布，求购信息发布。

1. 商品展示模块

商品的显示模块分为分类显示和商品详情页。

1. 商品分类显示页是将商品分类为学校最常见的几种如体育用品、生活用品、自行车、电子产品、书籍、办公用品、电脑配件等。用户进入不同的分类中就可以浏览相应的商品信息。
2. 商品详情页：包括更为具体的商品信息，在列表信息的基础上增加多张商品大图，商品详细描述。还有一些很重要的操作功能：联系卖家、留言、收藏等。联系卖家有三种方式：打电话、发短信和留言。

### 3.2.2 用户管理模块设计

如果说商品交易模块是以商品为中心的功能模块，那么用户管理模块就是以用户行为为中心的模块，本模块旨在让用户能方便的使用系统服务、掌握和管理自己的商品交易和物品收藏等信息。

1. 用户注册模块

要使用本系统，必须有本系统的账户。对于登录和未登录的用户，有不同的系统权限。未登录的用户只能浏览商品信息列表，不能发布商品信息，不能留言，不能联系卖家。登录的用户则拥有所有权限。所以在未登录用户想使用登录用户拥有的权限是可以快捷注册。用户的注册信息将提交到MySQL数据库保存下来。

1. 用户登录模块

用户在成功注册后，可以使用账号和密码登录本系统，登录后可以则拥有发布商品，收藏商品，联系卖家等高级功能。

1. 发布商品管理

使用本系统的用户想兼职零售一些商品或者转让自己的闲置商品，那么可以在个人中心发布自己想出售的商品。用户在发布商品前必须编辑商品详细信息，包括商品照片、商品标题、商品分类、商品描述、联系方式等。

1. 求购商品管理

一些用户在浏览商品列表的时候没有找到自己想要的商品，那么可以在个人中心的我的求购信息中发布自己所需求的商品，用户发布求购信息需要编辑求购商品的详细信息，包括商品名称、商品类型、商品详细的描述等。

1. 收藏商品管理

有些用户在浏览商品的时候对某种商品感兴趣但是暂时又不想买，那么可以选择将商品收藏起来，等想买的时候可以在个人中心的我的收藏轻松的找到。对于那些已经过期了的收藏可以快捷的删除。

1. 留言管理

用户在感兴趣的商品留言后再个人中心的我的留言中都可以看到，当然也可以看卖家的回复。如果这些留言都已经过期了可以很方便的一键删除，管理非常的方便。

### 3.2.2 买家卖家交流互动

买家和卖家的交流互动在整个商品交易中是必不可少的一个环节，交流方式的便捷程度间接的决定了交易成功的可能。本系统提供了打电话、发短信、留言这些非常简便的互动方式，让卖家买家随时随地随便交流。

## 3.3 数据库设计

在本系统中，用户使用Android智能手机通过网络的连接，向服务端请求数据处理，数据修改或者数据上传等。服务端访问数据库，将所需要的数据处理后发送回移动端。本系统的服务端需要保存大量的数据，如用户的个人信息、交易的商品信息等，并且还要定时对数据进行复杂的操作。本系统服务端使用MySQL数据库来存储相关数据。这是因为MySQL是跨平台的开源的关系型数据库，它操作简单、可靠性高、稳定性好。

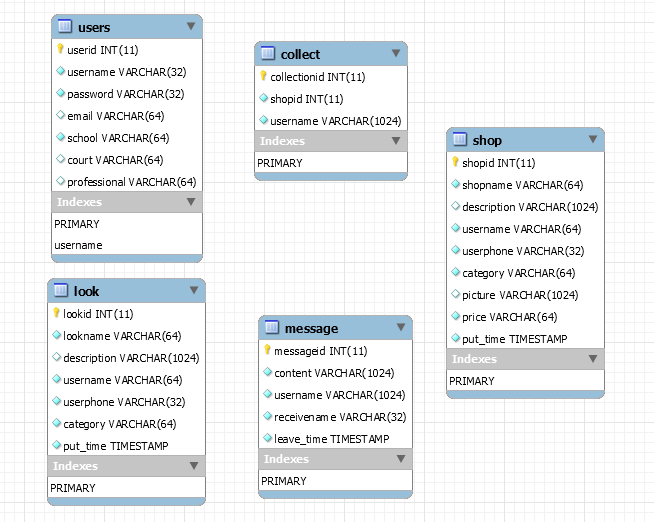
根据本系统的各个功能模块，本系统设计了5张数据库表，本节将简单介绍。图3-3为系统数据库E-R图：

图3-2 系统数据库E-R图

本数据库总共五张表，其中每张表的字段设计及详细功能如下：

1. 用户表（users）

用户表主要用于用户注册相关信息，表示的是每一个用户实体。表的字段主要包括：主键userid（用户id）、username（用户名）、password（用户密码）、email（用户邮箱）、school（用户所在学校）、curt（用户所在学院）、professional（用户所在专业）。

表3-3 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文名称 | 字段类型 | 备注 |
| userid | 用户id | int | 主键，自增 |
| username | 用户名 | varchar(32) | 不为空 |
| password | 用户密码 | varchar(32) | 不为空 |
| email | 用户邮箱 | varchar(64) |  |
| school | 用户所在学校 | varchar(64) | 不为空 |
| court | 用户所在学院 | varchar(64) |  |
| professional | 用户所在专业 | varchar(64) |  |

1. 商品信息表（shop）

商品信息表主要用于记录发布的商品信息。表的字段主要包括：shopid(商品id)、shopname（商品名）、description（商品描述）、username（发布者姓名）、userphone（发布者电话）、category（商品类别）、picture（商品图片）、price（商品价格）、put\_time（发布时间）。

表3-4 商品信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文名称 | 字段类型 | 备注 |
| shopid | 商品id | int | 主键，自增 |
| shopname | 商品名 | varchar(64) | 不为空 |
| description | 商品描述 | varchar(1024) |  |
| username | 发布者姓名 | varchar(64) | 不为空 |
| userphone | 发布者电话 | varchar(32) | 不为空 |
| category | 商品类别 | varchar(64) | 不为空 |
| picture | 商品图片 | varchar(1024) |  |
| price | 商品价格 | varchar（64） | 不为空 |
| put\_time | 发布时间 | timestamp | 不为空 |

1. 留言信息表（message）

留言信息表主要是用于记录用户所有的留言信息。表的字段主要包括：messageid（信息id）、content（留言内容）、username（留言者姓名）、receivename（接收留言者姓名）、leave\_time（留言时间）。

表3-5 留言信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文名称 | 字段类型 | 备注 |
| messageid | 信息id | int | 主键，自增 |
| content | 留言内容 | varchar(1024) | 不为空 |
| username | 留言者姓名 | varchar(32) | 不为空 |
| receivename | 接收留言者姓名 | varchar(32) | 不为空 |
| leave\_time | 留言时间 | timestamp | 不为空 |

1. 收藏信息表（collect）

收藏信息表用于记录商品收藏者的信息。表的字段主要包括：collectionid（收藏信息id）、shopid（商品id）、username（收藏者姓名）。

表3-6 收藏信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文名称 | 字段类型 | 备注 |
| collectionid | 收藏信息id | int | 主键，自增 |
| shopid | 商品id | int | 不为空 |
| username | 收藏者姓名 | varchar(32) | 不为空 |

1. 求购信息表（look）

求购信息表用于记录求购信息。表的字段主要包括：lookid（求购信息id）、lookname（求购商品名）、description（求购商品描述）、username（求购者姓名）、userphone（求购者电话）、category（求购类别）、put\_time（发布时间）。

表3-7 求购信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文名称 | 字段类型 | 备注 |
| lookid | 求购信息id | int | 主键，自增 |
| lookname | 求购商品名 | varchar(64) | 不为空 |
| description | 求购商品描述 | varchar(1024) |  |
| username | 求购者姓名 | varchar(64) | 不为空 |
| userphone | 求购者电话 | varchar(32) | 不为空 |
| category | 求购类别 | varchar(64) | 不为空 |
| put\_time | 发布时间 | timestamp | 不为空 |

## 3.4 系统总体部署

本系统采用主流的移动互联网相关技术，设计了一套相对完善的且符合自身需求的系统解决方案。系统部署方案如图3-8所示：

图3-8 系统部署图



用户的手机客户端通过移动网络访问本系统的服务器，服务器端负责响应请求，并且对数据库进行操作。

# 4. 移动电子简易交易系统的实现与测试

## 4.1 开发环境与工具

本系统是基于Android平台来进行开发的，包括Android客户端和服务端，主要用Java语言，具体的开发环境和工具如表4-1所示：

表4-1 系统开发环境和工具

|  |  |
| --- | --- |
| 操作系统 | Windows10 |
| 开发工具 | Eclipse+ADT+AndroidSDK |
| 运行环境 | JDK1.8+Android 4.2 |
| Web服务器 | Tomcat7 |
| 模拟器 | Genymotion |
| 数据库 | MySQL 5.7.9 |

Eclipse是开源的基于Java的可扩展的开发平台，可以用于通过插件来在组建开发环境。本系统使用Eclipse+ADT插件搭建Android开发环境。使用myeclipse开发服务端，此处仅叙述客户端的开发过程。

在Eclipse上可以很快的搭建好Android的开发环境，包括Android SDK（Software Development Kit）和ADT(Android Develop Tools)插件。Android SDK是Android的软件开发包，提供一些库文件和开发需要的工具。ADT是集成于eclipse上的插件，用于快速开发。本系统没有使用Android SDK自带的AVD（Android Virtual Device）测试，而是使用Genymotion，后者更加高效更加贴近实际智能设备。

Tomcat是目前非常流行的Java Web服务器，与Android一本同源，这样使得客户端和服务端的连接更加高效可靠。本系统的Android端需要通过Web服务端来和数据库进行交互。如前面所说，本系统采用的是MySQL数据库，使用JDBC技术对数据库进行操作。

## 4.2 商品交易模块的实现

商品交易模块是整个系统的核心，这部分功能的实现非常重要。后面将详细描述商品展示模块和商品发布模块的实现。

### 4.2.1 商品展示模块

商品的展示主要分为列表展示和详细信息展示。由客户端向服务端发送数据请求，然后由客户端将服务端返回的数据解析并且有序的展示出来。这里实现此功能主要的类有7个；各个类之间的关系如图4-2所示：

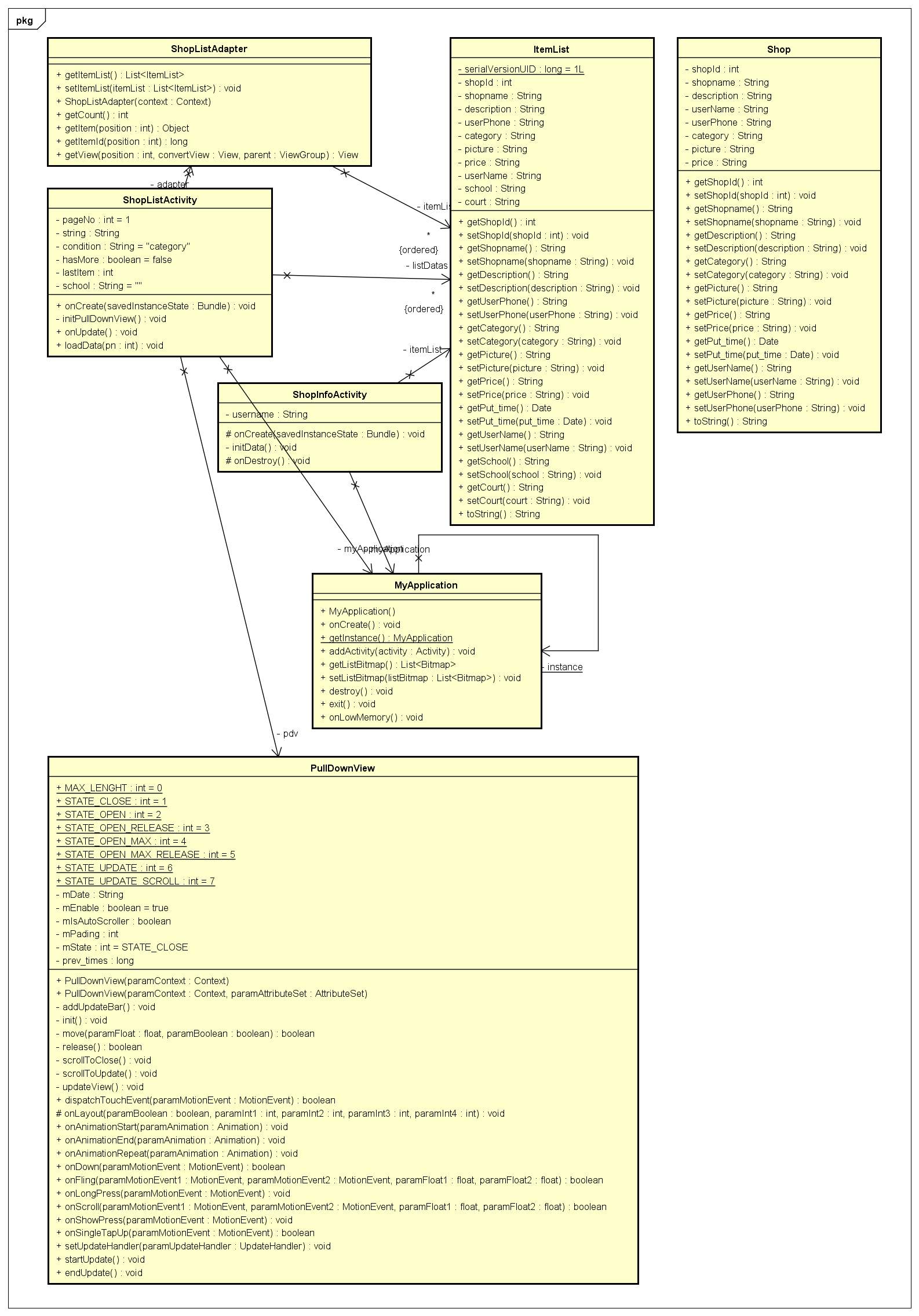


图4-1 商品展示模块类图

其中ShopListActivity类为商品列表展示主Activity。通过loadData()方法向服务端请求数据，然后由ShopListAdapter类对数据进行适配，并且通过MyApplication类创建的service动态加载在ItemList对象上，ItemList实现了Serializable接口，方便将数据有序化的保存。然后将适配器适配在页面上的listview中显示出来。并且实现实时更新。ShopInfoActivity类是商品详细信息展示Activity。在这个Activity中我们创建了一个GuidePageAdapter内部类和一个MyTask内部类前者功能是页面数据适配器，后者是为了异步加载图片，这样做的好处就是，当网络状况不是那么满意的时候大图片的加载也不会使Activity卡住。Shop类为商品类，用于保存每件商品的具体信息。这里我们重写了toString()方法，在调用这个方法的时候会将Shop数据构造成字符串，方便信息在网络上传递。下面我将贴出部分代码：

loadData()方法：

|  |
| --- |
| **public** **void** loadData(**int** pn) {  HashMap<String, Object> params = **new** HashMap<String, Object>();  params.put("pageNo", pn);  params.put("value", string);  params.put("condition", condition);  params.put("school", school);  **if** (pageNo == 1) {  listDatas.clear();  footer.setVisibility(View.*GONE*);  lv.setFooterDividersEnabled(**false**);  } **else** {  footer.setVisibility(View.*VISIBLE*);  lv.setFooterDividersEnabled(**true**);  }  HttpHelper.*asyncPost*(Constants.*URL* + "/second-hand/shop\_info.do",  params, **new** Callback() {  @Override  **public** **void** dataLoaded(Message msg) {  footer.setVisibility(View.*GONE*);  **if** (HttpStatus.*SC\_OK* != msg.what) {  AppException.*http*(msg.what).makeToast(  ShopListActivity.**this**);  pdv.endUpdate();  **return**;  }  String json = (String) msg.obj;  Log.*v*("dddddddddddddddddddddd", json);  PageModel<ItemList> pm = PageModel.*jsonConvert*(json);  listDatas.addAll(pm.getData());  adapter.setItemList(listDatas);  adapter.notifyDataSetChanged();  pdv.endUpdate();  **if** (pageNo < pm.getPageCount()) {  hasMore = **true**;  } **else** {  hasMore = **false**;  footer.setVisibility(View.*VISIBLE*);  listview\_foot\_progress  .setVisibility(View.*INVISIBLE*);  listview\_foot\_more.setText("已加载全部");  }  }  });  } |

GuidePageAdapter内部类：

|  |
| --- |
| **public** **class** GuidePageAdapter **extends** PagerAdapter {  @Override  **public** **int** getCount() {  **return** pageViews == **null** ? 0 : pageViews.size();  }  @Override  **public** **boolean** isViewFromObject(View arg0, Object arg1) {  **return** arg0 == arg1;  }  @Override  **public** **void** destroyItem(View arg0, **int** arg1, Object arg2) {  ((ViewPager) arg0).removeView(pageViews.get(arg1));  }  @Override  **public** Object instantiateItem(View arg0, **int** arg1) {  View view = pageViews.get(arg1);  view.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  **if** (!listBitmap.isEmpty()) {  myApplication.setListBitmap(listBitmap);  Intent intent = **new** Intent(ShopInfoActivity.**this**,  PicturePreview.**class**);  ShopInfoActivity.**this**.startActivity(intent);  } **else** {  Toast.*makeText*(ShopInfoActivity.**this**, "没有图片", 0).show();  }  }  });  ((ViewPager) arg0).addView(view);  **return** pageViews.get(arg1);  }  } |

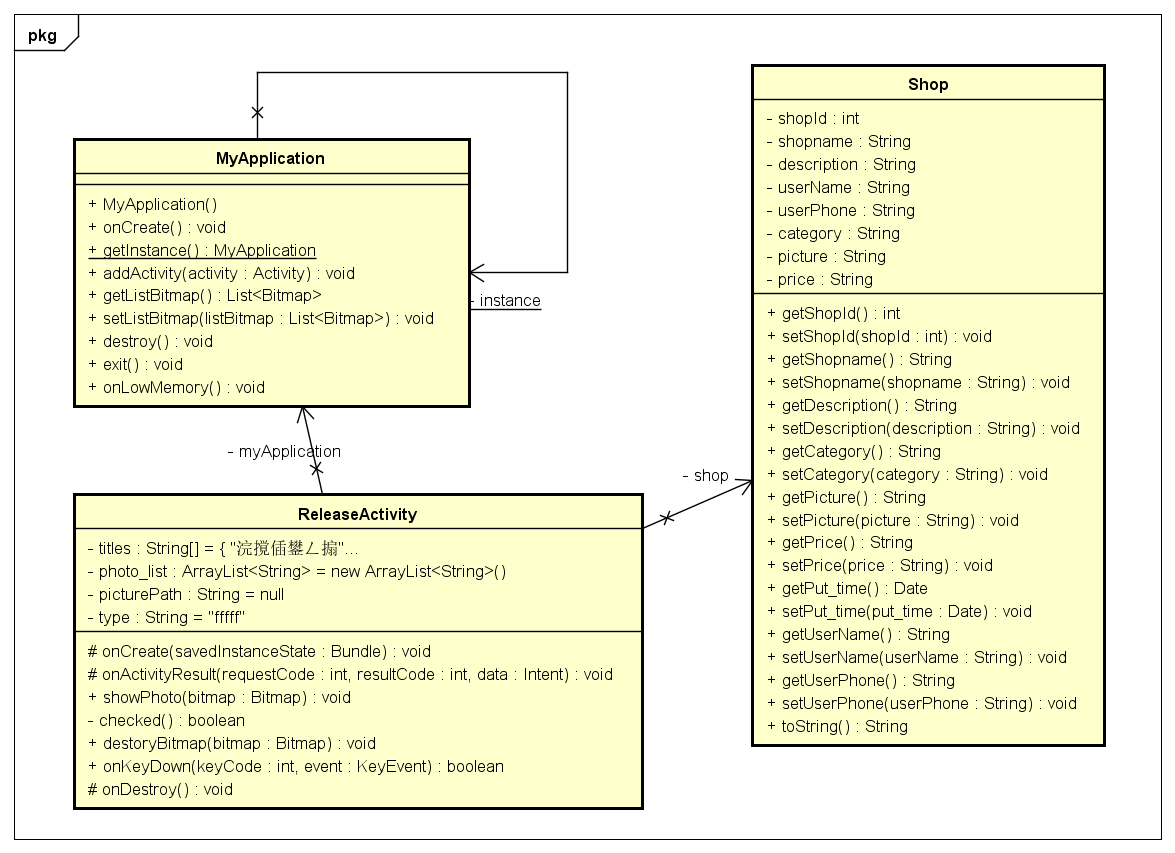
MyTask内部类：

|  |
| --- |
| **public** **class** MyTask **extends**  AsyncTask<String, Void, HashMap<String, Object>> {  @Override  **protected** HashMap<String, Object> doInBackground(String... params) {  **try** {  HttpClient httpClient = HttpHelper.*getHttpClient*();  String uri = params[0];  HttpResponse response = httpClient.execute(**new** HttpGet(uri));  **if** (response.getStatusLine().getStatusCode() == 200) {  InputStream is = (InputStream) response.getEntity()  .getContent();  BitmapFactory.Options opts = **new** BitmapFactory.Options();  opts.inSampleSize = 2;  Bitmap bitmap = BitmapFactory.*decodeStream*(is, **null**, opts);  is.close();  HashMap<String, Object> hashMap = **new** HashMap<String, Object>();  hashMap.put("bitmap", bitmap);  hashMap.put("int", params[1]);  **return** hashMap;  } **else** {  **return** **null**;  }  } **catch** (Exception e) {  e.printStackTrace();  **return** **null**;  }  }  @Override  **protected** **void** onPostExecute(HashMap<String, Object> result) {  **if** (result == **null**) {  Toast.*makeText*(ShopInfoActivity.**this**, "加载图片失败", 1).show();  **return**;  }  Bitmap bitmap = (Bitmap) result.get("bitmap");  String string = (String) result.get("int");  listBitmap.add(bitmap);  **if** (string.equals("1")) {  image1.setImageBitmap(bitmap);  } **else** **if** (string.equals("2")) {  image2.setImageBitmap(bitmap);  } **else** {  image3.setImageBitmap(bitmap);  }  }  } |

当然了，具体的实现方式并不是这样简单的几个方法，在这里我就不多说了，我只是将其概括到最容易理解的范围。

### 4.2.2 商品发布模块

发布商品是本系统最基本的功能。基本原理是客户端收集并验证发布商品的数据，数据经过处理后由客户端发送给服务端，由服务端写入数据库。实现此功能的类主要有3个，各个类之间的关系如图4-2：

图4-2 商品发布模块类图

ReleaseActivity类为发布功能类，提供了发布商品的基本功能。这个类继承了Activity类，用于添加物品详细信息，如商品名称、描述、图片、价格、联系方式等，并且进行上传操作。为了更加清晰的叙述发布功能的整个流程，我画出了功能实现流程图，如图4-3所示：

图4-3 发布详细流程图



在Activity中通过重写onClickListener接口里的onClick()方法可以实现监听触发事件。

本系统需要添加的商品信息包括两类：图片信息、文字信息（标题、描述、价格等）。接下来会详细介绍照片的添加实现。

添加图片的信息的方法有两种，分别是“拍照”和“从本地图库选择”，在ReleaseActivity中点击添加图片，会出现弹出选择两种方式的Dialog。当用户选择“相机”，就会直接调用系统的camera对象takePicture方法来进行拍照。选择“相册”，将会跳转到来源选择Dialog有图库和文件管理器两个来源，点击相应的图标会启动相应的系统应用。相关的代码如下：

|  |
| --- |
| addImageView = (ImageView) **this**.findViewById(R.id.*addImageView*);  addImageView.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  dialogView = getLayoutInflater().inflate(R.layout.*my\_dialog*,  **null**, **false**);  dialog = **new** Dialog(ReleaseActivity.**this**, R.style.*myDialogTheme*);  dialog.setContentView(dialogView);  dialog.show();  // 从相册选择图片  gallery = (TextView) dialogView.findViewById(R.id.*gallery*);  gallery.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  **if** (photo\_list.size() == 3) {  Toast.*makeText*(ReleaseActivity.**this**,  "对不起，内存限制不能再添加图片了～～", Toast.*LENGTH\_LONG*)  .show();  } **else** {  Intent intent = **new** Intent(  Intent.*ACTION\_GET\_CONTENT*);  intent.setType("image/\*");  intent.putExtra("return-data", **true**);  startActivityForResult(intent, 2);  }  dialog.dismiss();  }  });  // 拍照  camera = (TextView) dialogView.findViewById(R.id.*camera*);  camera.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  **if** (photo\_list.size() == 3) {  Toast.*makeText*(ReleaseActivity.**this**,  "对不起，内存限制只能添加少于等于三张图片", Toast.*LENGTH\_LONG*)  .show();  } **else** {  Intent intent = **new** Intent(  MediaStore.*ACTION\_IMAGE\_CAPTURE*);    startActivityForResult(intent, 1);  }  dialog.dismiss();  }  });  }  }); |

当然图片获取成功后需要处理成统一的规格大小，在ReleaseActivity中通过重写onActivityResult（）方法处理返回的图片，代码如下：

|  |
| --- |
| **protected** **void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  **super**.onActivityResult(requestCode, resultCode, data);  bitmap = **null**;  **if** (requestCode == 1 && resultCode == *RESULT\_OK*) {  String sdStatus = Environment.*getExternalStorageState*();  **if** (!sdStatus.equals(Environment.*MEDIA\_MOUNTED*)) { // 检测sd是否可用  Log.*v*("TestFile",  "SD card is not avaiable/writeable right now.");  **return**;  }  Bundle bundle = data.getExtras();  bitmap = (Bitmap) bundle.get("data");// 获取相机返回的数据，并转换为Bitmap图片格式  FileOutputStream b = **null**;  File file = **new** File("/sdcard/campus\_life/myImage/");  file.mkdirs();// 创建文件夹  picturePath = "/sdcard/campus\_life/myImage/"  + System.*currentTimeMillis*() + ".jpg";  **try** {  b = **new** FileOutputStream(picturePath);  bitmap.compress(Bitmap.CompressFormat.*JPEG*, 100, b);// 把数据写入文件  } **catch** (FileNotFoundException e) {  e.printStackTrace();  } **finally** {  **try** {  b.flush();  b.close();  } **catch** (IOException e) {  e.printStackTrace();  }  }  } **else** **if** (requestCode == 2 && resultCode == *RESULT\_OK* && **null** != data) {  Uri selectedImage = data.getData();  **try** {  String[] filePathColumn = { MediaStore.Images.Media.*DATA* };  Cursor cursor = getContentResolver().query(selectedImage,  filePathColumn, **null**, **null**, **null**);  cursor.moveToFirst();  **int** columnIndex = cursor.getColumnIndex(filePathColumn[0]);  picturePath = cursor.getString(columnIndex);  cursor.close();  BitmapFactory.Options options = **new** BitmapFactory.Options();  options.inSampleSize = 4;  bitmap = BitmapFactory.*decodeFile*(picturePath,options);  } **catch** (Exception e) {  e.printStackTrace();  }  }  **if** (bitmap != **null** && picturePath != **null**) {  Log.*v*("dddddddddddddddddddddd", picturePath);  photo\_list.add(picturePath);  bitmap\_list.add(bitmap);  showPhoto(bitmap);  }  } |

在商品信息提交到服务端之前，必须对用户输入的商品信息进行验证，在这里checked（）方法实现了这个功能，相关代码如下：

|  |
| --- |
| **private** **boolean** checked() {  **if** (shopname.getText().toString() == **null**  || shopname.getText().toString().equals("")) {  Toast.*makeText*(**this**, "标题不能为空", Toast.*LENGTH\_SHORT*).show();  **return** **false**;  } **else** **if** (price.getText().toString() == **null**  || price.getText().toString().equals("")) {  Toast.*makeText*(**this**, "价格不能为空", Toast.*LENGTH\_SHORT*).show();  **return** **false**;  } **else** **if** (userPhone.getText().toString() == **null**  || userPhone.getText().toString().equals("")  || userPhone.length() != 11) {  Toast.*makeText*(**this**, "电话是十一位", Toast.*LENGTH\_SHORT*).show();  **return** **false**;  } **else** {  **return** **true**;  }  } |

## 4.3 用户社交模块的实现

在整个交易的过程中，卖家与买家之间的交流必不可少，本系统提供了三种交流方式，分别是打电话、发短信、留言。下面将详细介绍留言功能的实现过程。

### 4.3.1 留言板模块

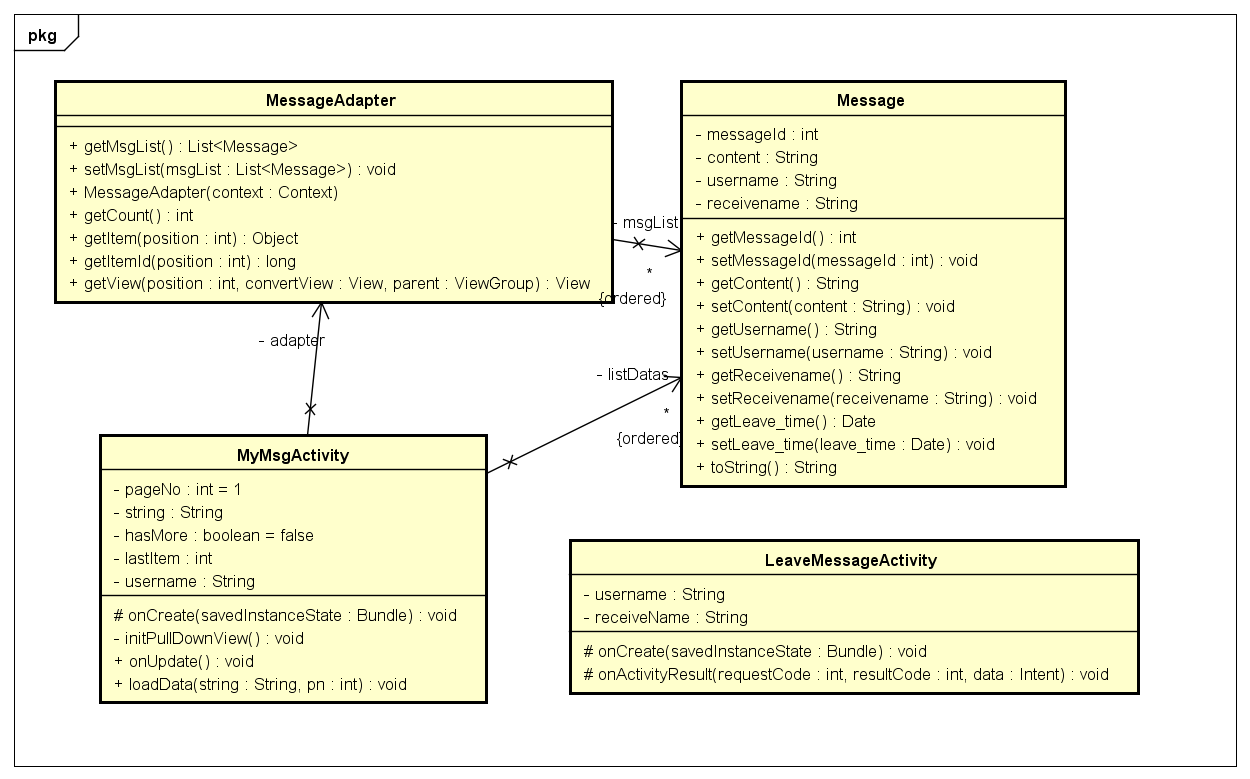
留言板是本系统用户之间交流的功能之一。用户可以直接给在商品详情页给用户留言，然后在用户的个人中心可以查看自己的留言实现这些功能主要的类有4个，图4-4为留言板模块类图：

图4-4 留言板模块类图

其中LeaveMessageActivity类实现留言功能。LeaveMessageActivity继承了activity类，在用户点击留言按钮后会跳转到此活动，在活动页面用户编辑好留言信息后点击发布按钮即可将留言内容提交到服务端然后由服务端写入数据库当中。留言按钮功能实现源码如下所示：

|  |
| --- |
| leaveMssage.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  **if** (!et\_message.getText().toString().equals("") && et\_message.getText().toString() != **null**) {  HashMap<String, Object> params = **new** HashMap<String, Object>();  params.put("content", et\_message.getText().toString());  params.put("username", username);  params.put("receivename", receiveName);  HttpHelper.*asyncPost*(Constants.*URL*  + "/second-hand/msg\_add.do", params,  **new** Callback() {  @Override  **public** **void** dataLoaded(Message msg) {  **if** (HttpStatus.*SC\_OK* != msg.what) {  AppException.*http*(msg.what).makeToast(  LeaveMessageActivity.**this**);  **return**;  }  Toast.*makeText*(LeaveMessageActivity.**this**,  msg.obj.toString(),  Toast.*LENGTH\_SHORT*).show();  LeaveMessageActivity.**this**.finish();  }  });  }**else** {  Toast.*makeText*(LeaveMessageActivity.**this**, "请填写内容", 0).show();  }  }  }); |

当用点击留言是，首先会验证留言内容，如果没有输入内容则会提示用户输入留言内容。

当用户留言成功后，可以在用户的个人中心查看自己的留言，这里会将用户所有的留言展示出来，这个功能由MyMsgActivity类实现。原理和过程跟商品的展示差不多，先由客户端向服务端发送请求，服务端返回数据，然后客户端处理数据这里由MessageAdapter对数据进行处理，然后适配到用于显示的listview。留言板既然能留言那么也得有删除留言的功能。MyMsgActivity类实现了这个功能，当用户点击删除功能按钮后会弹出一个用于确认操作的Dialog，当用户点击确定后，会启动服务器连接线程将删除留言的指令发送至服务端，然后服务端删除数据库中的数据。相关代码如下：

|  |
| --- |
| delete.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  View view = getLayoutInflater().inflate(R.layout.*my\_dialog*,  **null**, **false**);  TextView title = (TextView) view.findViewById(R.id.*title*);  title.setText("温馨提示");  TextView message = (TextView) view  .findViewById(R.id.*message*);  message.setVisibility(View.*VISIBLE*);  message.setText("确定：删除全部选项");  TextView gallery = (TextView) view  .findViewById(R.id.*gallery*);  gallery.setVisibility(View.*GONE*);  TextView camera = (TextView) view.findViewById(R.id.*camera*);  camera.setVisibility(View.*GONE*);  **final** Dialog dialog = **new** Dialog(MyMsgActivity.**this**,  R.style.*myDialogTheme*);  dialog.setContentView(view);  dialog.show();  Button cancel = (Button) view.findViewById(R.id.*button\_cancel*);  cancel.setVisibility(View.*VISIBLE*);  cancel.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  dialog.dismiss();  }  });  Button ok = (Button) view.findViewById(R.id.*button\_ok*);  ok.setVisibility(View.*VISIBLE*);  ok.setOnClickListener(**new** OnClickListener() {  @Override  **public** **void** onClick(View v) {  dialog.dismiss();  adapter.setMsgList(**null**);  adapter.notifyDataSetChanged();  HashMap<String, Object> params = **new** HashMap<String, Object>();  params.put("username", username);  HttpHelper.*asyncPost*(Constants.*URL* + "/second-hand/msg\_delete.do", params, **new** Callback() {    @Override  **public** **void** dataLoaded(Message msg) {  **if**(HttpStatus.*SC\_OK* != msg.what){  AppException.*http*(msg.what).makeToast(  MyMsgActivity.**this**);  **return**;  }**else** {  Toast.*makeText*(MyMsgActivity.**this**,  msg.obj.toString(),  Toast.*LENGTH\_SHORT*).show();  }  }  });  }  });  }  }); |

## 4.4 客户端与服务端通信

本系统是基于客户端和服务器的C/S结构，客户端需要大量的向服务器提交数据、验证数据和解析数据。客户端服务器的通信主要是基于Android平台的HttpPost来实现，在本系统中使用的是第三方的HttpHelper框架，其原理和Android平台内置HttpPost相同，由它向服务端发送请求和接受响应，使用JSON数据格式来交互。因为此次使用的是第三方封装好的框架所以在编译的时候会有相关支持库，第三方框架的使用和正常的类的使用如出一辙。在此就不赘述了。

服务器端使用Java来封装JSON，包括单一的JSON对象和带数组的JSON对象。

在客户端与服务端通信时，为了不阻塞用户UI界面，一般会在主线程中另开子线程来进行网络的请求的异步操作。获得数据后，再通知主线程进行处理。因为本系统使用的是封装好的框架，所以只需要按照框架的格式将特定参数传递到相关的方法中即可。

## 4.5 系统测试

为了保证系统开发实现后，与之前所定义的功能相符，并且保证系统能正常的运行，那么就需要对系统进行测试。

### 4.5.1 测试环境

因为本系统客户端的正常运行离不开服务端的支持所以在此也将对服务端进行测试，服务端的测试环境如表4-5所示：

表4-5 服务端测试环境

|  |  |  |
| --- | --- | --- |
| **软件环境** | 操作系统 | Windows10 |
| Java版本 | JDK1.8，JRE 8 |
| Tomcat版本 | 7.0 |
| MySQL数据库版本 | MySQL Server 5.7.9 |
| **硬件环境** | CPU | Intel Core i5-3230 2.6GHZ |
| 内存 | 6.00GB |

客户端的测试环境为PC端上的Genymotion，它可以模拟虚拟的Android手机来运行相应的应用程序，具有简洁、高效等优点。本次测试的模拟器Android系统版本从4.2.2-4.4.2。

### 4.5.2 功能测试

功能测试主要是测试系统预定功能是否完善可用，按照上面章节所分模块分别进行对这几个主要功能进行测试。

1. 商品交易模块

主要需要测试商品信息上传、商品列表展示、详细信息浏览、下拉刷新、图片加载等功能。看商品是否能够正常的上传并展示出来。测试结果如表4-6所示：

表4-6 商品交易模块测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 测试方法 | 期望结果 | 结果 |
| 1-1 | 商品列表页下拉刷新 | 获取最新增加的信息并以列表的方式展示出来 | 通过 |
| 1-2 | 点击发布按钮 | 进入发布信息编辑详情页 | 通过 |
| 1-3 | 编辑商品信息并上传 | 1. 能获取用户输入的文字信息、图片信息。 2. 能对图片进行处理 3. 在网络环境正常的情况下能成功上传至数据库 | 通过 |
| 1-4 | 点击商品单个列表 | 出现商品的详细信息，包括商品的图片，标题、描述、价格、卖家联系方式等 | 通过 |

商品交易模块的部分测试界面如图4-7所示：从左至右依次是本系统主界面、商品浏览界面、商品发布界面。

图4-6 商品交易模块测试界面

1. 用户管理模块

用户管理模块，主要是为用户提供一必要的功能，包括用户的注册，用户的登录，个人中心、发布商品的管理，收藏商品的管理、求购信息的管理、留言的管理等。具体的测试如下：

表4-7 用户管理模块测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 测试方法 | 期望结果 | 结果 |
| 2-1 | 登录 | 在用户使用需要登录的功能时能够显示登录提示框点击确定后能够跳转到登录界面 | 通过 |
| 2-2 | 注册 | 点击注册按钮后能够跳转到注册详细信息页 | 通过 |
| 2-3 | 点击个人中心 | 能够跳转到个人中心的页面 | 通过 |
| 2-4 | 点击我的收藏 | 能跳转到展示用户所有收藏商品的页面 | 通过 |
| 2-5 | 点击我的发布 | 能跳转到展示用户发布的所有的商品的页面 | 通过 |
| 2-6 | 点击我的留言 | 能跳转到展示用户所有的留言的页面 | 通过 |
| 2-7 | 点击我的求购信息 | 能跳转到展示用户所有的求购信息的页面 | 通过 |
| 2-8 | 我的收藏页面删除和刷新功能 | 能展示用户所有收藏的商品点击删除按钮能够删除收藏商品点击刷新能获取到最新的信息并展示出来 | 通过 |
| 2-9 | 我的发布页面发布、删除、刷新功能 | 能展示用户所有发布的商品，点击发布按钮能够跳转到发布详情页，点击删除按钮能够删除发布的商品，点击刷新功能能够将最新的数据展示出来 | 通过 |
| 2-10 | 我的留言页面删除和刷新功能 | 能够展示用户所有的留言，点击删除按钮能够删除留言，点击刷新能够将最新的数据展示出来 | 通过 |
| 2-11 | 我的求购信息页面发布、删除、刷新功能 | 能够展示用户所有发布的求购信息，点击发布按钮能够跳转到求购信息发布详情页，点击删除按钮能够删除求购信息，点击刷新能够获取最新的数据并且展示出来。 | 通过 |

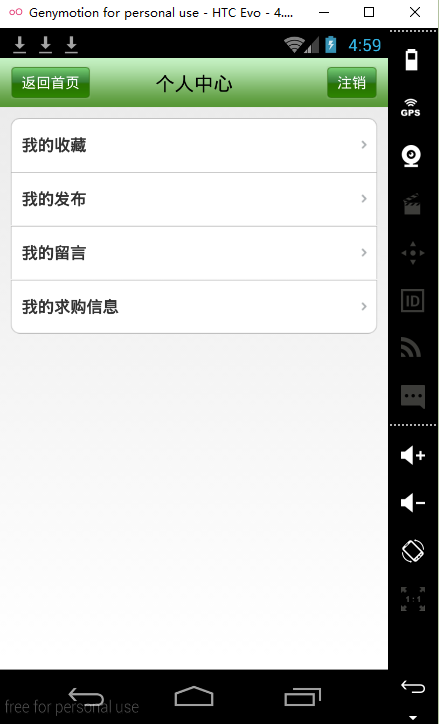
用户管理模块的部分测试界面图如图4-8所示，从左至右依次是登录界面、注册界面、个人中心界面。

图4-8 用户管理模块测试界面

1. 用户交流模块

用户交流模块主要是为用户提供方便的交流功能，包括打电话、发短信、留言等。具体的测试如下表：

表4-9 用户交流模块测试表

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 测试方法 | 期望结果 | 结果 |
| 3-1 | 打电话 | 点击打电话按钮能够跳转到系统打电话的界面并且自动填写卖家留下的电话号码 | 通过 |
| 3-2 | 发短信 | 点击发短信按钮能够跳转到系统发短信的界面并且自动填写卖家留下的电话号码 | 通过 |
| 3-3 | 留言 | 点击留言按钮能够跳转到留言编辑详情页 | 通过 |
| 3-4 | 发送留言 | 点击留言详情页的留言按钮能够将留言成功发送留言 | 通过 |

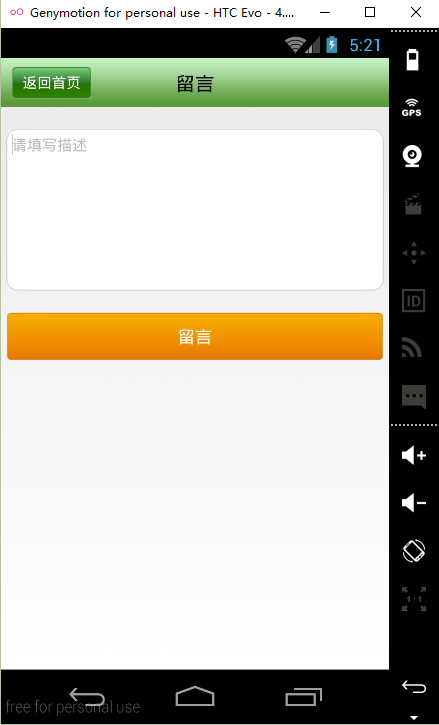
用户交流模块测试部分页面展示，如图4-10所示，从左只有依次是商品详情页，留言编辑页。

图4-10 用户交流模块测试页面

经过上面对各个功能模块的测试，可以看出本系统的功能基本上正常运行，基本符合系统的设计要求。

# 5. 结论

时间过的真快，现在回想起来折腾毕业设计也有半年时间了。刚开始的时候不知道从哪开始动手。到现在走完整个软件开发的一整个周期，真的受益匪浅。

在完成此次毕业设计的过程中对我开发能力也是一次非常大的提升。总结下来是一下几点：一、熟悉了整个软件的开发周期。二、知道如何对技术进行总结归纳。三、懂得了如何使用第三方开发框架。四、了解了基本的测试环节。五、提升了文档的编写能力。

在这之前，如果我们希望开发一款软件，我会马上打开开发工具就开始写代码。现在我开始学着对软件进行需求分析，市场调研。然后开始设计软件的功能。然后完成相关功能的设计以及相关技术的详细了解，接着才会着手编写代码实现这些功能。在功能基本完成后对软件进行基本的测试，排除能找的程序BUG。

以前写代码总是写一段是一段，项目做一个是一个。做完就不管了。等到以后需要这方面的功能的时候总是重新再思考重新设计代码，这样不仅浪费时间，也使得自己的编码效率非常的低，我觉得正确的方式应该是编写一个功能模块应该保留其Demo以及详细的代码解释，等到后面需要开发同样功能或者相关功能的时候可以以此为基础进行拓展。这样不仅方便管理自己写过的代码，也可以定时改进这些代码提高运行效率，提高自己的编码水平。

完成这个项目开发之前基本上没有接触过第三方的框架。这让我对我所学习的专业感到了深深的无知。虽然老师也不曾在课堂上着重讲这些东西，但这些东西在实际开发当中非常重要。虽然我们在课堂上学习了几乎所有的基础知识，但这些在实际开发中并不能完全解决问题，就相当于老师教会我们怎么钓鱼却没有教我们如何根据不同鱼类选择不同的渔具。也许能够做同一件事，但是肯定不是将最适合的资源用在最适合的位置上，也许跟本就不能解决问题。

软件测试这个概念在这之前是没有的，只有简单的debug，也不曾知道软件测试是一个系统化的过程。经过这次开发过程，我学会了基本的软件测试，知道了怎样去测试一个软件基本的功能，一个软件应该具有怎样稳定性。虽然这不是软件开发者的工作，但是还是得了解这个过程，以便以后工作中更好的与团队共同完成一款软件的整个开发流程。

完成此次毕业设计之前，我文档编写的能力非常的有限，总是不知道该如何设计编写一片完整的开发文档。现在我知道如何设计一篇开发文档的篇幅以及文档应该有哪些内容，整体的排版和布局应该怎么做。这是对我整理工作能力的很大提高。

经过此次毕业设计，将我大学所学的专业知识进行了一次非常有实际意义的综合训练，为我以后在工作中出色的完成工作任务打下了扎实的基础。

# 致谢

首先感谢学校能有这么一次机会，让我们对自己所学的专业知识进行一次实际的应用。这为我日后从事计算机相关工作提供了经验，奠定了基础。

本次毕业设计持续了一个学期的时间，现在也算是比较完美的结束了。回想我刚刚确定这个选题的时候是那么的兴奋和不安。不过这些都已经不重要了，重要的是花费了一个学期的时间终于完成了这项艰巨的任务。

然后，我要特别感谢指导老师对我悉心的指导，不管我问什么样的问题指导老师总是能够给我满意的答案或者教我寻找答案的方法。老师渊博的知识、严谨的作风、诲人不倦的态度和学术上精益求精的精神让我受益终生。

其次，学校方面也非常的重视，给我们提供了很大的支持和帮助，还专门开设了相关的课堂供我们学习和交流还给我们提供了专门的指导老师帮助我们顺利完成整个设计。对于学校和老师为我的毕业设计所提供的极大帮助和关心，在此我致以由衷的感谢！

另外，我也要感谢同学对我的关心和支持，感谢各位老师在学习期间对我的严格要求。同时也感谢身边朋友的热心帮助和宝贵的建议。

最后，我要向我服务致以最崇高的敬意，没有你们无私的支持，就没有我今天的成绩！

# 参考文献

1. Cay S. Horstmann,Cray Cornell.《Java核心技术：卷一》（原书第九版）.机械工业出版社,2014年12月:1-701.
2. Cay S. Horstmann,Cray Cornell.《Java核心技术：卷二》（原书第九版）.机械工业出版社,2014年12月:1-698.
3. BruceEckel.《Java编程思想》. 机械工业出版社,2013年10月:1-378.
4. FLANAGAN.《Java技术手册》. 中国电力出版社,2012年6月:1-465.
5. 孙一林,彭波.《Java数据库编程实例》. 清华大学出版社,2012年8月:30-210.
6. 郭霖.《第一行代码---Android》.人民邮电出版社,2015年4月：1-552.
7. 中国互联网信息中心.第36次中国互联网络发展状况统计报告.2015年
8. CSDN博客<http://blog.csdn.net/>