题 目：停车场管理手机客户端软件—登陆与车位预约模块设计

**目录**

摘要 3

ABSTRACT 4

引言 5

概述 5

2.1 JAVA介绍 7

2.1.1 面向对象思想 7

2.1.2 JAVA的优势 7

2.1.3 JAVA的特性 7

2.2 Android介绍 8

2.2.1 Android简介 8

2.2.1 四大组件，五大布局以及生命周期 8

2.2.2 Android基本控件 9

2.2.3 网络通信 10

2.3 OSX操作系统下搭建Android开发环境 11

2.3.1 JDK,ADT,Android Studio的下载及安装 11

2.3.2 Genymotion的下载及创建模拟器 12

2.4 Git介绍 14

2.4.1 什么是版本控制 14

2.4.2 在Github上创建仓库 14

2.4.3 使用Android Studio提交代码 15

3 软件的设计与结构 17

3.1 课题介绍 17

3.2 优秀开源项目 17

3.2.1 Fresco 17

3.2.2 OKHttp 18

3.2.3 Gson 18

3.2 MVC 18

3.2.1 MVC简介 18

3.2.2 MVC的优点 18

3.2.3 MVC在本课题中的使用 19

3.3 功能需求与界面设计 19

3.3.1 所需功能 19

3.3.2 界面展示 21

3.4 项目框架搭建 23

3.4.1 项目结构图 23

3.4.2 封装网络请求 25

3.4.3 定义常量 27

4 程序实现及结果 28

4.1 登录与注册 28

4.2 首页 29

4.2.2 查询预约页 29

4.2.3 个人中心页 30

4.3 个人信息完善 31

4.4 个人车辆信息的维护 31

4.4.1 查询 31

4.4.2 增加 31

4.4.3 修改 31

4.4.4 删除 31

4.5 停车记录 32

4.6 修改密码 32

4.7 退出登录 32

4.8 预约车位 32

4.8.1 选择停车车位 32

4.8.2 选择停放车辆 32

4.8.3 选择停车时间 33

4.8.4 预约结果 33

4.9 取消预约 33

4.10 结束停车 33

4.11 推送 33

4.11.1 在推送平台创建应用 33

4.11.2 集成SDK 34

4.11.3 处理推送消息 38

4.11.4 推送结果 38

4.12 build.gradle 39

4.13 结果展示 40

5. 总结与展望 48

致谢 49

参考文献 50

摘要

近些年来，随着基于智能移动端应用软件数量的增多，移动端的硬件水平和网络速度也有了很大程度的提高，人们对于智能手机的需求也日益增多。随着国内经济的快速发展，私家车如今已经高度普及，但是由此人们面临着寻找车位的问题。本课题主要是想让用户通过Android手机软件，预订停车车位，从而解决找不到车位的问题，为用户节省了大量的宝贵时间，也在极大程度上缓解了交通压力。因此，本软件具有极大的现实意义。

Android具有很好的开源性,便于开发和使用并且普及范围很广。本软件是基于Android Studio平台开发一个停车场管理的Android手机客户端软件。本课题主要实现客户端中登录与停车预约模块，主要有用户注册登录、完善个人信息、查询车辆、预约车位、结束停车的界面，查询停放车记录，推送等功能。在与停车场结合后，用户可以随时随地预约车位，使找停车位不再成为一个烦恼，用户还可以查看每次停车时长与支付费用，不再不明不白的花钱。

关键字：Android、停车场、软件

ABSTRACT

In recent years, with the increase number of applications based on smart mobile phone, hardware and network in mobile terminal has also been greatly improved, people's demand for smart phones is also increasing. With the rapid development of the domestic economy, private car are now highly popular, but people are thus faced with the problem of finding parking lots. The main issue in this topic is to make the user booking parking spaces by the Android application. So as to solve the problem that can not find parking spaces,save a lot of time for the user and also greatly ease the traffic pressure.So, this software has great practical significance.

Android has a very good open source, easy to develop and use, and widely spread.This Android software is based on the Android Studio platform to develop a parking lot management mobile client sofrware. This subject mainly achieved user regist,login,improve personal information,check user’s cars,booking parking spaces,end the parking,.check parking record,push message and so on. With the real parking lot combination, users can booking parking spaces in anytime,to make find a parking space will no longer be a worry, users can also view parkinglot time and payments int each record, no longer spend money without apparent cause.

Keywords: Android, parking lot, software

引言

随着移动互联网的发展以及Android设备的普及，开发出一款实用并且用户体验良好的App是当前开发者们所注重的。而Android Studio开发平台更是Google亲自推出的新的Android开发工具，采用了Google用于智能化构建项目的构建集成工具，让开发变得高效，便捷。

本软件从目前繁华地段停车困难的现状入手，分析用户的需求，设计出了帮助用户提前预约车位的系统。目前大部分停车场都只能在车开到停车场时才能知道该停车场能否停车，如果停车场已停满，车主只能继续寻找其他停车位，这不仅耽误了大量车主的时间，也在一定程度上最交通状况造成了压力。本次设计的停车场管理系统就是为了尽可能的解决这个问题。本课题就是完成这个系统的手机客户端的设计与开发。停车场只需要接入这个系统，用户只需要经过简单的注册登录，完善基本信息后，便可以随时随地预约车位，不仅极大地节约了停车场的人力成本，也节约了车主的时间。

本论文全面详细地介绍了实现本课题需要用到的基础技术、软件的设计与功能结构、在开发时的逻辑处理以及详细代码的编写和最终的实现结果。从章节上来看，第一章概括了文章结构，第二章介绍了实现软件需要用到的相关技术，第三章介绍了软件的架构以及相关界面的展示，第四章详细说明了软件的具体实现过程以及实现结果，第五章为课题总结。

# 概述

1.1 移动互联网的发展

目前,随着长期演进技术的快速发展,互联网逐渐从固定端到移动终端,这使它在各种各样的地方、任意时间,通过各种方式为用户提供网络服务。全球进行了与它相关研究的工作,IETF已经成立了一个团队来开发相关的技术。我国也进行了很多相关的项目,用来促进移动互联网的发展,与此同时也获得了大量的成就,如2004年北京交通大学开发了关键设备IPv6无线／移动路由器等。但是总的来说,移动互联网的研究仍处于起步阶段,有很大的发展空间，这也为中国的信息网络的发展提供了一个机会。

1.2应用前景

随着国内经济的高速发展，私家车的普及成爆发式增长，很多城市已经出现了“找车位难”的问题，与此同时大量的私家车也开始非法占用非机动车道路，这给城市的交通带来了极大的不便。本软件旨在解决以上问题，所以有非常大的应用前景。

1.3 主要内容

近些年来，私家车的数量急剧增加，人们对于停车场管理的要求不断增加。随着移动互联网的发展以及智能终端设备的迅速普及，使得开发基于停车场智能管理的移动客户端软件成为可能。在当前快节奏的生活中，时间越来越成为人们所关注的话题，本软件的优势便在于节约车主的时间以及停车场的人力成本。本次开发的需要研究的内容有：

1、学习Android原生控件的使用。

2、学习View的绘制流程以及开发自定义View满足特殊的视觉效果。

3、了解Metrial Design的设计理念，并在应用开发中体现出来。

4、学习Android线程间的通信。

5、学习网络协议，tcp、udp、http。

1.4 实现思路

完成本课题，具体的实现思路如下:

1. 搭建Android开发环境，安装开发工具。
2. 分析需要完成的功能，搜集相关资料，设计相应的界面，搭建项目框架。
3. 整个项目主要通过Http协议的post方法与服务器通信，客户端解析服务器返回的Json并处理，完成用户注册登录、完善个人信息、查询车辆、预约车位、结束停车的界面，推送等功能。还需要封装Http请求，完成项目中需要的工具类，尤其是有关时间的方法，以确保用户能够正确地预约车位。
4. 性能调优，在多图列表界面，采用压缩图片的方法，使图片内存占用减少，对列表（ListView）的每个Item进行重用即懒加载，避免OOM。在容易发生崩溃的地方采用try，catch捕获异常，避免应用直接崩溃等等。

**2 技术基础**

## 2.1 JAVA介绍

### 2.1.1 面向对象思想

面向对象的基本哲学是认为世界是由各种各样具有自己的运动规律和内部状态的对象所组成的；不同对象之间的相互作用和通讯构成了完整的现实世界。因此，人们应当按照现实世界这个本来面貌来理解世界，直接通过对象及其相互关系来反映世界。这样建立起来的系统才能符合现实世界的本来面目。

### 2.1.2 JAVA的优势

Java 编程语言是个简单、面向对象、分布式、解释性、健壮、安全与系统无关、可移植、高性能、多线程和静态的语言。

### 2.1.3 JAVA的特性

抽象类：规定一个或多个抽象方法的类别本身必须定义为abstract，抽象类只是用来派生子类，而不能用它来创建对象。

final类：又称“最终类”，它只能用来创建对象，而不能被继承，与抽象类刚好相反，而且抽象类与最终类不能同时修饰同一个类。

包：Java中的包是相关类和接口的集合，创建包须使用关键字package。

继承：Java作为面向对象编程语言，支持继承这基本概念。但Java只支持单根继承，java.lang.Object是所有其他类的基类。

多态类：在Java中，对象变量是多态的。而Java中不支持多重继承。

接口：Java中的接口是一系列方法的声明，是一些方法特征的集合，一个接口只有方法的特征没有方法的实现，因此这些方法可以在不同的地方被不同的类实现，而这些实现可以具有不同的行为。

封装：把数据和行为结合起在一个包中，并对对象使用者隐藏数据的实现过程，一个对象中的数据叫他的实例字段（instance field）。

重载：当多个方法具有相同的名字而含有不同的参数时，便发生重载。编译器必须挑选出调用哪个方法进行编译。

重写：也可称为方法的“覆盖”。在Java中，子类可继承父类中的方法，而不需要重新编写相同的方法。但有时子类并不想原封不动地继承父类的方法，而是想作一定的修改，这就需要采用方法的重写。值得注意的是，子类在重新定义父类已有的方法时，应保持与父类完全相同的方法头声明。

Class类：Object类中的getClass方法返回Class类型的一个实例，程序启动时包含在main方法的类会被加载，虚拟机要加载他需要的所有类，每一个加载的类都要加载它需要的类。

## 2.2 Android介绍

### 2.2.1 Android简介

Android是一种以Linux为基础的开放源代码操作系统，主要使用于便携设备。Android的系统架构和其它操作系统一样，采用了分层的架构。android分为四个层，从高层到低层分别是应用程序层、应用程序框架层、系统运行库层和linux核心层。

### 2.2.1 四大组件，五大布局以及生命周期

Android四大基本组件分别是

Activity：一个Activity通常就是一个单独的屏幕，它上面可以显示一些控件也可以监听并处理用户的事件做出响应。

Service(服务)：一个Service 是一段长生命周期的，没有用户界面的程序，可以用来开发如监控类程序。

Content Provider(内容提供者)：android平台提供了Content Provider使一个应用程序的指定数据集提供给其他应用程序。这些数据可以存储在文件系统中、在一个SQLite数据库、或以任何其他合理的方式,

其他应用可以通过ContentResolver类(见ContentProviderAccessApp例子)从该内容提供者中获取或存入数据。

BroadcastReceiver(广播接收器)：你的应用可以使用它对外部事件进行过滤只对感兴趣的外部事件(如当电话呼入时，或者数据网络可用时)进行接收并做出响应。广播接收器没有用户界面。

Android的五大布局分别是LinearLayout（线性布局）、FrameLayout（单帧布局）、RelativeLayout（相对布局）、AbsoluteLayout（绝对布局）和TableLayout（表格布局）。



图2.1 Android生命周期

### 2.2.2 Android基本控件

TextView：只读显示控件，可通过getText()获取其android:text属性、setText()设置其android:text属性。

Button：按钮控件，用户通过该控件来提交自己的请求，是用户与应用程序交互的最常用控件之一。

EditText：Android系统提供给用户输入的文本框。

ImageView：显示图片的控件，通过background、src分别设置背景图和前景图。

ProgressBar：进度条控件，可以向用户展示进度。

### 2.2.3 网络通信

#### 2.2.3.1 线程

Android中线程分为2种，主线程(UI线程)和子线程，所有有关UI的操作都必须在主线程中执行，因为若果多个线程都可以直接修改同一个控件的属性，很容易造成混乱，造成读写不一致，所以Android只允许将对控件的修改操作提交给控件所在的线程，由该线程去执行实际的修改动作。而比较耗时的操作如果放在主线程中执行则会造成ANR（Application Not Responding）影响用户体验，所以耗时操作都由子线程执行，最后由Handler机制让主线程更改UI。

thread,handler，looper，messageQueue

简单来讲，主线程创建是会默认创建一个looper，而looper创建是会自动创建一个messageQueue,其他时候可以通过looper.prepare()方法创建。thread通过handler.sendMessage()将message传递给looper，looper再将message放入消息队列中。当Looper对象看到MessageQueue中含有Message，就将其广播出去。对应的handler 对象收到该消息后，调用相应的handler 对象的handleMessage()方法对其进行处理。

#### 2.2.3.2 TCP、HTTP协议

TCP:是一种面向连接的、可靠的、基于字节流的[传输层](http://baike.baidu.com/view/239605.htm)通信协议。

1.TCP的三次握手最主要是防止已过期的连接再次传到被连接的主机。

2.拥塞控制就是防止过多的数据注入网络中，这样可以使网络中的路由器或链路不致过载。

3.滑动窗口：所谓滑动窗口协议，自己理解有两点：1. “窗口”对应的是一段可以被发送者发送的字节序列，其连续的范围称之为“窗口”；2. “滑动”则是指这段“允许发送的范围”是可以随着发送的过程而变化的，方式就是按顺序“滑动”。

HTTP: 是一个[客户端](http://baike.baidu.com/view/930.htm)和[服务器](http://baike.baidu.com/view/899.htm)端请求和应答的标准（TCP）

Get: 根据HTTP规范，GET用于信息获取，而且应该是安全的和幂等的。

Post: 根据HTTP规范，POST表示可能修改变服务器上的资源的请求。

#### 2.2.3.3 JSON

JavaScript Object Notation 是一种轻量级的数据交换格式

JSON 语法是 JavaScript 对象表示语法的子集。数据在键值对中，数据由逗号分隔，花括号保存对象，方括号保存数组。

JSON 值可以是：数字（整数或浮点数），字符串（在双引号中），逻辑值（true 或 false），数组（在方括号中），对象（在花括号中），null。

## 2.3 OSX操作系统下搭建Android开发环境

### 2.3.1 JDK,ADT,Android Studio的下载及安装

JDK:在<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html下载对应自己系统的版本>。

ADT&Android Studio:在<https://developer.android.com/sdk/index.html下载对应自己系统版本的ADT和开发工具>。

安装:将Android Studio.app 拖入Application文件夹，打开Android Studio.app按照默认配置即可安装成功。安装完成后会自动下载ADT。



图2.2 安装Android Studio

配置ADB环境变量: 对于android studio而言，默认的adb路径为：

~/Library/Android/sdk/platform-tools

在终端输入以下命令

>touch .bash\_profile

>open -e .bash\_profile

将路径添加到文件

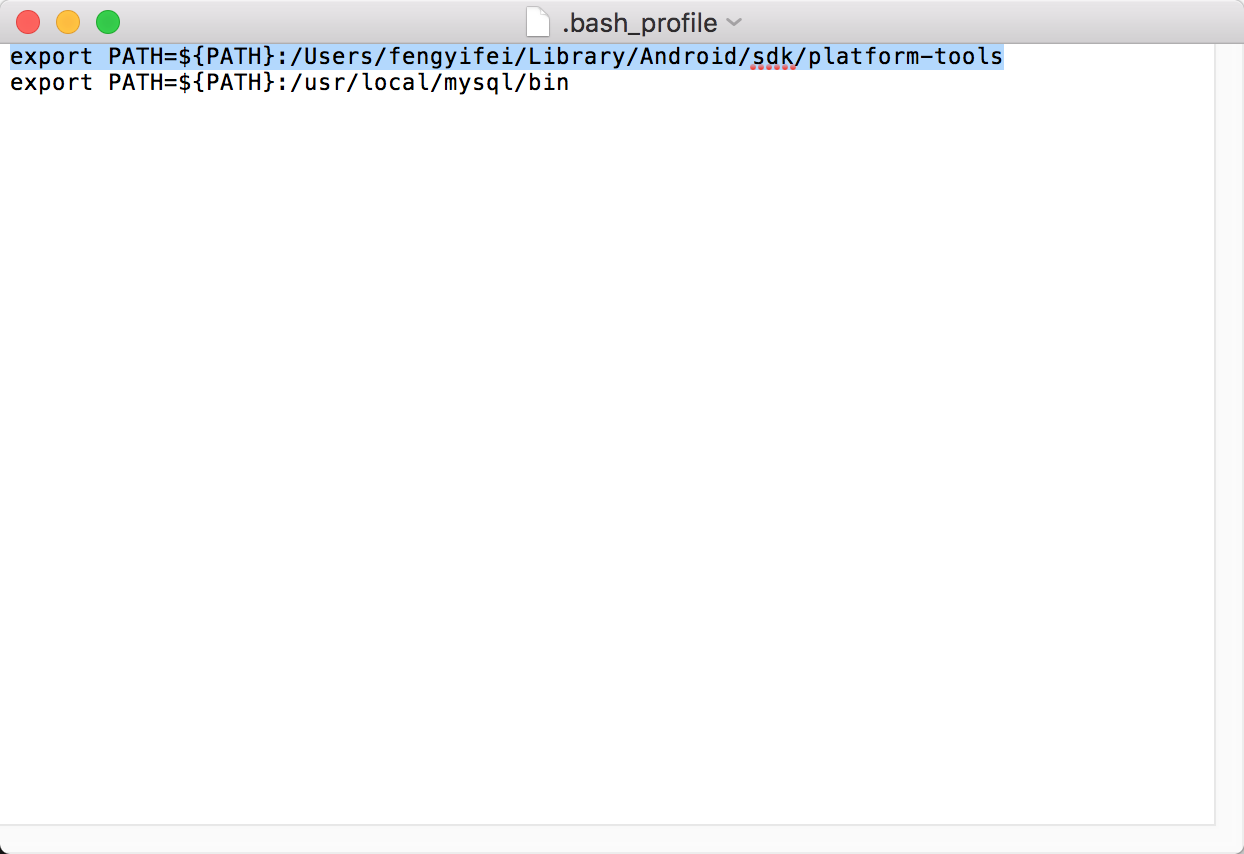


图2.3 配置adb环境变量

最后更新刚配置的环境变量，命令如下：

source .bash\_profile

### 2.3.2 Genymotion的下载及创建模拟器

Genymotion是一款Android模拟器，拥有非常流畅的界面和很快的启动速度，用在开发测试比方面原生的Android模拟器好很多。

在<https://www.genymotion.com/>注册一个用户，登陆后即可下载Genymotion，下载时选择个人用户。

安装完成后，在客户端登录刚才所创建的账户，点击Add按钮便可下载各个版本的模拟器。下载完成后选择一模拟器点击Start按钮即可。



图2.4 安装Genymotion

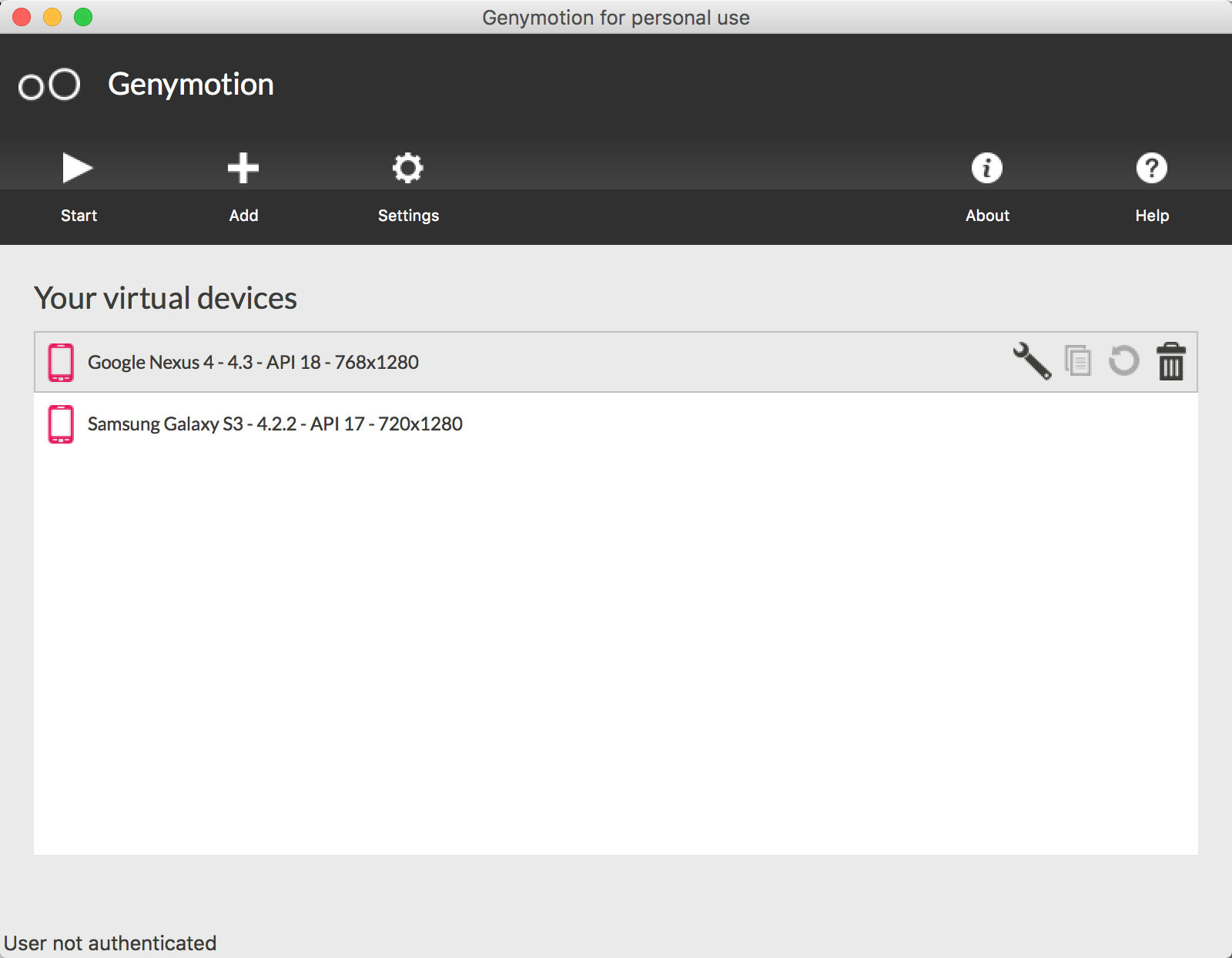


图2.5 配置Genymotion

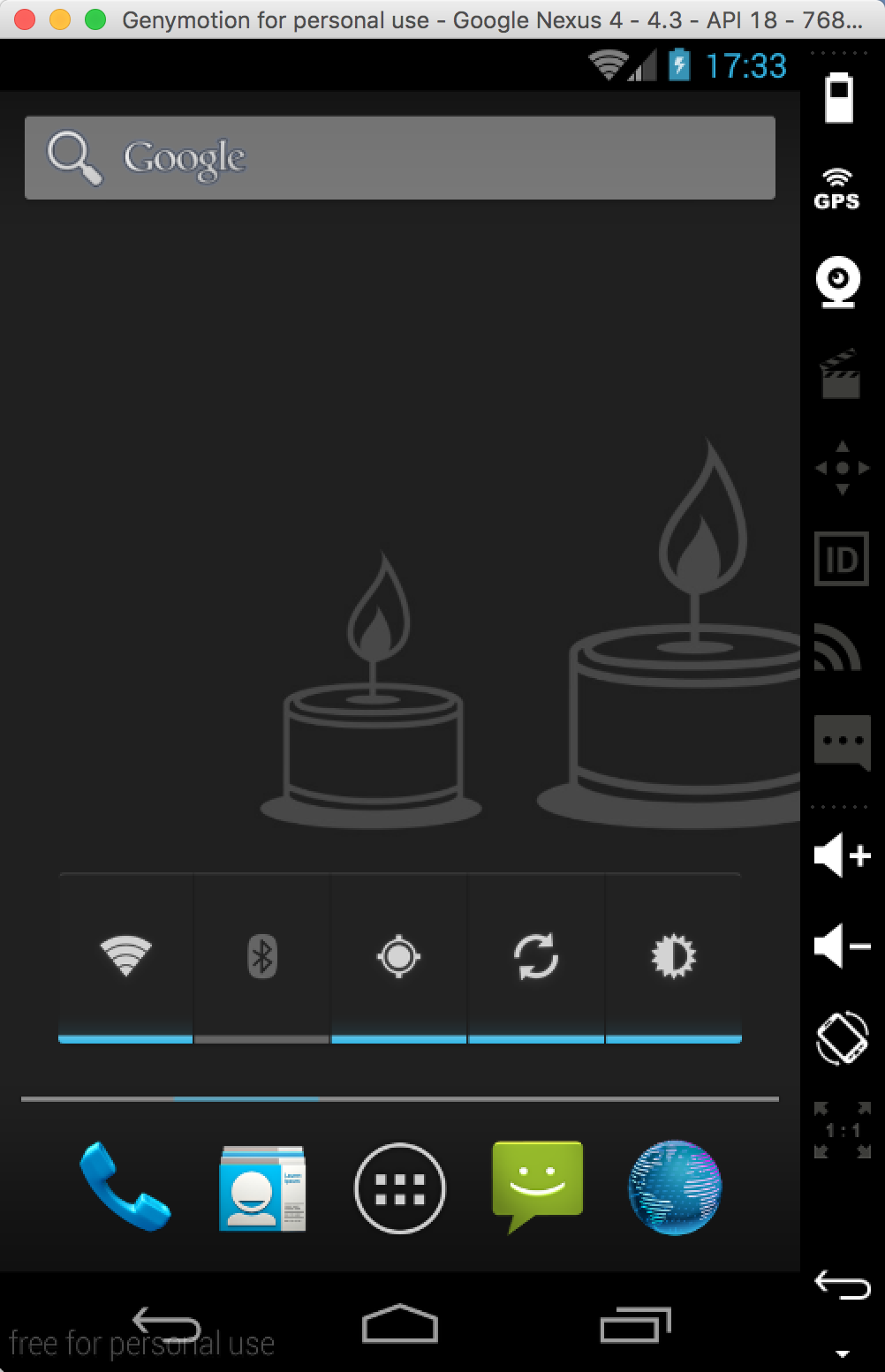


图2.6 Genymotion主界面

## 2.4 Git介绍

Git是一款分布式版本控制系统，可以克隆服务器上完整的Git仓库，在本地根据不同的需要创建分支，commit代码，最后再push到服务器。与svn相比，他的优势在于可以离线工作，即可以在没有网络的情况下commit代码。

github是一个基于git的项目托管平台，它提供了web界面，你可以在上面创建资源仓库来存放你的项目。

### 2.4.1 什么是版本控制

版本控制就是让你得到任何时间修改的版本。版本控制可以管理软件多人协同开发，不会导致软件互相覆盖。每次提交都会有相应的版本号和提交说明，方便回滚或者其他人阅读。

### 2.4.2 在Github上创建仓库

在<https://github.com/>上创建自己的账户，登陆后点击New repository

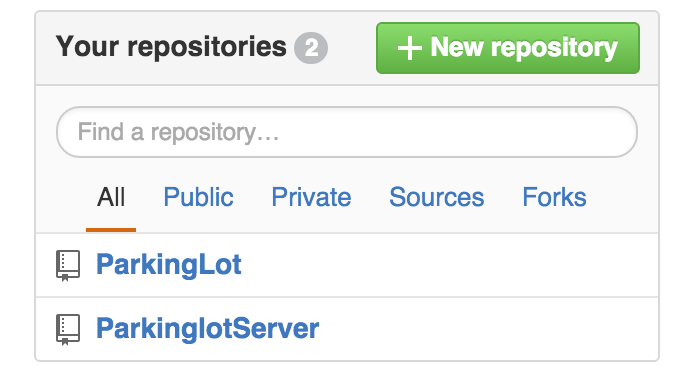


图2.7 Github首页

输入项目名称与描述，并选择Public，最后点击Create repository。

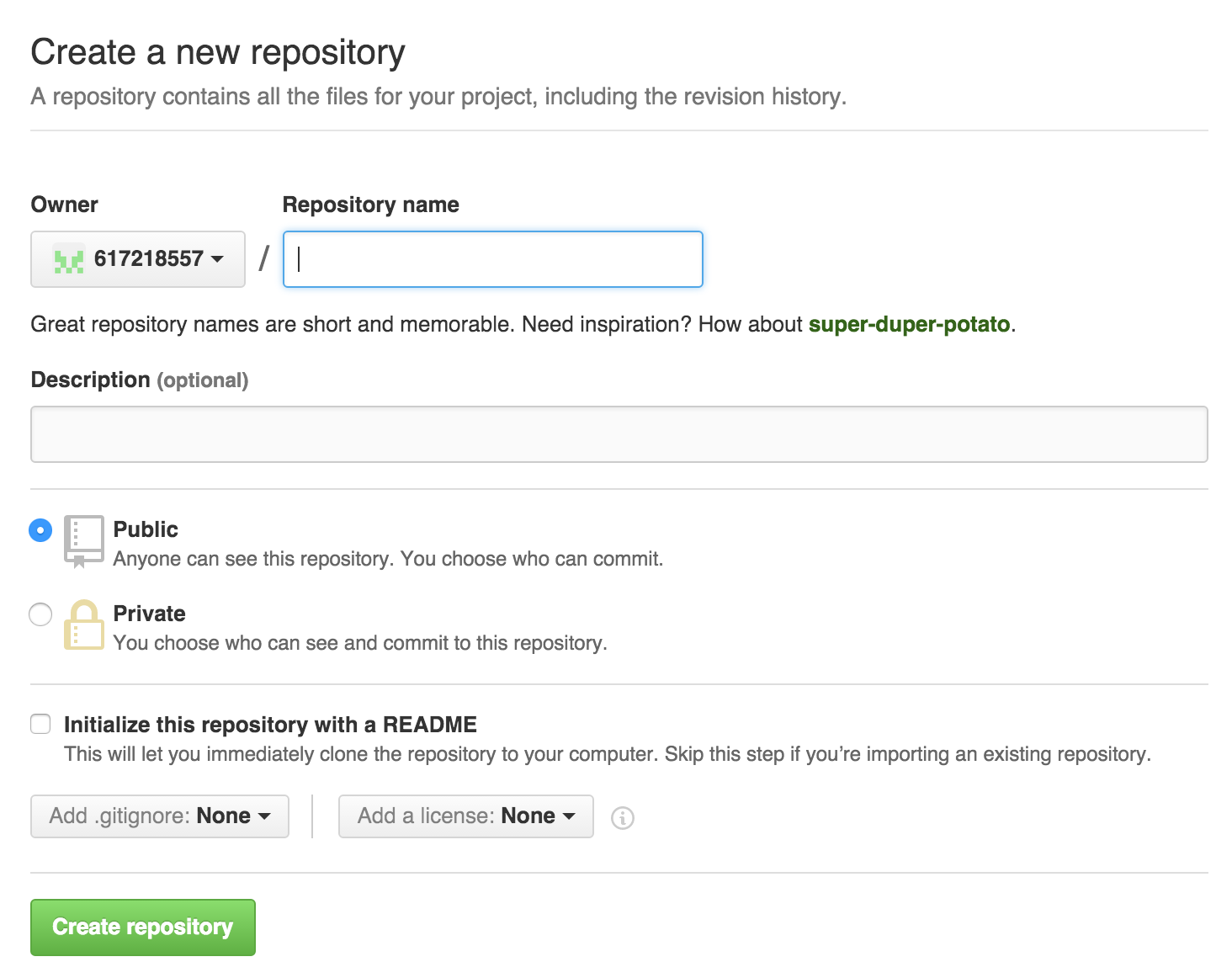


图2.8 Github 创建项目

### 2.4.3 使用Android Studio提交代码

打开Android Studio的偏好选项，选择Version Control下的Github,配置自己的Github账号。然后右键项目选择Git下的Commit Directory选项，在弹出的界面中填写Commit Message,点击Commit and Push即可提交代码。

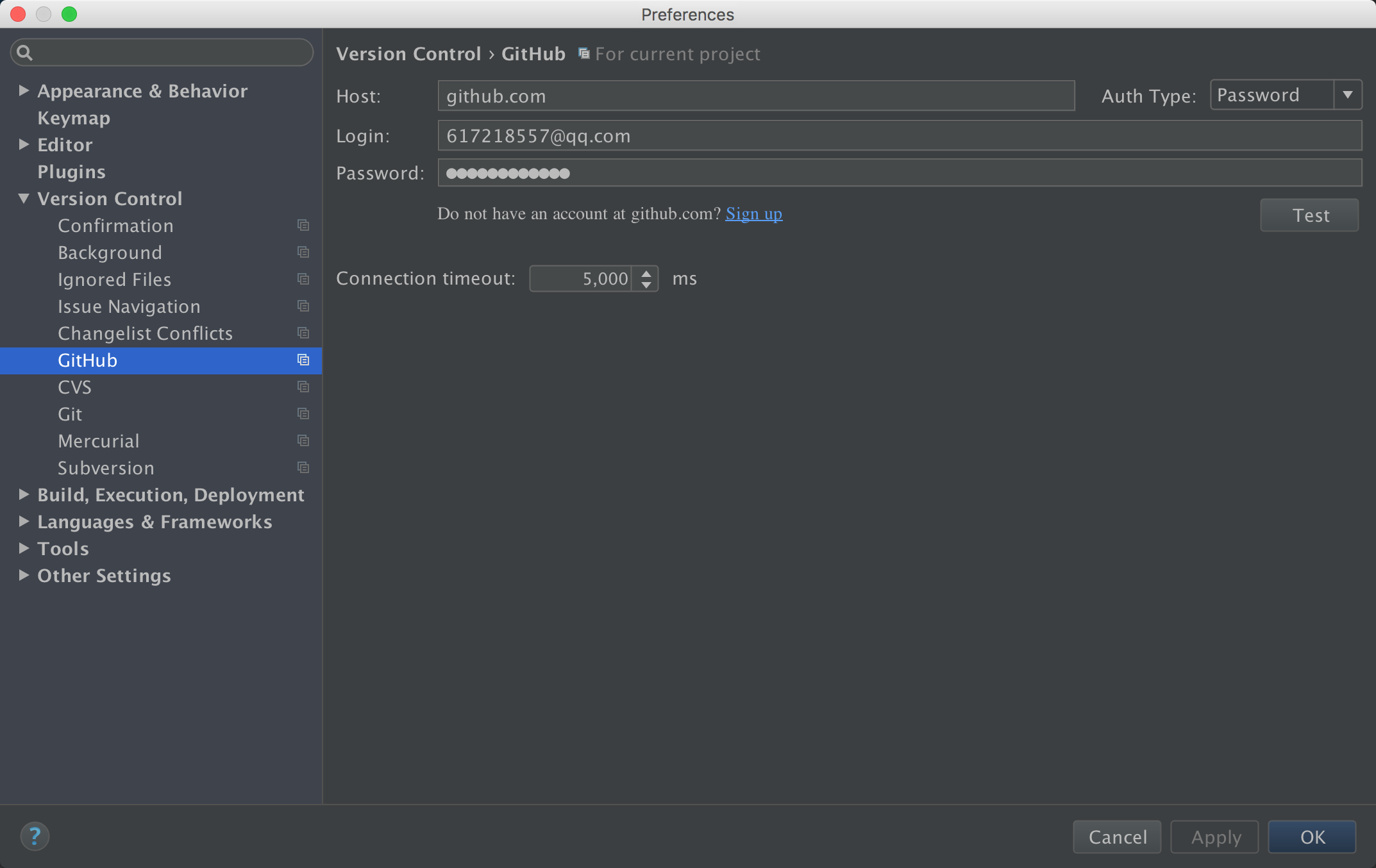


图2.9 Android Studio连接Github

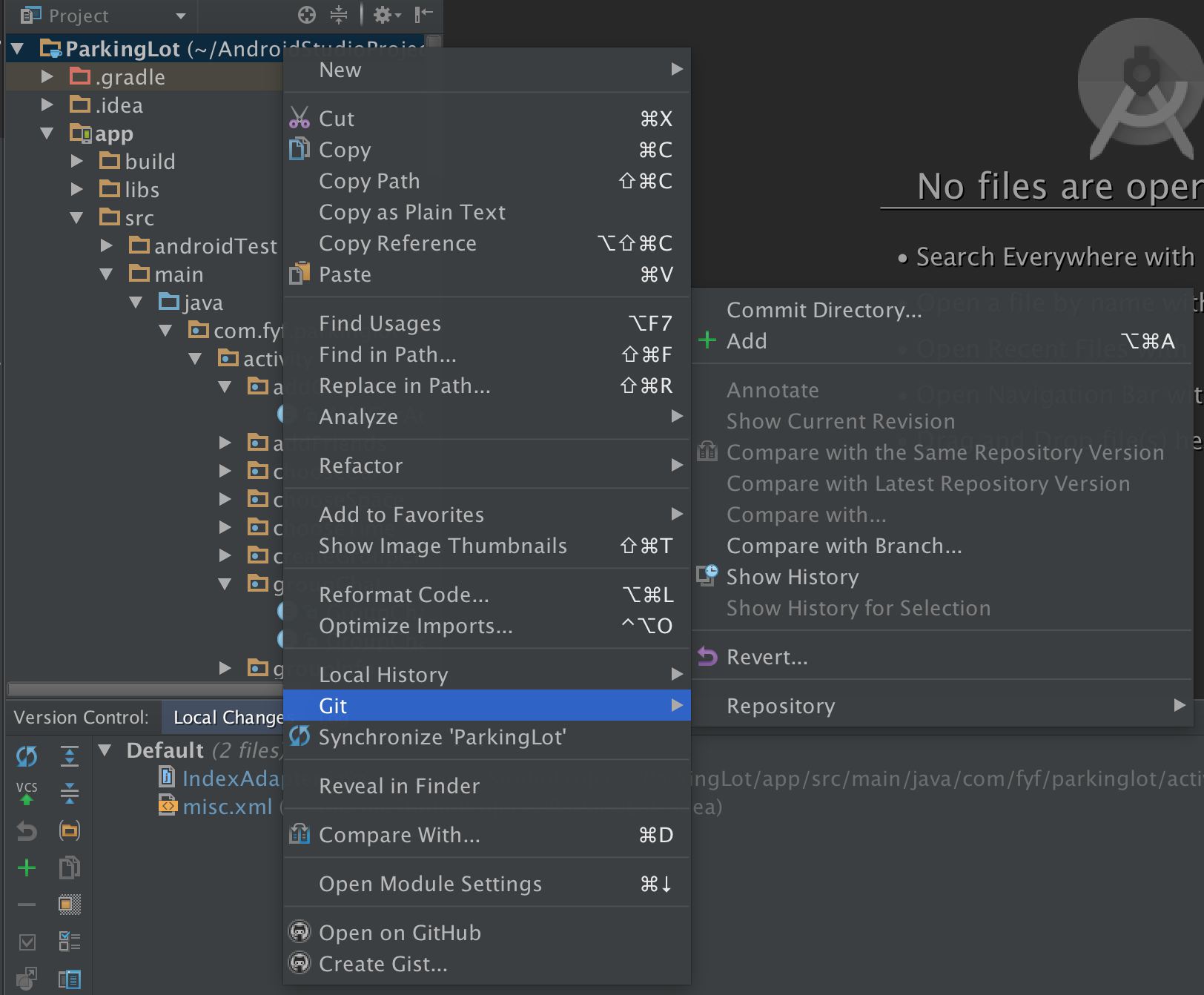


图2.10 Android Studio提交代码

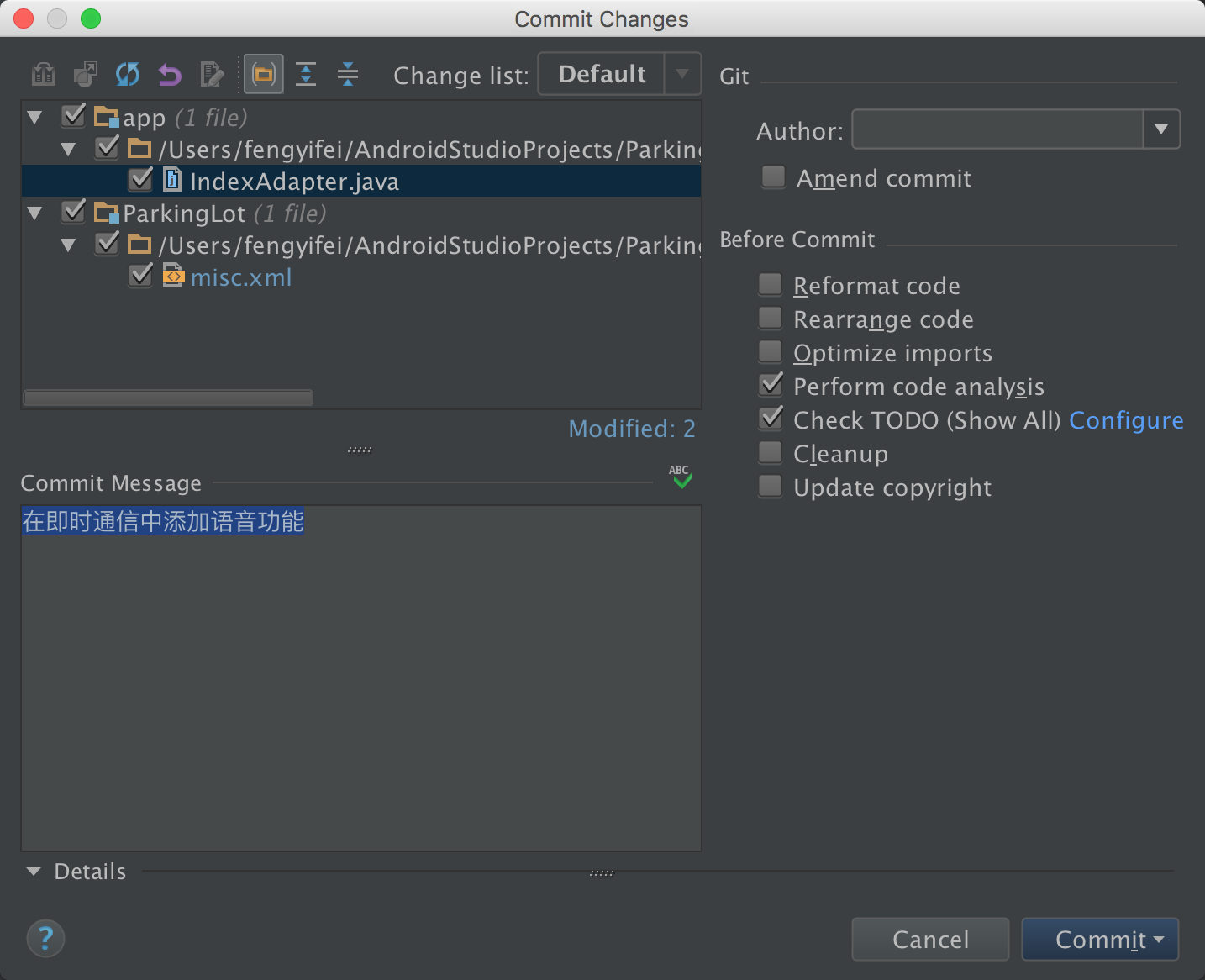


图2.10 Android Studio向Github Push代码

# 3 软件的设计与结构

## 3.1 课题介绍

随着宽带移动通信网络技术的快速发展、智能移动设备的迅速普及及硬件处理能力的不断增强，人们对基于移动终端的信息化管理软件应用需求日益增长。本课题是实心停车场管理的手机端客户软件，要求实现登陆与车位预约功能模块，包括系统架构设计、用户登陆、车位选择、车位预约、相关信息推送、与服务器端的通信等功能。

## 3.2 优秀开源项目

### 3.2.1 Fresco

Fresco是Facebook最新推出的一款用于Android应用中展示图片的强大图片库,可以从网络、本地存储和本地资源中加载图片。其优点有：

1.支持webp格式的图

2.Android5.0以下系统：使用”ashmem”（匿名共享内存）区域存储Bitmap缓存，这样Bitmap对象的创建、释放将永远不会触发GC，关于”ashmem”存储区域，它是一个不在Java堆区的一片存储内存空间，它的管理由Linux内核驱动管理，不必深究，只要知道这块存储区域是别于堆内存之外的一块空间就行了，且这块空间是可以多进程共享的，GC的活动不会影响到它。5.0以上系统，由于内存管理的优化，所以对于5.0以上的系统Fresco将Bitmap缓存直接放到了堆内存中。

3.使用了三级缓存：Bitmap缓存+未解码图片缓存+硬盘缓存。

4.Fresco中的MVC模式

DraweeView——View，DraweeController——Control，DraweeHierarchy——Model，它们之间的关系大致如下：

DraweeHierarchy意为视图的层次结构，用来存储和描述图片的信息，同时也封装了一些图片的显示和视图层级的方法。

DraweeView用来显示顶层视图（getTopLevelDrawable()）。DraweeController控制加载图片的配置、顶层显示哪个视图以及控制事件的分发。

### 3.2.2 OKHttp

OkHttp是一个高效的HTTP库，实现了几乎和java.net.HttpURLConnection一样的API。OKHttp总体设计主要是通过Diapatcher不断从RequestQueue中取出请求（Call），根据是否已缓存调用Cache或 Network这两类数据获取接口之一，从内存缓存或是服务器取得请求的数据。该引擎有同步和异步请求，同步请求通过Call.execute()直接返 回当前的Response，而异步请求会把当前的请求Call.enqueue添加（AsyncCall）到请求队列中，并通过回调（Callback） 的方式来获取最后结果。

### 3.2.3 Gson

GSON是Google开发的Java API，用于转换Java对象和Json对象。在JAVA对象中设置和JSON对象中相同的属性名称（大小写敏感），并且添加get、set方法，最后调用Gson.fromJson(json,Class)便可将json对象转换为java对象;

## 3.2 MVC

### 3.2.1 MVC简介

MVC全名是Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写，一种软件设计典范，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。

Model（模型）是应用程序中用于处理应用程序数据逻辑的部分。通常模型对象负责在数据库中存取数据。

View（视图）是应用程序中处理数据显示的部分。通常视图是依据模型数据创建的。

Controller（控制器）是应用程序中处理用户交互的部分。通常控制器负责从视图读取数据，控制用户输入，并向模型发送数据。

### 3.2.2 MVC的优点

MVC 分层有助于管理复杂的应用程序，可以在一个时间内专门关注一个方面。例如，可以在不依赖业务逻辑的情况下专注于视图设计。同时也让应用程序的测试更加容易。

MVC 分层同时也简化了分组开发。不同的开发人员可同时开发视图、控制器逻辑和业务逻辑。

它的优点有：耦合性低、重用性高、生命周期成本低、MVC使开发和维护用户接口的技术含量降低、部署快、可维护性高、分离视图层和业务逻辑层也使得WEB应用更易于维护和修改、有利软件工程化管理

### 3.2.3 MVC在本课题中的使用

本课题的项目架构中应用了类似于MVC的框架模式。M层：做一些业务逻辑处理，比如数据库存取操作，网络操作(HttpUtils)，复杂的算法(有关于时间的Utils)，耗时的任务等都在model层处理。 V层：应用层中处理数据显示的部分，XML布局可以视为V层，显示Model层的数据结果。 C层：在Android中，Activity处理用户交互问题，因此可以认为Activity是控制器，Activity读取V视图层的数据（eg.读取当前EditText控件的数据），控制用户输入（eg.EditText控件数据的输入），并向Model发送数据请求（eg.发起网络请求等）。

## 3.3 功能需求与界面设计

### 3.3.1 所需功能

本课题需要实现用户的登录，注册，预约车位(选择车位，选择车辆，选择时间)，取消预约，结束停车，查询停车记录，用户个人信息的显示与维护（头像，姓名，年龄，性别），修改密码，个人车辆信息的维护（添加，修改（图片，车型，车牌号），删除），退出登录，以及一些widgit和自定义布局。流程图如下：

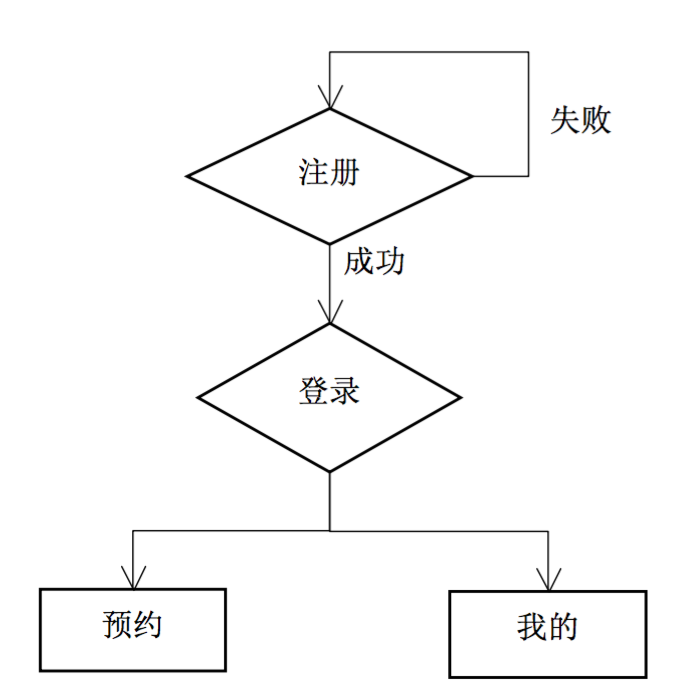


图3.1 登录流程图

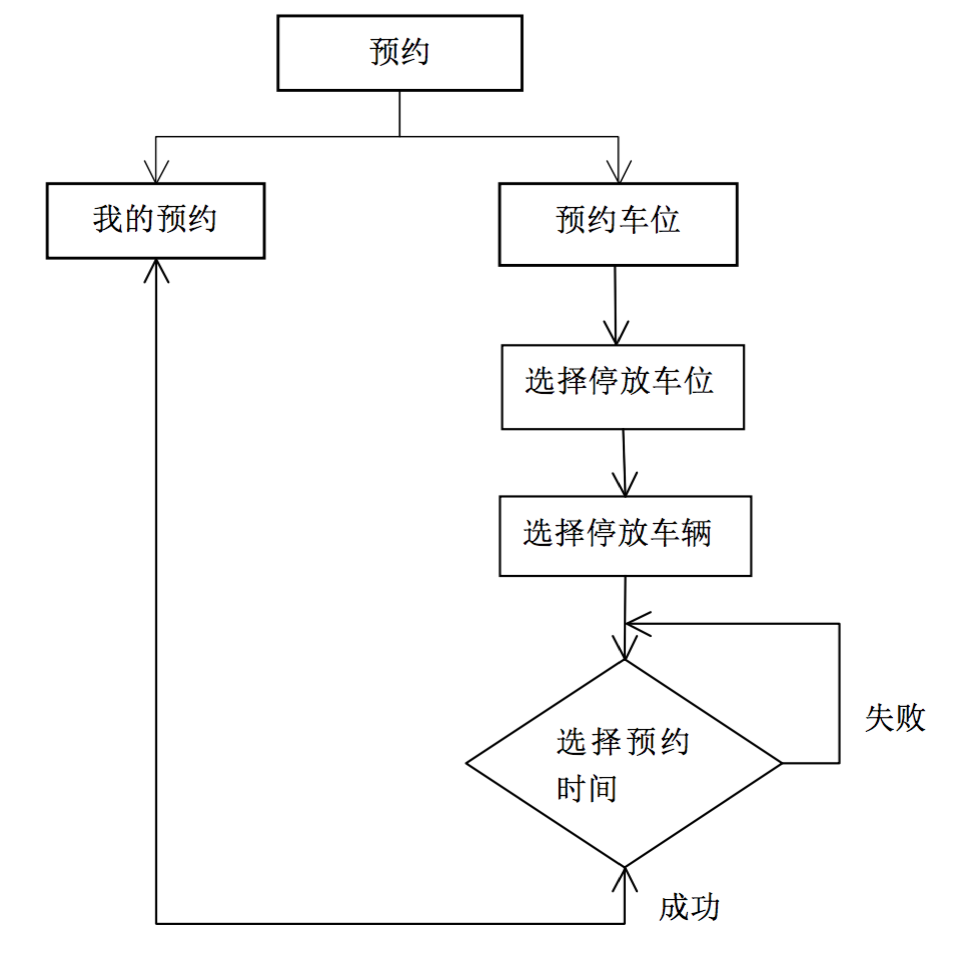


图3.2 预约车位流程图

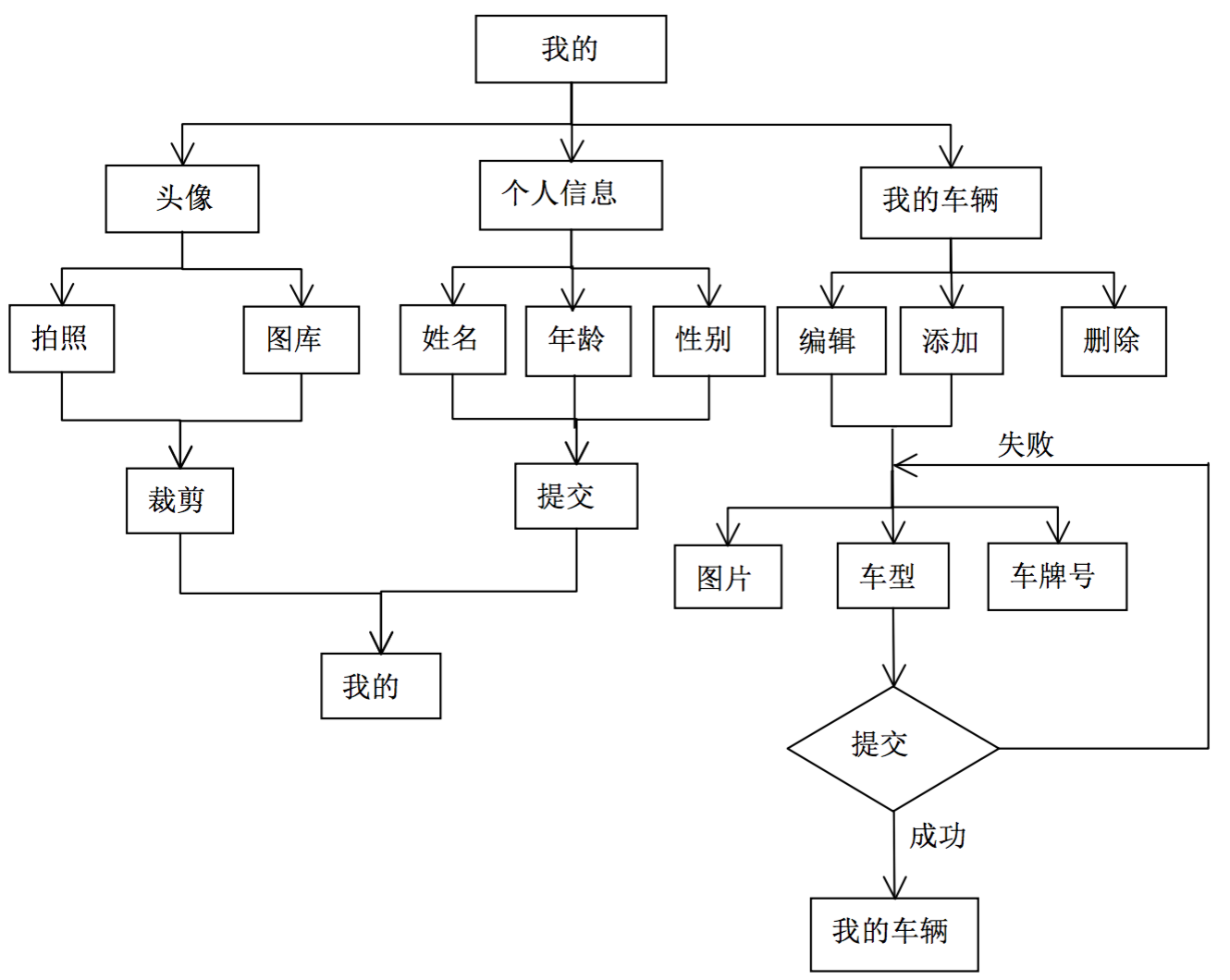


图3.3 个人中心中头像、个人信息、我的车辆流程图

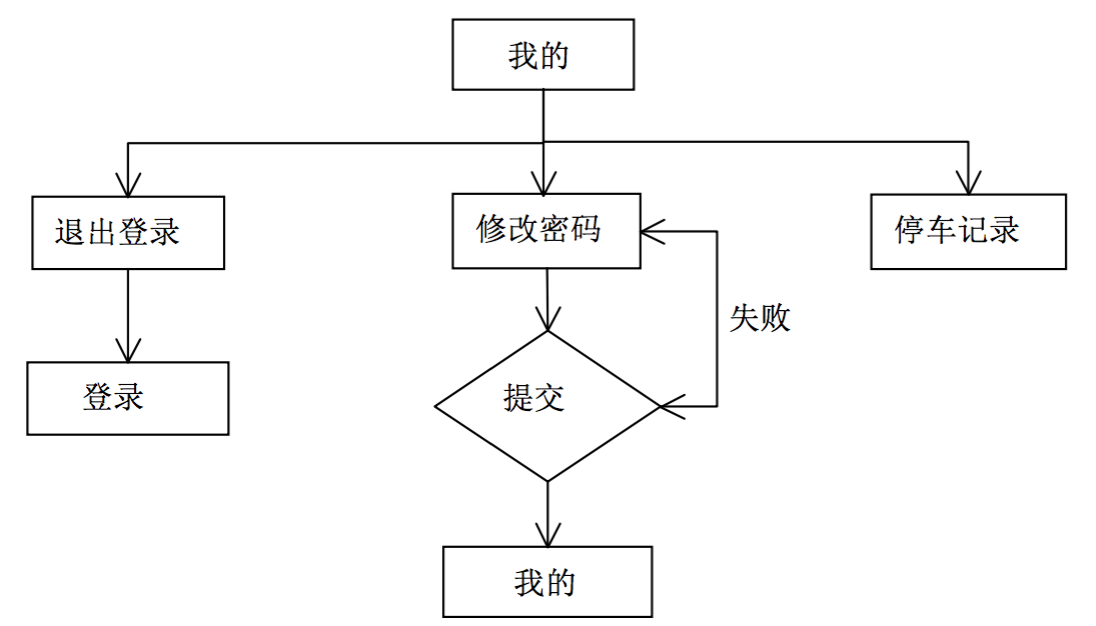


图3.4 个人中心中停车记录、修改密码、退出登录的流程图

### 3.3.2 界面展示

**尺寸资源 dimens.xml：**

界面的设计需要统一，所以一些常用的尺寸资源写在这里，使软件的布局统一，如左边距，右边距，文字大小等。

<resources>

<dimen name="activity\_horizontal\_margin">16dp</dimen>

<dimen name="activity\_vertical\_margin">16dp</dimen>

<dimen name="fab\_margin">16dp</dimen>

<dimen name="appbar\_padding\_top">8dp</dimen>

<dimen name="app\_bar\_height">180dp</dimen>

<dimen name="text\_margin">16dp</dimen>

<dimen name="text\_size\_default">16sp</dimen>

<dimen name="text\_size\_big">18sp</dimen>

<dimen name="text\_size\_small">14sp</dimen>

</resources>

**标题栏 layout\_actionbar.xml：**

ActionbBar是Android3.0(API level 11)中引入的新控件，但是由于开发中对于ActionBar的灵活性要求较高，原生的Actionbar无法满足要求，所以采用了自己实现一个ActionBar，同时对Application设置Theme.AppCompat.Light.NoActionBar属性，即去掉系统原生的ActionBar。

自定义ActionBar的背景为主题色黑色，高度为48dp，宽度占满全屏，其中有3个控件，分别是左侧的Button、中间的TextView和右侧的Button。左侧的Button主要用作于返回按钮，中间的title主要用于显示当前界面的标题，右侧的Button主要用于一些自定义事件，每个界面都有其自己的事件，比如添加车辆，删除车辆，预约车位界面中的下一步等等，有些界面没有此类事件，则将Button的visiable属性设置为不可见。

**登录及注册界面 activity\_login.xml：**

本软件将注册于登录放于同一个界面中，采用动画淡入淡出的效果切换。为了防止某些机型的屏幕过小，无法显示全部界面的情况，用ScrollView将整个界面包裹起来，让用户可以上下滑动。界面的EditText用Google在2015年发布的Design Support Library中的 TextInputLayout包裹，此控件的优点是可以动态的提示用户在EditText中输入文本的错误，而不用传统的Toast提示，达到更好的交互效果。软件中所有的确定、提交按钮采用CircularProgressButton，一个可以动态显示当前状态与执行结果的按钮。

**主activity activity\_main.xml：**

主界面采用滑动导航方式，界面中的导航栏采用Design Support Library中的 AppBarLayout包裹TabLayout，展示的页面采用viewpager。

**选择车辆以及列表布局 activity\_choose\_car.xml activity\_choose\_car\_list\_item.xml：**

所有的listView都采用了JazzyListView，一个可以在滑动时以特效的方式显示列表中的每个item。每个item的图片加载则采用facebook发布的图片加载框架SimpleDraweeView。Item中左侧垂直居中显示宽150dp高100dp的图片，右侧以纵向居中显示每个车辆的车辆类型和车牌号。

**选择车位 activity\_choose\_space.xml：**

界面采用ScrollView包裹以适配更多的机型。每个车位都由一个圆角正方形表示，通过自定义的layout根据屏幕尺寸来布局停车场。每个车位都可能有3种情况，分别为已停车、未停车和我的车位，每种情况显示不同的颜色。在下方，纵向显示了3个情况的颜色标注，方便用户识别区分。

**选择时间 activity\_choose\_time.xml：**

这个界面主要由2个部分构成，分别是安卓原生的DatePicker（日期选择器）和TimePicker（时间选择器）。用户可以滑动屏幕改变预约车位的时间。

**个人中心界面 fragment\_mine.xml：**

界面分为2个纵向展示的部分，第一部分的背景使用了自定义的view，动态的显示多个大小、透明度不同的光晕在做直线运动，用户的圆形头像居中展示，用户的个人信息（手机号，年龄，性别）在头像下方水平居中展示。第二部分纵向的显示了我的车辆、停车记录、修改密码、天气预报、退出登录等功能，点击进入相应的界面。

**个人信息的维护 activity\_update\_user\_info.xml：**

界面纵向展示了2个EditText，一个RadioGroup和一个CircularProgressButton，分别是编辑姓名、性别（男女），和提交修改按钮。

**修改密码 activity\_update\_user\_password.xml：**

界面纵向展示了2个EditText和一个CircularProgressButton，分别是输入原密码、新密码，和提交修改按钮。

**个人车辆信息以及列表 activity\_my\_car.xml activity\_my\_car\_list\_item.xml：**

此界面与选择车辆的界面展示基本相同，右上角多了个添加车辆的按钮。

**个人车辆信息的维护 activity\_update\_car.xml：**

界面顶部居中显示了一个SimpleDraweeView用来展示车辆图片。下面用2个水平居中的TextInputLayout以及EditText分别提示用户输入车辆类型和车牌号。最后用一个水平居中的CircularProgressButton来显示提交按钮。

**查询我的预约以及列表 fragment\_order.xml：**

**fragment\_order\_ist\_item.xml:**

fragment以纵向显示，顶部用了一个TextView靠右侧显示“我的预约”标题，紧接着是一条颜色为#979797的分割线，之后只一个占满父控件宽于高的JazzyListView，最后在右下角有一个Design Support Librar中的FloatingActionButton作为预约按钮。

每个item中垂直居中展示了3个TextView，每两个TextView中以一条颜色为#979797的竖线分割，这三个TextViewde 功能为别是展示预约时间，预约车位以及计费速率。

**停车记录以及列表 activity\_record.xml：**

**activity\_record\_list\_item.xml:**

activity中只有一个actionBar和JazzyListView。

每个item左侧居中展示了一个宽150dp高100dp的SimpleDraweeView用来展示车辆图片，右侧垂直居中纵向展示了4个LinearLayout，每个layout中都包含了2个TextView。4个LinearLayout分别表示停车时间，停车费用，车辆类型和车牌号。

## 3.4 项目框架搭建

### 3.4.1 项目结构图

**com.fyf.parkinglot(包名)**

**activity(activity在这个路径下)**

**addCar**

AddCarActivity.java(添加车辆的activity)

**chooseCar**

ChooseCarActivity.java(选择车辆的Activity)

ChooseCarListViewAdapter.java(选择车辆列表的适配器)

**chooseSpace**

ChooseSpaceActivity.java(选择车位的activity)

DrawParkinglot.java(布局停车场的layout)

GridViewAdapter,java(每个车位的适配器)

**chooseTime**

ChooseTimeActivity,java(选择预约时间的Activity)

**login**

LoginActivity,java(登录注册的Activity)

**main**

MainActivity.java(首页Activity)

MainPagerAdapter.java(首页Activity的ViewPager适配器)

**myCar**

MyCarActivity.java(我的车辆的Activity)

CarListViewAdapter.java(我的车辆Activity列表的适配器)

**record**

RecordActivity.java(停车记录的Activity)

RecordListViewAdapter.java(停车记录Activity列表的适配器)

**updateCar**

UpdateCarActivity.java(更新用户车辆信息的Activity)

**updateUserInfo**

UpdateUserInfoActivity.java(更新用户信息的Activity)

**updateUserPassword**

UpdateUserPassword.java(更新用户密码的Activity)

**application(application在这个路径下)**

ParkinglotApplication.java

**common(一下常量定义在这个路径下)**

ContextManager.java(存放Context)

GlobalDefine.java(定义常量)

ScreenInfo.java(手机屏幕信息)

SQLWord.java(服务器数据库的字段信息)

URLAddress.java(网络请求地址)

**fragment(fragment在这个路径下)**

**mine**

MineFragment.java(我的fragment)

CropperUserActivity.java(裁剪用户头像的Activity)

**order**

OrderFragment.java(预约车位的fragment)

OrderAdapter.java(已预约车位列表的适配器)

**model(数据模型在这个路径下)**

CarInfoBean.java(车辆信息模型)

ParkinglotInfoBean.java(车位信息模型)

ParkingRecordInfoBean.java(停车记录信息模型)

ResultInfoBean.java(返回的json固定格式模型)

UserInfoBean.java(用户信息模型)

UserInfoInCache.java(内存中缓存的用户信息)

**push(推送相关功能在这个路径下)**

MyPushMessageReceiver.java(处理推送)

Utils.java(推送的工具类)

**utils(工具类在这个路径下)**

DataOperateUtils.java(操作本地缓存的文件)

HttpUtils.java(封装网络请求)

JsonUtils.java(解析Json的工具类)

ListViewUtils.java(ListView的工具类)

NotifyDataChangeUtils.java(通知状态改变的工具类)

Utils.java(计算日期,解码,正则表达式验证等工具类)

**view(自定义view在这个路径下)**

CustomPrgressDailog.java(自定义加载动画)

CustomToast.java(自定义Toast)

HaloView.java(自定义光晕背景)

MyGridView.java(自定义GridView)

MyListView(完全展开的listView)

### 3.4.2 封装网络请求

网络请求采用OKHttp框架，在本地再次封装后使用。由于服务器返回的json都经过base64编码，所以本地接收到json后需要进行base64解码，再进行逻辑处理。

package com.fyf.parkinglot.utils;

import android.util.Log;

import com.squareup.okhttp.OkHttpClient;

import com.squareup.okhttp.Request;

import com.squareup.okhttp.RequestBody;

import com.squareup.okhttp.Response;

/\*\*

\* Created by fengyifei on 15/11/23.

\*/

public class HttpUtils {

private static OkHttpClient client = new OkHttpClient();

// post请求

public static String httpPost(String url, RequestBody body) {

try {

Request request = new Request.Builder().url(url).post(body).build();

Response response = client.newCall(request).execute();

if (response.isSuccessful()) {

String result = Utils.base64Decode(response.body().string());

Log.e("HttpResult",result);

return result;

} else {

return response.code() + "";

}

} catch (Exception e) {

return "";

}

}

// get请求

public static String httpGet(String url) {

try {

Request request = new Request.Builder().url(url).build();

Response response = client.newCall(request).execute();

if (response.isSuccessful()) {

String result = Utils.base64Decode(response.body().string());

Log.e("HttpResult",result);

return result;

} else {

return response.code() + "";

}

} catch (Exception e) {

return "";

}

}

// 返回的数据不用进行base64解码的get请求

public static String httpGetWithOutBase64(String url) {

try {

Request request = new Request.Builder().url(url).build();

Response response = client.newCall(request).execute();

if (response.isSuccessful()) {

String result = response.body().string();

Log.e("HttpResult",result);

return result;

} else {

return response.code() + "";

}

} catch (Exception e) {

return "";

}

}

}

### 3.4.3 定义常量

public class GlobalDefine {

public static String APP\_NAME = "parkinglot";

// 七牛

public static String QINIU\_AK = "XPHnXDfTqIHcGg5M1nqnaLXtmupPqSEJfKkB3Czz";

public static String QINIU\_SK = "zFC8HWzqUXVxJIT5kQ5aKO\_aZVipCJa1U3LSUGUU";

// 七牛中默认车辆图片名称

public static String QINIU\_CAR\_DEFAULT = "car\_default.jpg";

// IM即时通信

public static String IM\_APPKEY= "d0044ef8d4baa85de9e393dd";

public static String IM\_USER\_PASSWORD = "parkinglot";

//百度车联网API

public static String CAR\_API\_AK = "fTvBOmDjpfu5bulB06ByOToR";

}

public class URLAddress {

// 服务器地址

public static final String publicURL = "";

public static final String localURL = "http://115.28.111.133:8080/Parkinglot";

public static final String URL = localURL;

public static final String QINIU\_URL = "http://www.7xoqdw.com1.z0.glb.clouddn.com";

public static final String loginURL = URL + "/userlogincl";// 登录

public static final String registURL = URL + "/userregistcl"; // 注册

public static final String allSpaceInfoURl = URL + "/findallparkingspacecl"; // 查询所有车位信息

public static final String findUserCarURL = URL + "/findcarinfocl"; // 查询用户车辆

public static final String orderSPaceURl = URL + "/orderparkingspacecl"; // 预约车位

public static final String findTodayOrderURl = URL + "/findtodayorderbyuseridcl";// 查询用户今日预约

public static final String updateUserInfoURL = URL + "/updateuserinfocl";// 更新用户信息

public static final String updateUserPasswordURL = URL + "/updateuserpasswordcl";// 查询用户密码

public static final String findRecordByUserIdURL = URL + "/findparkingrecordbyuseridcl";//根据用户id查询停车记录

public static final String cancleOrderURL = URL + "/cancleorderparkingspacecl"; // 取消预约

public static final String endParkingURL = URL + "/endorderparkingspacecl";// 结束停车

public static final String addCarURL = URL + "/addcarinfocl";// 添加车辆

public static final String updateCarURL = URL + "updatecarinfocl";// 更新车辆

public static final String deleteCarURL = URL + "/deletecarinfocl";// 删除车辆

public static final String findImaccountURl = URL + "/findimaccountcl";// 查找即时通信账户

public static final String weatherForcastURL = "http://api.map.baidu.com/telematics/v3/weather";// 天气预报

}

# 4 程序实现及结果

## 4.1 登录与注册

登录与注册写在一个界面，采用动画效果来切换注册与注册功能。

淡入动画效果:时长500毫秒，透明度从0变为1，并且在动画后保持动画最后的效果

<set xmlns:android="http://schemas.android.com/apk/res/android">

<alpha

android:duration="500"

android:fillAfter="true"

android:fromAlpha="0.0"

android:toAlpha="1.0" />

</set>

淡出动画效果: 时长500毫秒，透明度从1变为0，并且在动画后保持动画最后的效果

<set xmlns:android="http://schemas.android.com/apk/res/android">

<alpha

android:duration="500"

android:fillAfter="true"

android:fromAlpha="1.0"

android:toAlpha="0.0" />

</set>

对于手机号的验证采用正则表达式

public static boolean isMobileNum(String mobiles) {

Pattern p = Pattern.compile("^((13[0-9])|(15[^4,\\D])|(18[0-9])|177)\\d{8}$");

Matcher m = p.matcher(mobiles);

return m.matches();

}

当点击切换登录注册按钮时，会将当前所有按钮设置为不可点击，执行淡出动画，当变淡动画执行完之后，如果当前状态为登录，则将标志flag设置为注册状态，按钮的文字变为注册。如果当前状态为注册，则将标志flag设置为登录状态，按钮的文字变为登录。然后执行淡入动画，当淡入动画执行完后将所有按钮设置为可点击。

登录和注册采用异步处理

登陆的AsyncTask：在线程执行前，将确定按钮设置为运行状态，所有的按钮设置为不可点击状态，获取手机号和密码输入框中的内容。然后开始在线程中向后台接口发送post请求，请求中带有用户名和密码，接收到服务器返回的结果后，先将按钮设置为可点击状态，若登录成功，则将手机号密码通过SharedPreference储存到文件并跳转到主界面，若登录失败，则显示服务器返回的失败原因。

注册的AsyncTask：在线程执行前，将确定按钮设置为运行状态，所有的按钮设置为不可点击状态，获取手机号、密码输入框中的内容。然后开始在线程中向后台接口发送post请求，请求中带手机号、密码、姓名（默认为空）、年龄（默认为0）、性别（默认为女），接收到服务器返回的结果后，先将按钮设置为可点击状态，若注册成功，则将注册信息储存并跳转到主界面，若注册失败，则显示服务器返回的失败原因。

## 首页

首页采用ViewPager的方式，ViewPager初始化三个界面，分别为预约状态，车友，以及我的界面。

在MainActivity中需要进行一些数据的初始化，比如

查询个人车辆信息:向查询车辆的接口发送post请求，请求中带有用户id，将返回的json解析后存入内存。

获取屏幕信息：通过getWindow().getWindowManager().getDefaultDisplay()获取屏幕的宽高及屏幕的像素密度比，通过getWindow().getDecorView().getWindowVisibleDisplayFrame获取状态栏高度，并存入内存中。

### 4.2.2 查询预约页

进入该页面，首先会自动查询用户的预约车位的信息:首先显示加载动画，然后在线程中通过post方法向查询预约车位的接口发送带有用户id的请求，解析返回的结果，若有数据则显示，若无数据则不做任何处理

同时，在这个页面也可以点击右下角的悬浮按钮进行预约车位:右下角的悬浮按钮点击后弹出一个Snackbar，点击确定后跳转到预约车位的界面，点击否则不做任何处理。

### 4.2.3 个人中心页

个人中心主要功能是用户头像的处理，拍照功能如下:首先获取sd卡的绝对路径，如果sd可用，则通过在sd卡的根目录下创建一个名为“parkinglot”的文件夹，然后根据时间生成一个后缀为jpg的文件，将这个图片的路径通过intent传递，最后通过startActivityForResult向系统相机发送intent。

从相册中获取图片:直接通过startActivityForResult调用系统相册功能。

得到图片后需要调用图片裁剪的功能, android4.4会出现从相册选取图片查询路径为空的情况,所以需要将路径转换下:

private String uri2filePath(Uri uri, Activity activity) {

String path = "";

if (Build.VERSION.SDK\_INT >= 19 && DocumentsContract.isDocumentUri(activity, uri)) {

String wholeID = DocumentsContract.getDocumentId(uri);

String id = wholeID.split(":")[1];

String[] column = {MediaStore.Images.Media.DATA};

String sel = MediaStore.Images.Media.\_ID + "=?";

Cursor cursor = activity.getContentResolver().query(

MediaStore.Images.Media.EXTERNAL\_CONTENT\_URI, column, sel,

new String[]{id}, null);

int columnIndex = cursor.getColumnIndex(column[0]);

if (cursor.moveToFirst()) {

path = cursor.getString(columnIndex);

}

cursor.close();

} else {

String[] projection = {MediaStore.Images.Media.DATA};

Cursor cursor = activity.getContentResolver().query(uri, projection, null, null, null);

int column\_index = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);

cursor.moveToFirst();

path = cursor.getString(column\_index);

}

return path;

}

图片裁剪完成后将新的用户头像存储到本地并且更新界面。

个人中心的背景是动态的光晕:光晕的运动是通过TimerTask执行定时任务，每85毫秒向Handler发动一次Message，每一次执行光晕都会运动几像素。

点击修改用户信息按钮后进入到完善个人信息的activity。在个人信息展示之下有5个item，点击后分别进入到查询个人车辆、停车记录，修改密码、天气预报、退出登录的界面。

### 个人信息完善

在此界面可已更新用户的个人信息，点击确定按钮后，开始执行AsyncTask：在线程执行前，将确定按钮设置为运行状态，所有的按钮设置为不可点击状态，获取手机号、密码、姓名、年龄、性别输入框中的内容并判断输入的合法性。然后开始在线程中向后台接口发送post请求，请求中带手机号、密码、姓名、年龄、性别，接收到服务器返回的结果后，先将按钮设置为可点击状态，若更新成功，则将信息储存，解析服务器返回的json更新界面并跳转到主界面，若更新失败，则显示服务器返回的失败原因。

## 个人车辆信息的维护

### 查询

在MainActivity中已经查询过一次用户车辆，如果车辆信息有更新的话，只需将新的车辆信息储存到内存缓存中再调用adapter.notifyDataSetChanged()更新界面即可。列表都采用JazzyListView,可以添加动画效果。

### 增加

从本地选取车辆照片或者拍照获取图片的逻辑与用户头像的逻辑相同。完成基本信息的填写之后，首先检查表单的完整性与合法性，若没有问题，则进行下一步

添加车辆信息的逻辑处理如下:显示加载Dialog->向服务器发送车辆图片->若不成功Toast提示失败，若成功则获取图片存储地址并进行下一步->向服务器发送车辆的基本信息（首先读取输入框中的车辆类型，车牌号，并通过post方法向接口发送用户id，base64编码后的车辆类型和车牌号，以及车辆图片地址）->接收处理结果->关闭加载Dialog->显示结果若成功，重新从服务器获取该用户的车辆信息，更新内存中的缓存数据，若失败则显示服务器返回的失败原因。

### 修改

修改车辆信息与添加车辆信息逻辑相同只是与更新车辆信息调用的接口不同，先上传图片再上传车辆信息，根据服务器返回的结果，判断是将新的车辆信息更新到内存缓存中还是通过Toast提示失败原因。

### 删除

删除车辆的AsyncTask：首先显示正在加载的dialog，然后在线程中向服务器的删除车辆接口发送要删除的车辆的id。删除车辆成功后，需要在内存缓存中移除该车辆信息，若未成功，则通过Toast提示删除失败，最后取消dialog的显示。

## 停车记录

查询停车记录的AsyncTask：首先显示正在加载的dialog，然后在线程中向服务器的查询停车记录接口发送要要查询的用户id。查询成功后，在ListView中显示停车记录，若未成功，则通过Toast提示查询失败的原因，最后取消dialog的显示。

## 修改密码

修改密码时首先需要校验表单（原密码与新密码输入框中的内容必须都不能为空并且大于6位），之后下个服务器修改密码的接口通过post请求发送用户输入的原密码与新密码，根据服务器返回的结果提示用户修改成功或者修改失败的原因。若修改成功则更新本地缓存文件中的用户密码，并关闭当前activity。

## 退出登录

退出登录时首先弹出一个AlertDialog，提示是否确定退出登录，若点否，则AlertDialog消失，若点事，则执行清除本地缓存文件（包括用户名，密码、头像等）、注销推送的tag、返回到登陆界面。

## 预约车位

### 选择停车车位

在选择车位时，用到了自定义View，根据屏幕的长宽计算每个车位的应该显示在屏幕上的大小。这里定义每一行显示7个车位，每个车位之间的间隔是10dp。private final int column = 7;//定义车位显示列数

private final int padding = 10;//车位间距

首先计算这个手机屏幕下，每个车位的宽度，width = (gv\_gridView.getWidth() - padding \* (column - 1)) / column;

然后根据每个车位是否被占用的状态，设置不同的颜色，并且给每个车位设置点击事件，改变颜色。最后将所有的车位放入自定义的布局中。

在Activity中需要查询所有车位的状态：首先显示正在加载的dialog，通过get请求获取停车场数据，解析数据后根据数据生成停车场布局，最后dialog消失。

用户选择好停车位后可以进行下一步，通过intent向选择下一个activity传递用户选择的车位id，若用户没有选择车位，则提示用户未选择车位。

### 选择停放车辆

与首页查询车辆的逻辑相同，查询到用户车辆后在listView中显示出来。用户点击某一辆车之后，触发ListView的监听：将用户选择的车位id与选择的车辆id通过intent传入下一个activity。

### 选择停车时间

选择停车时间界面采用DatePicker与TimePicker结合，将时间精确到秒，在用户选择完停车时间时候本地会对停车时间做一次校验，满足停车时间必须在当前时间之后以及智能预约2小时以内的车位，避免资源浪费。

校验表单：通过写好的工具类比较停车时间是否在未来的2小时之内，若不在，则提示时间错误，若在则进行下一步。

表单校验无误后，便向服务器提交预约数据：首先显示正在加载的dialog，然后在线程中向服务器的接口通过post请求发送要要查询的停车的车位id、车辆id、与预约停车的时间。

### 预约结果

预约成功后会向用户提示预约结果，并且可以直接返回到主界面书界面采用了singleTask的启动模式，所以可以直接跳转。主界面会在onResume()中刷新用户预约情况的数据。若失败，则提示失败原因。

## 取消预约

在主界面中，停车记录是以列表形式现实的，如果用户有预约记录但是还未开始停车，则可以通过长按这条记录来确定是否取消预约:若点否则AlertDialog消失，若点是，则执行以下流程：

取消预约的AsyncTask：弹出加载dialog并在在线程中向取消预约的接口发送该车位，接收到服务器返回的结果并解析后，若成功则更新界面，若失败则提示原因，最后加载dialog消失。

## 结束停车

如果用户已经开始停车,则可以通过长按这条记录来确定是否结束停车，若点否，则AlertDialog消失，若点是则弹出加载dialog并在在线程中向结束停车的接口发送该车位，接收到结果并解析后，若成功则更新界面，若失败则提示原因，最后加载dialog消失。

## 4.11 推送

### 4.11.1 在推送平台创建应用

推送采用百度云推送的SDK，首先登陆http://push.baidu.com/ 注册账号，完成开发者身份认证，点击创建应用，完成基本信息的填写。

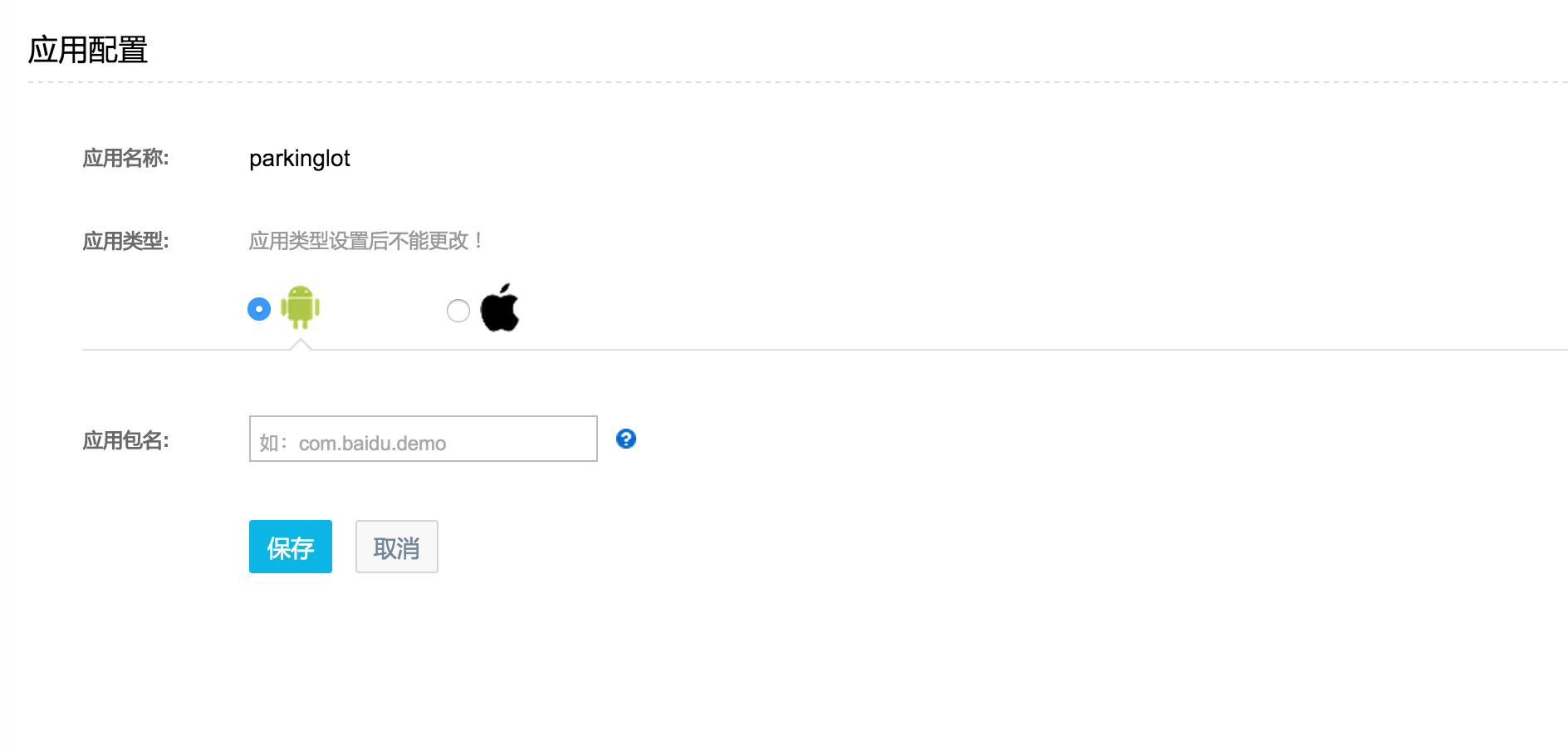


图4.1 配置云推送

配置完成后即可获取到应用的APP ID与API KEY，在集成推送SDK是需要用到。

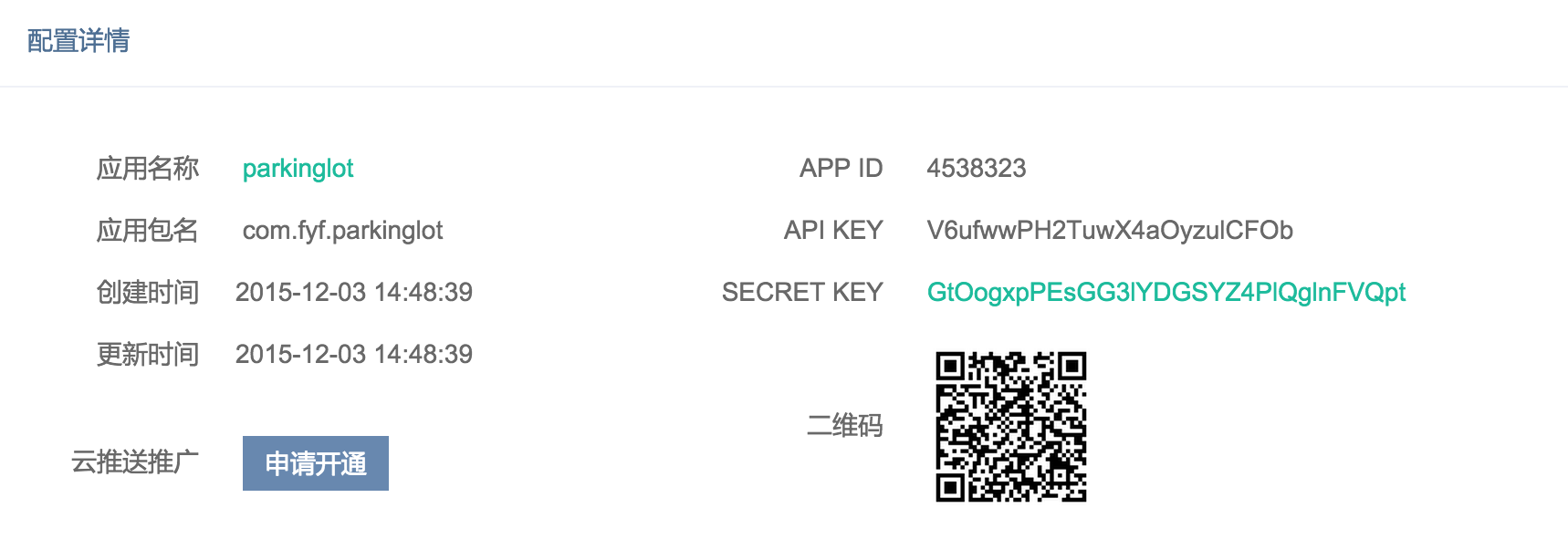


图4.2 获取Key

### 4.11.2 集成SDK

在<http://push.baidu.com/sdk/push_client_sdk_for_android> 下载安卓版的推送SDK。



图4.3 下载推送sdk

将so文件拷贝至jniLibs下，将jar包拷贝至libs下

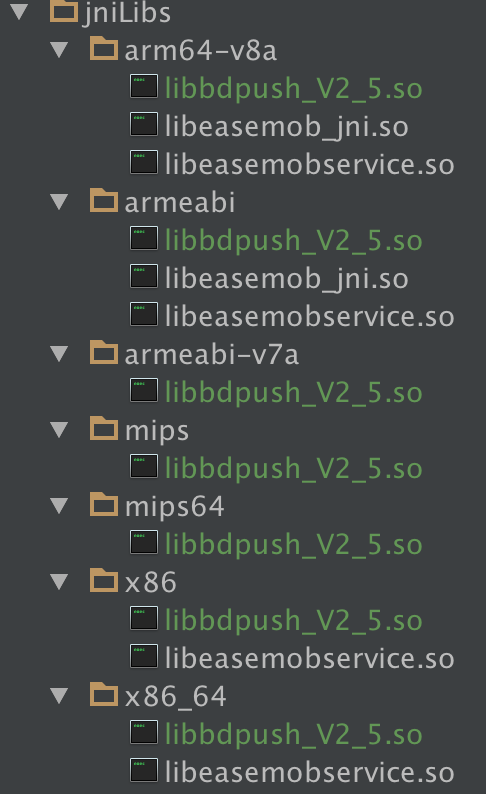


图4.4 在项目中导入推送sdk

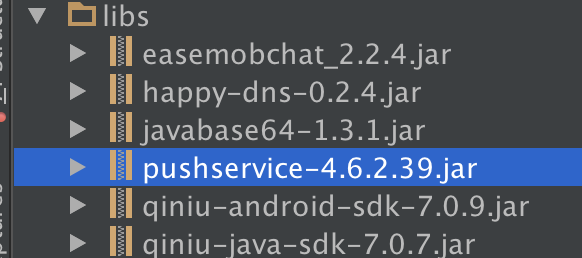


图4.5 在项目中导入推送sdk

在AndroidManifest.xml中声明权限，注册服务及广播。

<!-- Push service 运行需要的权限 -->

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.RECEIVE\_BOOT\_COMPLETED" />

<uses-permission android:name="android.permission.WRITE\_SETTINGS" />

<uses-permission android:name="android.permission.VIBRATE" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.DISABLE\_KEYGUARD" />

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_DOWNLOAD\_MANAGER"/>

<uses-permission android:name="android.permission.DOWNLOAD\_WITHOUT\_NOTIFICATION" />

<uses-permission android:name="android.permission.EXPAND\_STATUS\_BAR" />

<!-- push service start -->

<!-- 用于接收系统消息以保证PushService正常运行 -->

<receiver android:name="com.baidu.android.pushservice.PushServiceReceiver"

android:process=":bdservice\_v1" >

<intent-filter>

<action android:name="android.intent.action.BOOT\_COMPLETED" />

<action android:name="android.net.conn.CONNECTIVITY\_CHANGE" />

<action android:name="com.baidu.android.pushservice.action.notification.SHOW" />

<action android:name="com.baidu.android.pushservice.action.media.CLICK" />

<!-- 以下四项为可选的action声明，可大大提高service存活率和消息到达速度 -->

<action android:name="android.intent.action.MEDIA\_MOUNTED" />

<action android:name="android.intent.action.USER\_PRESENT" />

<action android:name="android.intent.action.ACTION\_POWER\_CONNECTED" />

<action android:name="android.intent.action.ACTION\_POWER\_DISCONNECTED" />

</intent-filter>

</receiver>

<!-- Push服务接收客户端发送的各种请求-->

<receiver android:name="com.baidu.android.pushservice.RegistrationReceiver"

android:process=":bdservice\_v1" >

<intent-filter>

<action android:name="com.baidu.android.pushservice.action.METHOD" />

<action android:name="com.baidu.android.pushservice.action.BIND\_SYNC" />

</intent-filter>

<intent-filter>

<action android:name="android.intent.action.PACKAGE\_REMOVED" />

<data android:scheme="package" />

</intent-filter>

</receiver>

<service android:name="com.baidu.android.pushservice.PushService" android:exported="true"

android:process=":bdservice\_v1" >

<intent-filter >

<action android:name="com.baidu.android.pushservice.action.PUSH\_SERVICE" />

</intent-filter>

</service>

<!-- 4.4版本新增的CommandService声明，提升小米和魅族手机上的实际推送到达率 -->

<service android:name="com.baidu.android.pushservice.CommandService"

android:exported="true" />

<!-- push结束 -->

<!-- 4.6版本新增的Activity声明，提升Push后台存活率 -->

<activity

android:launchMode="singleInstance"

android:name="com.baidu.android.pushservice.PushKeepAlive"

android:theme="@android:style/Theme.Translucent.NoTitleBar"/>

最后在Application中添加执行初始化的代码

private void initPush() {

PushManager.startWork(getApplicationContext(),

PushConstants.LOGIN\_TYPE\_API\_KEY,

Utils.getMetaValue(getApplicationContext(), "BAIDU\_PUSH\_API\_KEY"));

// Push: 如果想基于地理位置推送，可以打开支持地理位置的推送的开关

PushManager.enableLbs(getApplicationContext());

}

### 4.11.3 处理推送消息

在用户成功登陆客户端后需要将该用户的用户id注册到百度云推送，通过PushManager.setTags（）来绑定，以确保该用户能单独收到的推送消息。在用户退出登录时，需要将该用户的id在百度云推送解绑，通过PushManager.delTags()解绑，以保证该用户不会再收到的推送消息。

### 4.11.4 推送结果

在云推送页面可以创建推送消息，配置完成后点击确定发送即可。

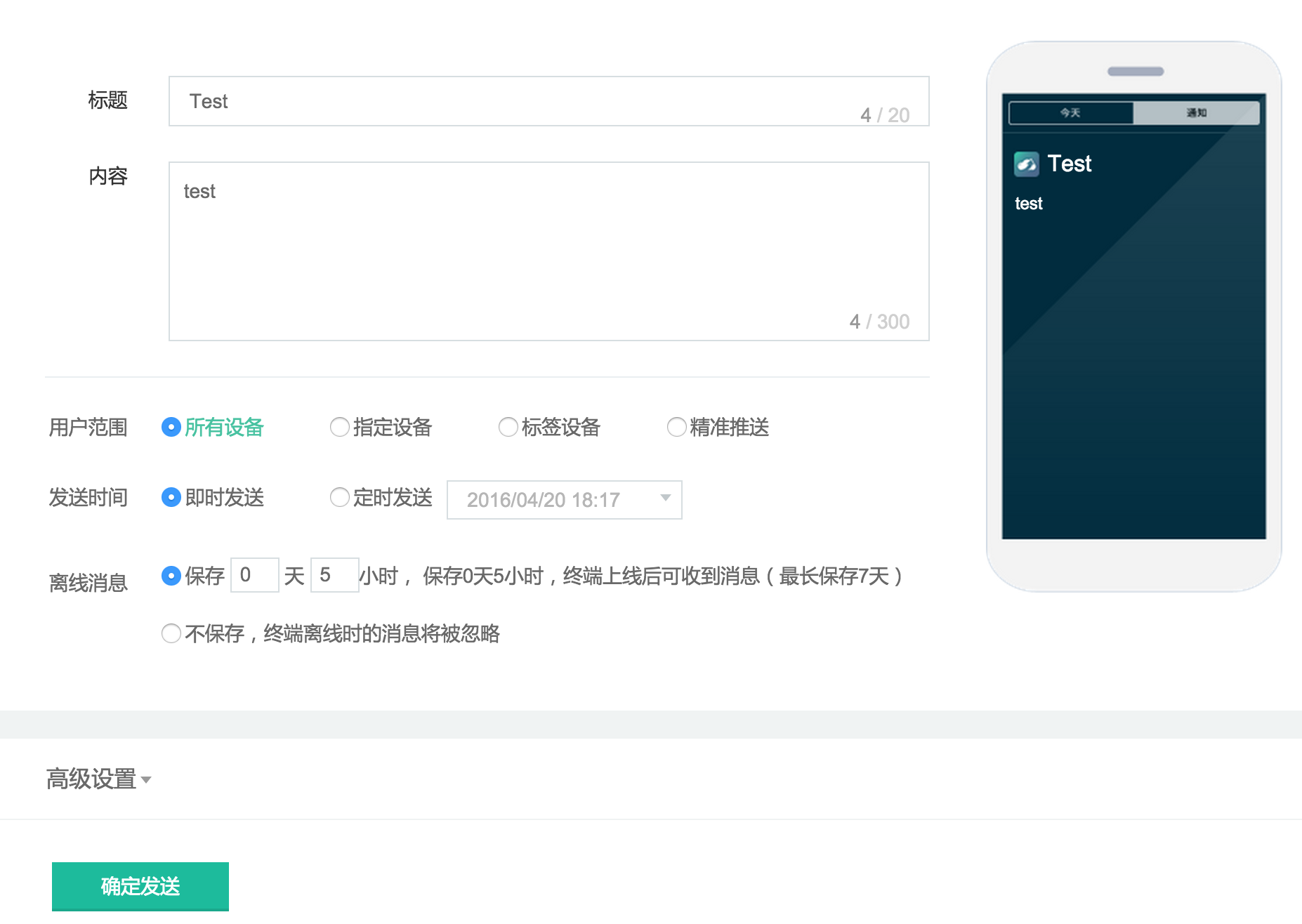


图4.6 通过网页进行推送

在客户端可以看到推送结果。

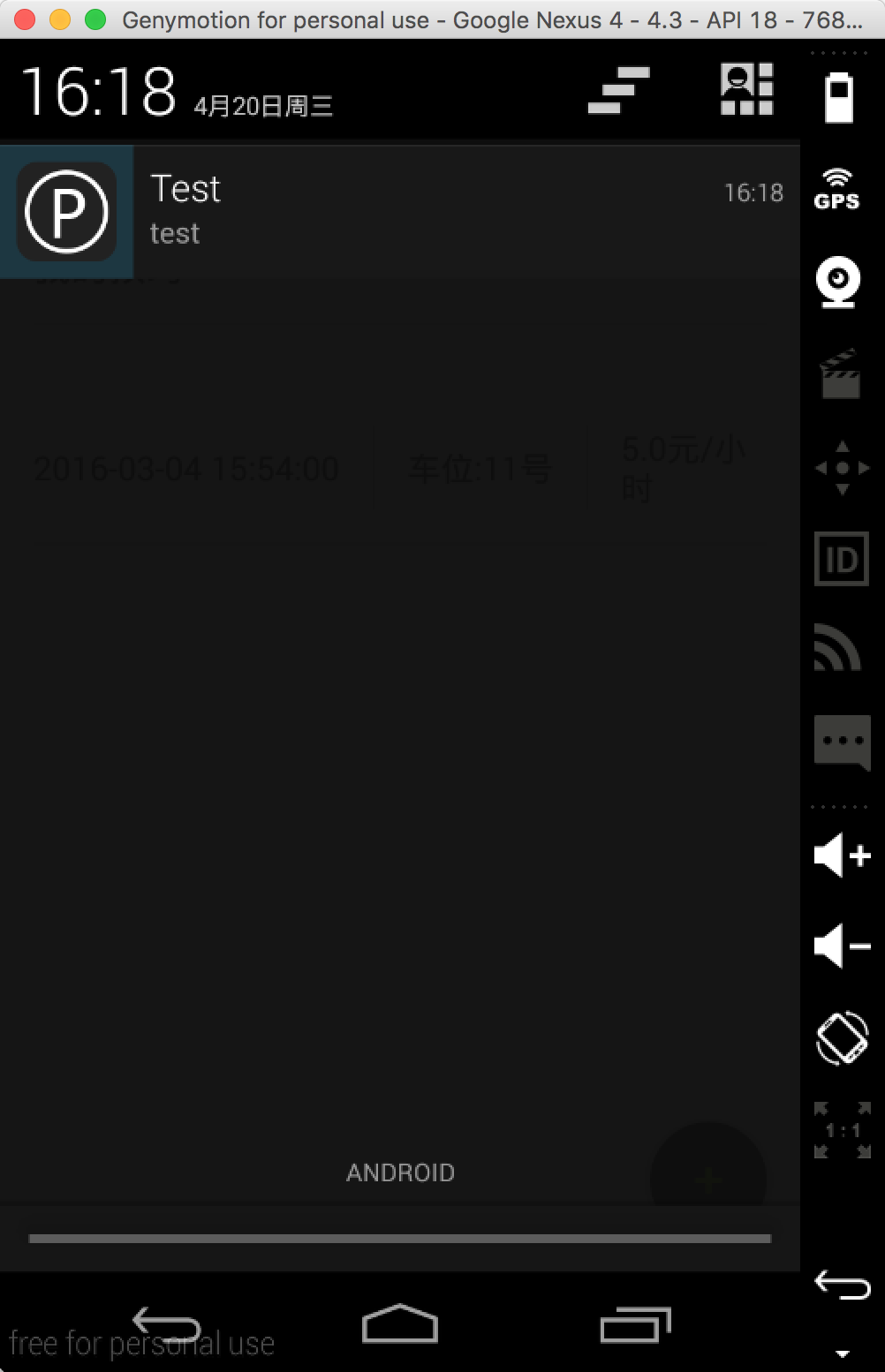


图4.7 推送结果

## 4.12 build.gradle

项目需要在在build.gradle中配置运行项目所需最低的安卓版本，以及版本号、版本名称，同时因为引入了jar包，所以需要在gradle中配置。

apply plugin: 'com.android.application'

android {

compileSdkVersion 23

buildToolsVersion "23.0.3"

defaultConfig {

applicationId "com.fyf.parkinglot"

minSdkVersion 15

targetSdkVersion 23

versionCode 1

versionName "1.0"

}

buildTypes {

release {

minifyEnabled false

proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'

}

}

packagingOptions {

}

}

dependencies {

compile fileTree(include: ['\*.jar'], dir: 'libs')

testCompile 'junit:junit:4.12'

compile 'com.android.support:appcompat-v7:23.3.0'

compile 'com.android.support:design:23.3.0'

compile 'com.android.support:cardview-v7:23.3.0'

compile 'com.github.dmytrodanylyk.circular-progress-button:library:1.1.3'

compile 'com.google.code.gson:gson:2.4'

compile 'com.sefford:circular-progress-drawable:1.31@aar'

compile 'com.twotoasters.jazzylistview:library:1.2.1'

compile 'com.twotoasters.jazzylistview:library-recyclerview:1.2.1'

compile 'de.hdodenhof:circleimageview:2.0.0'

compile 'com.edmodo:cropper:1.0.1'

compile 'com.squareup.okhttp:okhttp:2.6.0'

compile 'com.loopj.android:android-async-http:1.4.9'

compile 'com.facebook.fresco:fresco:0.9.0+'

compile files('libs/happy-dns-0.2.4.jar')

compile files('libs/qiniu-android-sdk-7.0.9.jar')

compile files('libs/javabase64-1.3.1.jar')

compile files('libs/pushservice-4.6.2.39.jar')

compile files('libs/easemobchat\_2.2.4.jar')

compile 'com.jakewharton:butterknife:5.1.1'

}

## 4.13 结果展示

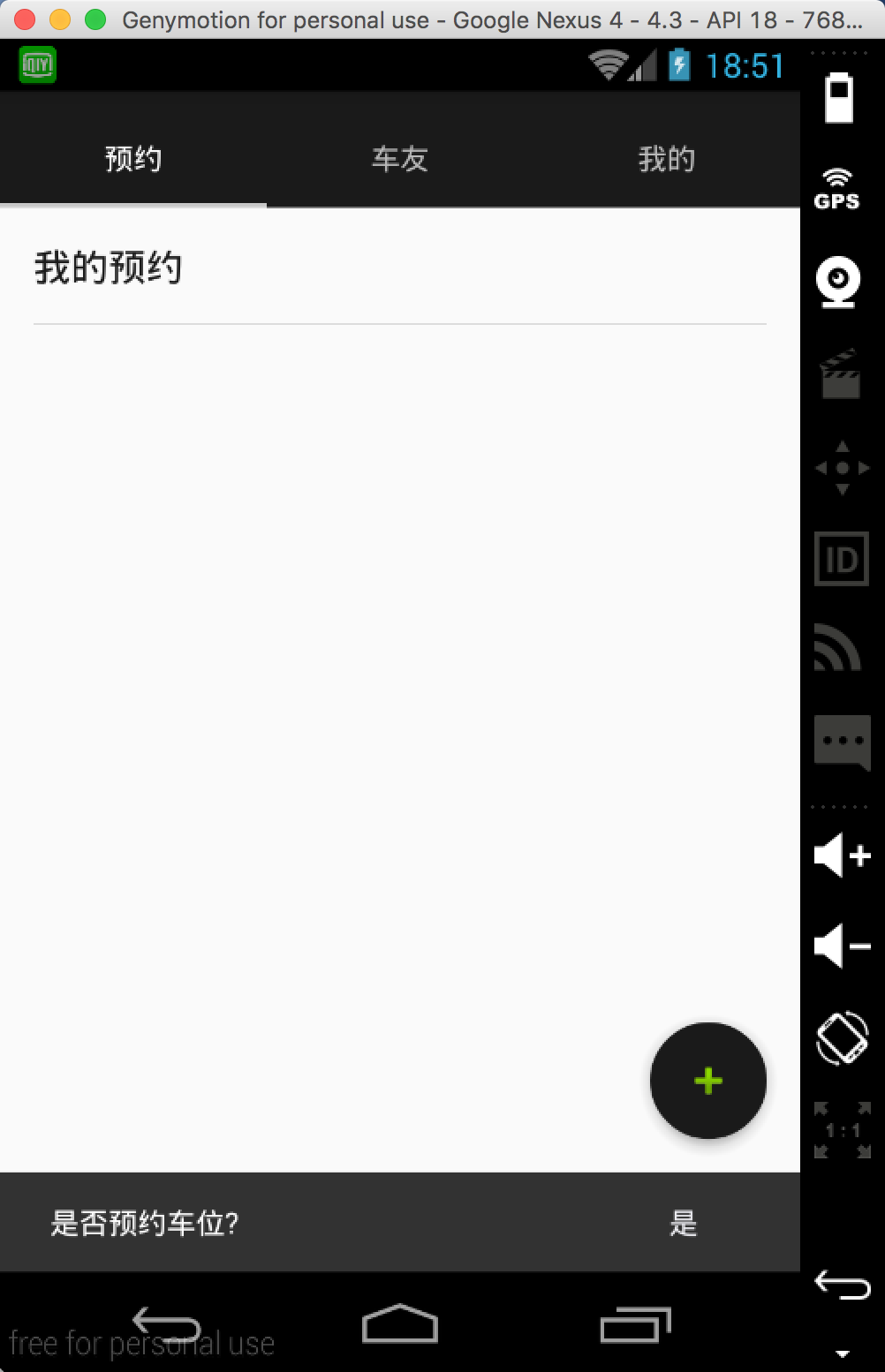
 

图4.8 登录 图4.9 首页与预约车位

图4.10 选择车位 图4.11 选择车辆 

图4.11 选择预约时间 图4.12 预约成功

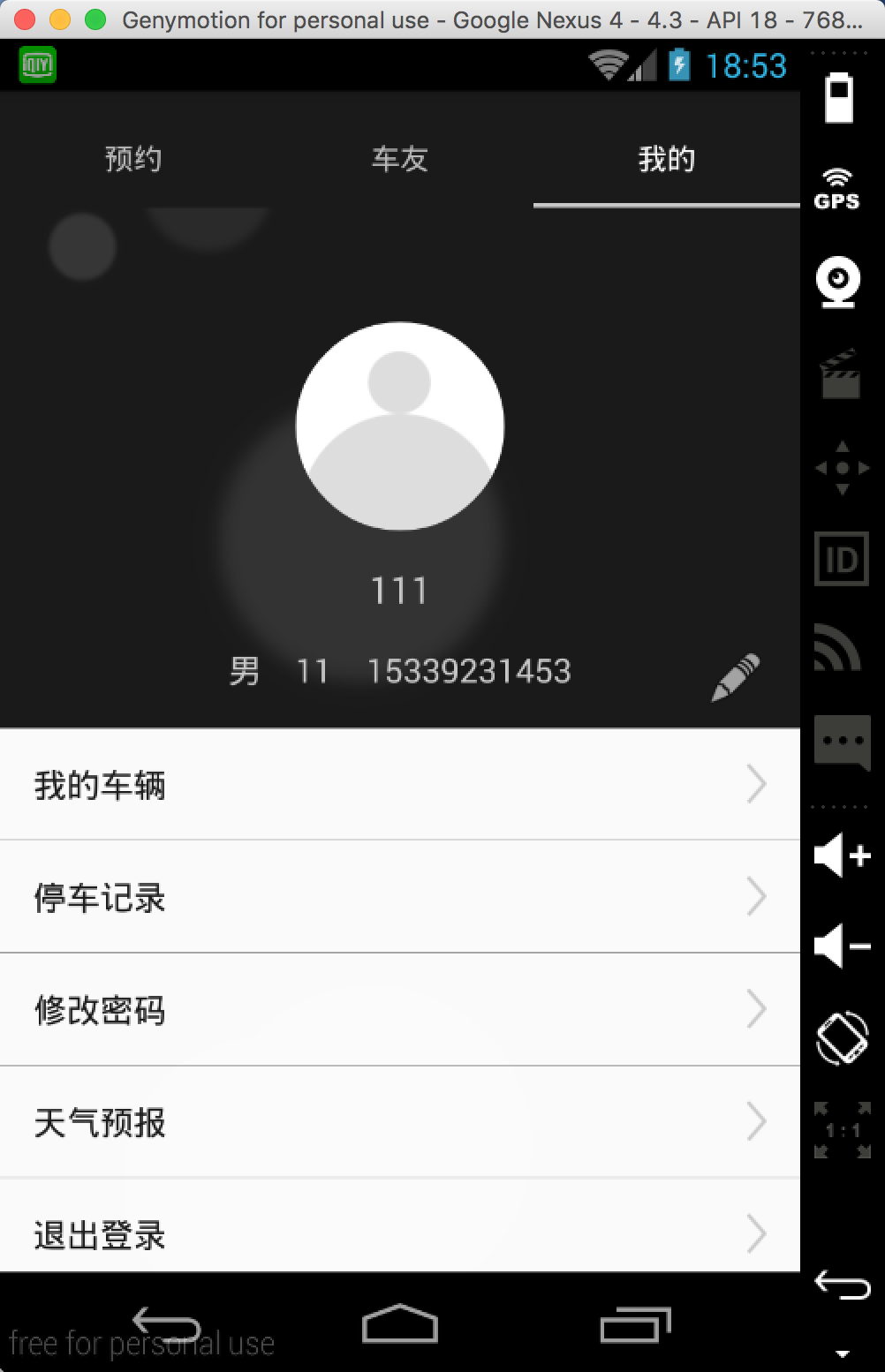
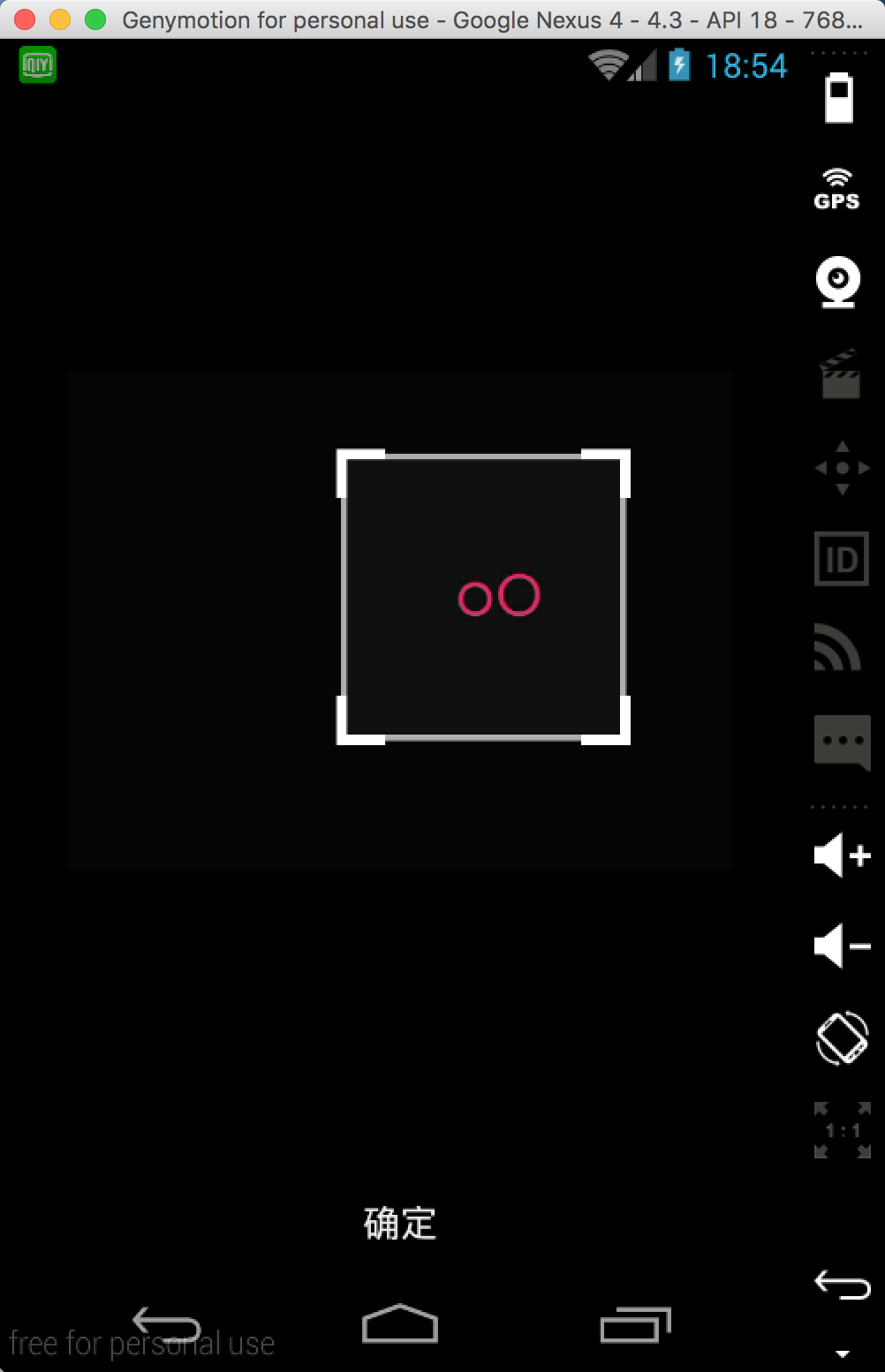
图4.13预约结果 图4.14 取消预约 

图4.15个人中心界面 图4.16 裁剪图片

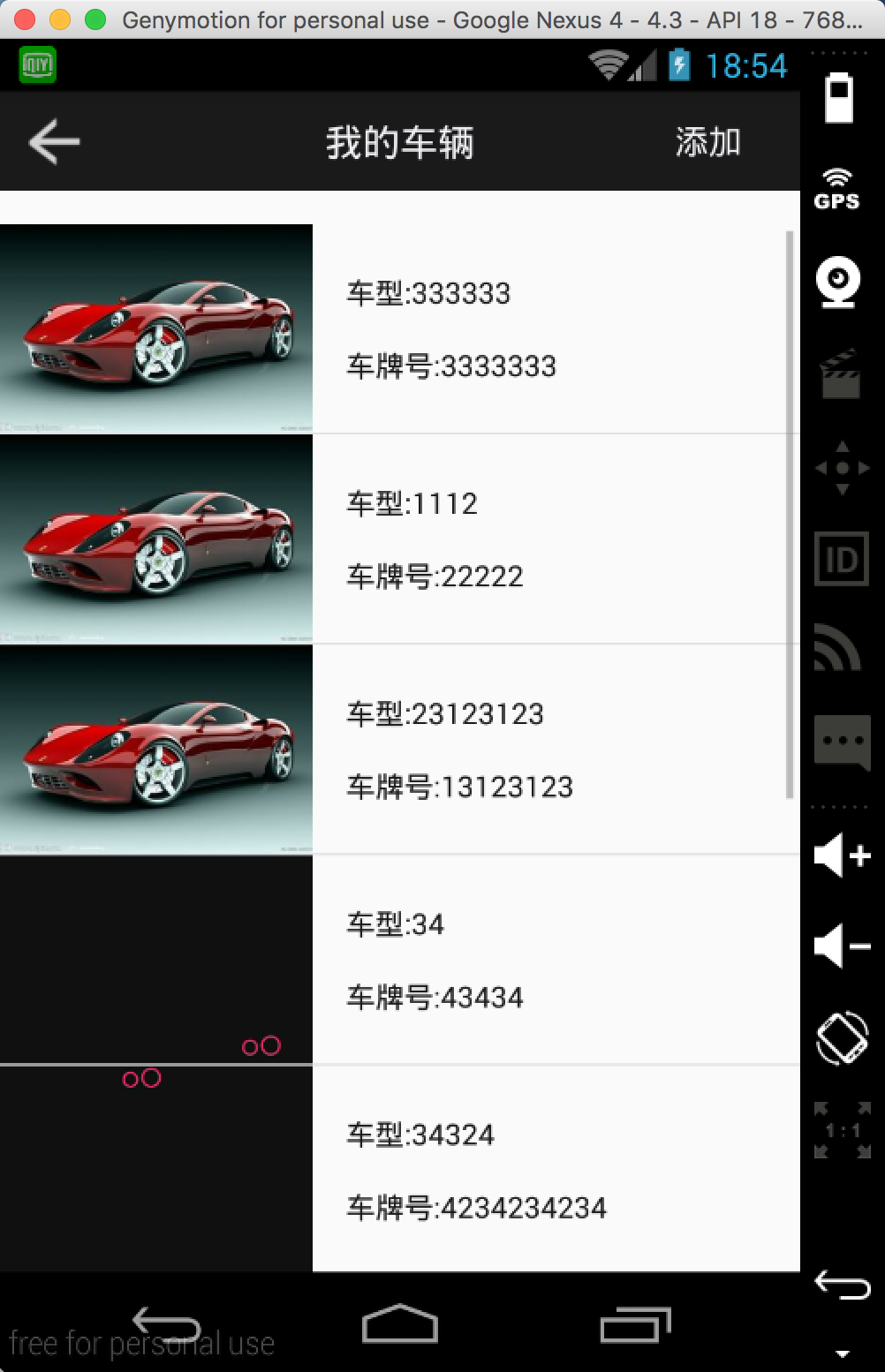
 

图4.17修改个人信息 图4.18 我的车辆列表 

图4.19添加车辆 图4.20 修改车辆

图4.21停车记录 图4.22 修改密码

# 总结与展望

通过此次的停车场管理手机客户端软件开发，我对java与android有了更深的认识，对于Android Studio这个开发平台的掌握更深。

在整个毕设过程中，我首先仔细阅读了毕设的要求与题目，构建出了软件大致的开发结构，之后与同学一起探讨软件的功能流程、界面的设计与配色，做出了原型图，最后在Android Studio平台上编写代码，编译运行，调试。

在这次毕业设计中，遇到了几个比较麻烦的问题，第一个是上传图片使用的是七牛云，再上传图片时需要传递appkey与secertkey，并且设置相关参数以及随机文件名，花费了很长时间阅读文档，调试。第二个是预约车位的时间判断，需要先获取当前时间，并且判断预约时间是否在当前时间后的2小时之内，将字符串转化为Calendar，再通过Calendar获取长整形的unix时间戳，最后进行时间比较，同时还要考虑到字符串格式有问题的情况，在这里也花费了较长时间调试。

本项目目前的开始停车与结束停车都是需要用户手动控制，未来可以与实际的停车场结合使用，配合车牌号识别的功能，实现停车，结束停车完全自动化，极大地节约人工成本。

# 致谢

非常感谢杨光老师在我大学的最后学习阶段——毕业设计阶段给自己的指导，从最初的定题，到资料收集，到写作、修改，到论文定稿，她们给了我耐心的指导和无私的帮助，在此我向她表示我诚挚的谢意。同时，感谢所有任课老师和所有同学在这四年来给自己的指导和帮助，是他们教会了我专业知识，教会了我如何学习，教会了我如何做人。正是由于他们，我才能在各方面取得显著的进步，在此向他们表示我由衷的谢意，并祝所有的老师培养出越来越多的优秀人才，桃李满天下！

通过这一阶段的努力，我的毕业论文《停车场管理手机客户端软件—登陆与车位预约模块设计》终于完成了，这意味着大学生活即将结束。在大学阶段，我在学习上和思想上都受益非浅，这除了自身的努力外，与各位老师、同学和朋友的关心、支持和鼓励是分不开的。

在本论文的写作过程中，我的导师杨光老师倾注了大量的心血，从选题到开题报告，从写作提纲，到一遍又一遍地指出每稿中的具体问题，严格把关，循循善诱，在此我表示衷心感谢。同时我还要感谢在我学习期间给我极大关心和支持的各位老师以及关心我的同学和朋友。

写作毕业论文是一次再系统学习的过程，毕业论文的完成，同样也意味着新的学习生活的开始。

感谢各位专家的批评指导。

# 参考文献

1. 李刚. 疯狂 Android 讲义[M]. 北京：电子工业出版社，2013:25-42.
2. 杨云君. Android 的设计与实现[M]. 北京：机械工业出版社，2013:45-49.
3. 百度.百度开发者心.http://developer.baidu.com
4. 杜吉志，徐明昆. Android 系统内存管理研究及优化[J]. 软件，2012,24(5)：69-80
5. 任玉刚. Android开发艺术探索 电子工业出版社 2015.09
6. 伽玛，李英军. 设计模式 可复用面向对象软件的基础. 机械工业出版社 2013.
7. 柯元旦. Android 内核剖析[M]. 北京：电子工业出版社，2011:59-70.
8. 高焕堂. Google Android 应用框架原理与程序设计 36 计[M]. Misoo,2010:8-13.