**学 号： 3162052051433#**

**桂林理工大学**

**GUILIN UNIVERSITY OF TECHNOLOGY**

**生产实习报告**

**学 院： 信息科学与工程学院 #**

**专 业： 软件工程 #**

**班 级： 2016-1班 ##**

**学 生： 徐仕成 #**

**指导教师： 陈基漓 ##**

**评 分： #**

**评 语：**

**2019年 10 月 27 日**

目录

[1.系统分析说明 2](#_Toc23547200)

[1.1项目可行性分析 2](#_Toc23547201)

[1.2用户需求分析 2](#_Toc23547202)

[1.2.1功能需求 2](#_Toc23547203)

[2.系统设计说明 2](#_Toc23547204)

[2.1功能模块设计 3](#_Toc23547205)

[2.1.1功能模块图 3](#_Toc23547206)

[2.1.2具体模块设计 3](#_Toc23547207)

[3.系统实现 17](#_Toc23547208)

[3.1关键代码 17](#_Toc23547209)

[3.2程序运行界面 23](#_Toc23547210)

[4.系统的测试和评价说明 24](#_Toc23547211)

[4.1系统测试结果 24](#_Toc23547212)

[4.1.1注册测试数据及结果 24](#_Toc23547213)

[4.1.2登录测试数据及结果 26](#_Toc23547214)

[4.1.3地点搜索测试数据及结果 27](#_Toc23547215)

[4.2系统的特点 28](#_Toc23547216)

[4.3系统的缺点 28](#_Toc23547217)

[5.实习总结 29](#_Toc23547218)

[5.1遇到的问题及解决 29](#_Toc23547219)

[5.2负责的模块 30](#_Toc23547220)

[5.3收获体会 30](#_Toc23547221)

# 1.系统分析说明

本系统为基于Android开发的地图导航应用程序，通过使用高德地图的API实现地图的显示与路线规划等功能，同时自己开发了登录注册等其他功能来完善整个软件的功能，使软件的各功能完整可用。

## 1.1项目可行性分析

· 软件的功能主要围绕地图导航展开，高德地图官方提供了丰富的API和相关开发文档，有一定的资源可以借鉴开发，且相关开发技术已经成熟，在技术上可以实现。

· 小组成员有相关Java开发的经验，具备完成项目的能力，在实现上可行。

## 1.2用户需求分析

本软件（鸿鹄导航，英文：Swan，后面称鸿鹄导航）主要通过使用高德地图API完成基本的地图显示及放大缩小、实时定位、地址搜索、规划路线、导航、搜索历史记录等功能，并自定义实现注册登录功能。并在注册登录、搜索历史记录模块实现数据持久化，同时，路线规划提供驾车、公交、步行、跨城公交四种方式的路线规划，并且会计算出打车费用，以及驾车、公交、步行所需的时间，还有详细的路线规划详情，另外，还提供了全国各城市的离线地图下载。

### 1.2.1功能需求

（1）地图的显示

（2）地图的放大缩小

（3）实时定位

（4）地址搜索

（5）路线规划

（6）驾车导航

（7）公交路线

（8）步行导航

（9）跨城公交路线

（10）搜索的历史记录（数据持久化）

（11）登录（数据持久化）

（12）注册（数据持久化）

（13）离线地图下载

# 2.系统设计说明

在这个系统中，主要采用了面对对象的原则，并且采取了一系列的设计原则来降低耦合度。如采用单例模式等等。将功能封装到相关的类中，然后接口让主线程进行调用。

相关功能主要参考高德地图API进行设计。

## 2.1功能模块设计

### 2.1.1功能模块图

见图2-1 功能模块图：

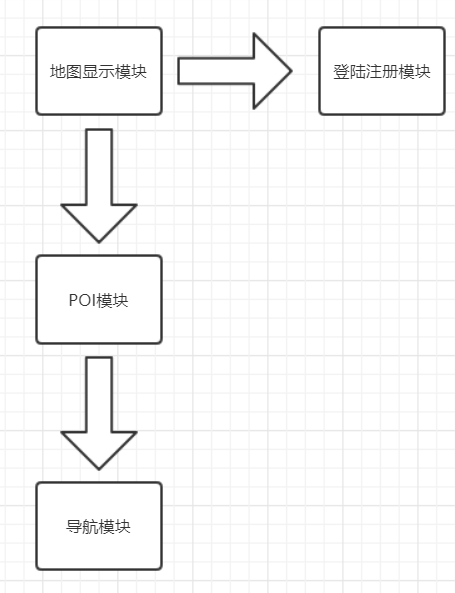


图2-1 功能模块图

### 2.1.2具体模块设计

#### 2.1.2.1地图显示模块

第一步：配置AndroidManifest.xml

首先，声明权限

//地图SDK（包含其搜索功能）需要的基础权限

<!--允许程序打开网络套接字-->

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

<!--允许程序设置内置sd卡的写权限-->

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

<!--允许程序获取网络状态-->

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

<!--允许程序访问WiFi网络信息-->

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

<!--允许程序读写手机状态和身份-->

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

<!--允许程序访问CellID或WiFi热点来获取粗略的位置-->

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

然后，设置高德Key

在application标签中加入如下内容：

<meta-data  
 android:name="com.amap.api.v2.apikey"  
 android:value="开发者申请的key" />

第二步：向工程中添加地图开发包

通过Gradle集成SDK：

在Project的build.gradle文件中配置repositories，添加maven或jcenter仓库地址,

配置如下：

allprojects {

repositories {

jcenter() // 或者 mavenCentral()

}

}

在主工程的build.gradle文件配置dependencies

配置如下：

android {

defaultConfig {

ndk {

//设置支持的SO库架构（开发者可以根据需要，选择一个或多个平台的so）

abiFilters "armeabi", "armeabi-v7a", "arm64-v8a", "x86","x86\_64"

}

}

}

dependencies {

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

//3D地图so及jar

compile 'com.amap.api:3dmap:latest.integration'

//定位功能

compile 'com.amap.api:location:latest.integration'

//搜索功能

compile 'com.amap.api:search:latest.integration'

}

第三步，初始化地图容器

首先在布局xml文件中添加地图控件:

<com.amap.api.maps.MapView

android:id="@+id/map"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"/>

第四步，显示地图

//定义了一个地图view

mapView = (MapView) findViewById(R.id.map);

mapView.onCreate(savedInstanceState);// 此方法须覆写，虚拟机需要在很多情况下保存地图绘制的当前状态。

//初始化地图控制器对象

AMap aMap;

if (aMap == null) {

aMap = mapView.getMap();

}

显示地图：见图2-2 显示地图



图2-2 显示地图

#### 2.1.2.2注册模块

第一步：在app/src/main/res/layout/activity\_register.xml中实现注册页面的布局

布局架构：

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

<android.support.constraint.ConstraintLayout>

<ScrollView>

<LinearLayout>

<com.qmuiteam.qmui.widget.QMUITopBar>

<ImageView/>

<TextView/>

</com.qmuiteam.qmui.widget.QMUITopBar>

<!--头部的logo和标题-->

<LinearLayout>

<ImageView/>

<TextView/>

</LinearLayout>

<!--注册信息输入框-->

<LinearLayout>

<!--账号文字-->

<TextView/>

<!--账号输入框-->

<EditText/>

<!--密码文字-->

<TextView/>

<!--密码输入框-->

<EditText/>

<!--确认密码文字-->

<TextView/>

<!--确认密码输入框-->

<EditText/>

<!--验证码文字-->

<TextView/>

<LinearLayout>

<!--验证码输入框-->

<EditText/>

<!--图形验证码-->

<ImageView/>

</LinearLayout>

</LinearLayout>

<!--注册按钮-->

<Button/>

</LinearLayout>

</ScrollView>

</android.support.constraint.ConstraintLayout>

第二步：在app/src/main/java/com/example/swan/activity/RegisterActivity.java中添加引用布局文件的代码：

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_register);

getSupportActionBar().hide();

}

第三步：在app/src/main/AndroidManifest.xml中添加注册文件：

<activity

android:name=".activity.RegisterActivity"

android:parentActivityName=".activity.LoginActivity" />

注册页面显示：见图2-3注册页面显示



图2-3 注册页面显示

第四步：在app/src/main/java/com/example/swan/activity/RegisterActivity.java中实现注册验证的处理，代码较长，见系统源码。

#### 2.1.2.3登录模块

第一步：在app/src/main/res/layout/activity\_login.xml中实现登录页面的布局

布局架构:

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

<android.support.constraint.ConstraintLayout>

<ScrollView>

<LinearLayout>

<com.qmuiteam.qmui.widget.QMUITopBar>

<ImageView/>

<TextView/>

</com.qmuiteam.qmui.widget.QMUITopBar>

<!--头部的logo和标题-->

<LinearLayout>

<ImageView/>

<TextView/>

</LinearLayout>

<!--登录信息输入框-->

<LinearLayout>

<TextView/>

<EditText/>

<LinearLayout>

<TextView/>

<TextView/>

</LinearLayout>

<EditText/>

<Button/>

</LinearLayout>

<!--新用户注册-->

<LinearLayout>

<TextView/>

<TextView/>

</LinearLayout>

</LinearLayout>

</ScrollView>

</android.support.constraint.ConstraintLayout>

第二步：在app/src/main/java/com/example/swan/activity/LoginActivity.java中添加引用布局文件的代码：

引用代码：

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

getSupportActionBar().hide();

}

第三步：在app/src/main/AndroidManifest.xml中注册文件：

<activity

android:name=".activity.LoginActivity"

android:parentActivityName=".activity.MenuActivity" />

登录页面显示：见图2-4 登录页面显示

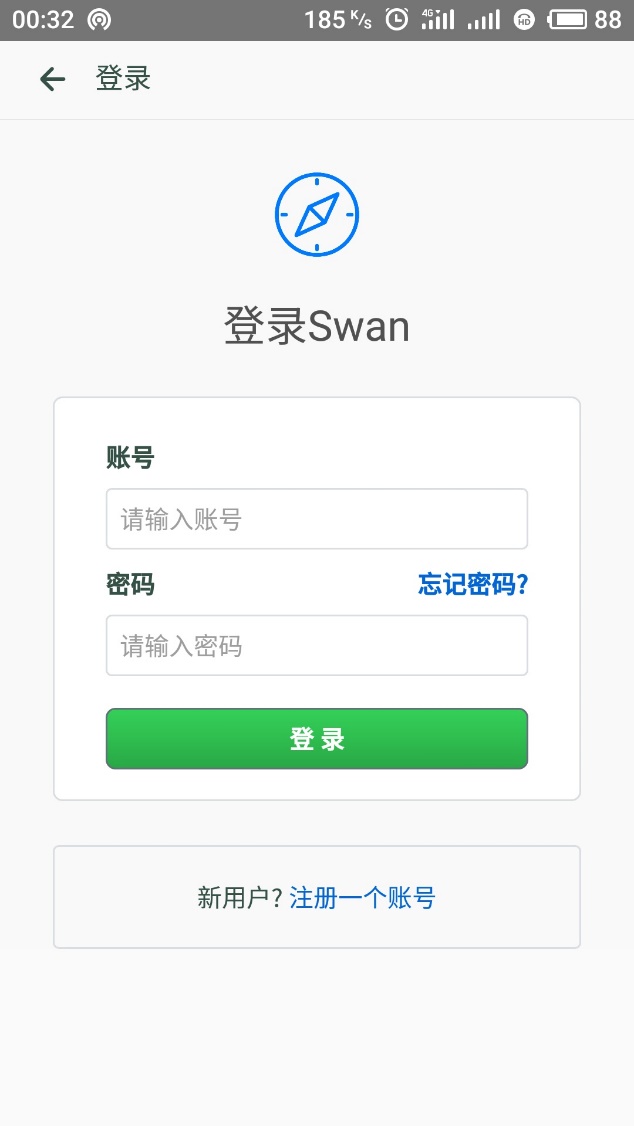


图2-4 登录显示

第四步：在app/src/main/java/com/example/swan/activity/LoginActivity.java中进行登录验证处理，代码较长，可参见系统源码。

#### 2.1.2.4忘记密码模块

第一步：app/src/main/res/layout/activity\_forget\_pwd.xml中实现忘记密码页面布局

布局架构：

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

<android.support.constraint.ConstraintLayout>

<ScrollView>

<LinearLayout>

<com.qmuiteam.qmui.widget.QMUITopBar>

<ImageView/>

<TextView/>

</com.qmuiteam.qmui.widget.QMUITopBar>

<!--头部的logo和标题-->

<LinearLayout>

<ImageView/>

<TextView/>

</LinearLayout>

<!--修改密码信息输入框-->

<LinearLayout>

<!--账号文字-->

<TextView/>

<!--账号输入框-->

<EditText/>

<!--新密码文字-->

<TextView/>

<!--新密码输入框-->

<EditText/>

<!--确认新密码文字-->

<TextView/>

<!--确认新密码输入框-->

<EditText/>

</LinearLayout>

<!--修改密码按钮-->

<Button/>

</LinearLayout>

</ScrollView>

</android.support.constraint.ConstraintLayout>

第二步：在app/src/main/java/com/example/swan/activity/ForgetPwdActivity.java中添加引用布局文件的代码：

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_forget\_pwd);

getSupportActionBar().hide();

}

第三步：在app/src/main/AndroidManifest.xml中添加注册文件：

<activity

android:name=".activity.ForgetPwdActivity"

android:parentActivityName=".activity.LoginActivity" />

第四步：在app/src/main/java/com/example/swan/activity/ForgetPwdActivity.java中实现处理忘记密码的逻辑，具体代码参加系统源码。

忘记密码页面显示：见图2-5 忘记密码显示



图2-5 忘记密码页面显示

#### 2.1.2.5地点搜索模块

第一步：在app/src/main/res/layout/activity\_search.xml中实现搜索框的显示，布局架构：

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

<LinearLayout>

<com.qmuiteam.qmui.widget.QMUIWindowInsetLayout>

<com.qmuiteam.qmui.widget.QMUITopBar>

<SearchView>

</SearchView>

</com.qmuiteam.qmui.widget.QMUITopBar>

</com.qmuiteam.qmui.widget.QMUIWindowInsetLayout>

<!-- 清空历史记录 -->

<LinearLayout>

<ListView/>

<TextView/>

</LinearLayout>

</LinearLayout>

在app/src/main/java/com/example/swan/activity/SearchActivity.java中调整搜索框的样式：

TextView searchText = (TextView) searchView.findViewById(searchView

.getContext()

.getResources()

.getIdentifier("android:id/search\_src\_text", null, null));

searchText.setTextSize(TypedValue.COMPLEX\_UNIT\_SP, 14);

searchText.setTextColor(Color.parseColor("#365146"));

searchText.setHintTextColor(Color.parseColor("#9e9e9e"));

第二步：在app/src/main/java/com/example/swan/activity/SearchActivity.java中引用布局文件：

setContentView(R.layout.activity\_search);

getSupportActionBar().hide();

第三步：在app/src/main/AndroidManifest.xml中注册文件：

<activity android:name=".activity.SearchActivity" />

第四步：在app/src/main/java/com/example/swan/activity/SearchActivity.java实现搜索的逻辑处理，代码较长，请参见系统源码。

地点搜索显示：见图2-6 地点搜索显示



图2-6 地点搜索显示

#### 2.1.2.6路线规划模块

第一步：在app/src/main/res/layout/main\_bottom.xml中实现显示路线的可点击按钮

布局架构：

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

<RelativeLayout>

<LinearLayout>

<!-- 搜附近 -->

<TextView/>

<View/>

<!-- 导航 -->

<TextView/>

<View/>

<!-- 路线 -->

<TextView/>

</LinearLayout>

</RelativeLayout>

第二步：在app/src/main/java/com/example/swan/route/RouteActivity.java中实现显示路线的布局：

private RelativeLayout mBottomLayout;

mBottomLayout = (RelativeLayout) findViewById(R.id.bottom\_layout);

第三步：在app/src/main/AndroidManifest.xml中注册文件：

<activity android:name=".route.RouteActivity" />

第四步：在app/src/main/java/com/example/swan/route/RouteActivity.java中设置监听事件，同时进行一系列的数据处理，主要包括：

初始化 RouteSearch 对象：

mRouteSearch = new RouteSearch(this);

设置数据回调监听器：

mRouteSearch.setRouteSearchListener(this);

设置搜索参数：

DriveRouteQuery query = new DriveRouteQuery(fromAndTo, mode, null,

null, "");// 第一个参数表示路径规划的起点和终点，第二个参数表示驾车模式，第三个参数表示途经点，第四个参数表示避让区域，第五个参数表示避让道路

发送请求：

mRouteSearch.calculateDriveRouteAsyn(query);// 异步路径规划驾车模式查询

接收数据：

public void onDriveRouteSearched(DriveRouteResult result, int errorCode) {

//具体代码参加系统源码

}

路线规划功能演示：见图2-7路线按钮显示、图2-8驾车路线显示、图2-9 路线详情显示、图2-10 公交路线显示、图2-11 步行路线显示

图2-7 路线按钮显示 图2-8 驾车路线显示



图2-9 路线详情显示 图2-10 公交路线显示



图2-11 步行路线显示

#### 2.1.2.7导航模块

第一步：在app/src/main/res/layout/activity\_navi.xml中定义AmapNaviView：

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<com.amap.api.navi.AMapNaviView

android:id="@+id/navi\_view"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_weight="1"/>

</LinearLayout>

第二步：实现 AMapNaviView 的生命周期

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

//获取 AMapNaviView 实例

mAMapNaviView = (AMapNaviView) findViewById(R.id.navi\_view);

mAMapNaviView.setAMapNaviViewListener(this);

mAMapNaviView.onCreate(savedInstanceState);

}

@Override

protected void onResume() {

super.onResume();

mAMapNaviView.onResume();

}

@Override

protected void onPause() {

super.onPause();

mAMapNaviView.onPause();

}

@Override

protected void onDestroy() {

super.onDestroy();

mAMapNaviView.onDestroy();

}

第三步：进行路线规划

第四步：开启导航

public void onCalculateRouteSuccess() {

super.onCalculateRouteSuccess();

mAMapNavi.startNavi(NaviType.GPS);

}

导航页面显示：见图2-12 导航显示

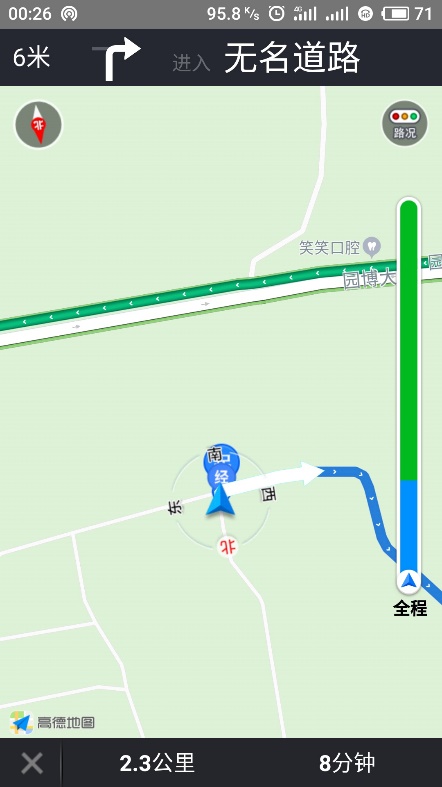


图2-12 导航显示

# 3.系统实现

## 3.1关键代码

**//MainActivity.java 部分代码**

@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 getSupportActionBar().hide();  
 checkPermission();  
 *//获取地图控件引用* mapView = findViewById(R.id.*main\_map*);  
 initTopBar();  
 *//在activity执行onCreate时执行mapView.onCreate(savedInstanceState)，创建地图* mapView.onCreate(savedInstanceState);  
 if (aMap == null) {  
 aMap = mapView.getMap();  
 setUpMap();  
 }  
 initMyLocation();  
 initSearch();  
 initRoute();  
}

**//SearchActivity.java 部分代码**

public class SearchActivity extends AppCompatActivity {

@Override

protected void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

Intent it = getIntent();

mCurrentCityName = it.getStringExtra("mCurrentCityName");

latLonPoint = new LatLonPoint(it.getDoubleExtra("startPointLat",25.063734),it.getDoubleExtra("startPointLon",110.300496));

setContentView(R.layout.activity\_search);

getSupportActionBar().hide();

init();

}

private void init() {

inputListView = findViewById(R.id.search\_list);

inputListView.setOnItemClickListener(itemClickListener);

initTopBar();

initSearchView();

}

private void initTopBar() {

topBar = findViewById(R.id.search\_top\_bar);

topBar.addLeftImageButton(R.drawable.ic\_icon\_back, 0).setOnClickListener(v -> finish());

QMUIButton searchButton = new QMUIButton(this);

topBar.addRightTextButton("搜索", searchButton.getId());

}

private void initSearchView() {

searchView = findViewById(R.id.search\_search\_view);

clearHistoryView = findViewById(R.id.search\_list\_clear);

//修改SearchView的样式

TextView searchText = (TextView) searchView.findViewById(searchView

.getContext()

.getResources()

.getIdentifier("android:id/search\_src\_text", null, null));

searchText.setTextSize(TypedValue.COMPLEX\_UNIT\_SP, 14);

searchText.setTextColor(Color.parseColor("#365146"));

searchText.setHintTextColor(Color.parseColor("#9e9e9e"));

searchView.setOnQueryTextListener(queryTextListener);

//设置SearchView默认为展开显示

searchView.setIconified(false);

searchView.onActionViewExpanded();

searchView.setSubmitButtonEnabled(false);

initHistory();

}

private void initHistory() {

LinkedList<String> stringList = AccountHelper.getHistorySearch(this);

LinkedList<Tip> tempTipList = new LinkedList<Tip>();

stringList.forEach(s -> {

Tip tip=null;

try {

tip= JSON.parseObject(s,Tip.class);

}catch (Exception e){

//如果反序列化失败

tip=new Tip();

tip.setName(s);

tip.setTypeCode("o(╯□╰)o");

}

tempTipList.add(tip);

});

inputTipsAdapter = new InputTipsAdapter(

getApplicationContext(),

tempTipList);

inputListView.setAdapter(inputTipsAdapter);

inputTipsAdapter.notifyDataSetChanged();

inputListView.setOnItemClickListener(itemClickListener);

clearHistoryView.setOnClickListener(v -> {

inputListView.setAdapter(null);

AccountHelper.clearHistorySearch(this);

});

}

SearchView.OnQueryTextListener queryTextListener = new SearchView.OnQueryTextListener() {

@Override

public boolean onQueryTextSubmit(String query) {

Intent intent = new Intent();

intent.putExtra("KeyWord", query);

setResult(MainActivity.RESULT\_CODE\_KEYWORDS, intent);

AccountHelper.writeHistorySearch(SearchActivity.this, query);

finish();

return false;

}

@Override

public boolean onQueryTextChange(String newText) {

Log.i("query触发了", newText);

if (!IsEmptyOrNullString(newText)) {

InputtipsQuery inputQuery = new InputtipsQuery(newText, mCurrentCityName);

inputQuery.setLocation(latLonPoint);

Inputtips inputTips = new Inputtips(SearchActivity.this.getApplicationContext(), inputQuery);

inputTips.setInputtipsListener(inputTipsListener);

inputTips.requestInputtipsAsyn();

} else {

if (inputTipsAdapter != null && currentTipList != null) {

currentTipList.clear();

inputTipsAdapter.notifyDataSetChanged();

}

}

return false;

}

};

Inputtips.InputtipsListener inputTipsListener = new Inputtips.InputtipsListener() {

@Override

public void onGetInputtips(List<Tip> tipList, int rCode) {

if (rCode == 1000) {// 正确返回

currentTipList = tipList;

List<String> listString = new ArrayList<String>();

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

listString.add(tipList.get(i).getName());

}

inputTipsAdapter = new InputTipsAdapter(

getApplicationContext(),

currentTipList);

inputListView.setAdapter(inputTipsAdapter);

inputTipsAdapter.notifyDataSetChanged();

} else {

ToastUtil.showerror(SearchActivity.this, rCode);

}

}

};

AdapterView.OnItemClickListener itemClickListener = new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> adapterView, View view, int position, long id) {

if (currentTipList != null) {

Tip tip = (Tip) adapterView.getItemAtPosition(position);

Intent intent = new Intent();

intent.putExtra("ExtraTip", tip);

AccountHelper.writeHistorySearch(SearchActivity.this, JSON.toJSONString(tip));

setResult(MainActivity.RESULT\_CODE\_INPUT\_TIPS, intent);

finish();

}else {

//在初始化历史纪录时

Tip tip = (Tip) adapterView.getItemAtPosition(position);

if(!tip.getTypeCode().equals("o(╯□╰)o")){

Intent intent = new Intent();

intent.putExtra("ExtraTip", tip);

AccountHelper.writeHistorySearch(SearchActivity.this, tip.getName());

setResult(MainActivity.RESULT\_CODE\_INPUT\_TIPS, intent);

finish();

}else {

Intent intent = new Intent();

intent.putExtra("KeyWord", tip.getName());

setResult(MainActivity.RESULT\_CODE\_KEYWORDS, intent);

AccountHelper.writeHistorySearch(SearchActivity.this, tip.getName());

finish();

}

}

}

};

}

**//RouteActivity.java 部分代码**

@Override

protected void onCreate(Bundle bundle) {

super.onCreate(bundle);

setContentView(R.layout.route\_activity);

mContext = this.getApplicationContext();

mapView = (MapView) findViewById(R.id.route\_map);

mapView.onCreate(bundle);// 此方法必须重写

init();

// getIntentData();

Intent it = getIntent();

mStartPoint = new LatLonPoint(it.getDoubleExtra("startPointLat",39.942295),it.getDoubleExtra("startPointLon",116.335891));

mEndPoint = new LatLonPoint(it.getDoubleExtra("endPointLat",39.942295),it.getDoubleExtra("endPointLon",116.335891));

mCurrentCityName = it.getStringExtra("mCurrentCityName");

mStartPoint\_bus = mStartPoint;

mEndPoint\_bus = mEndPoint;

setfromandtoMarker();

}

@Override

public void onDriveRouteSearched(DriveRouteResult result, int errorCode) {

dissmissProgressDialog();

aMap.clear();// 清理地图上的所有覆盖物

if (errorCode == AMapException.CODE\_AMAP\_SUCCESS) {

if (result != null && result.getPaths() != null) {

if (result.getPaths().size() > 0) {

mDriveRouteResult = result;

final DrivePath drivePath = mDriveRouteResult.getPaths()

.get(0);

if(drivePath == null) {

return;

}

DrivingRouteOverlay drivingRouteOverlay = new DrivingRouteOverlay(

mContext, aMap, drivePath,

mDriveRouteResult.getStartPos(),

mDriveRouteResult.getTargetPos(), null);

drivingRouteOverlay.setNodeIconVisibility(false);//设置节点marker是否显示

drivingRouteOverlay.setIsColorfulline(true);//是否用颜色展示交通拥堵情况，默认true

drivingRouteOverlay.removeFromMap();

drivingRouteOverlay.addToMap();

drivingRouteOverlay.zoomToSpan();

mBottomLayout.setVisibility(View.VISIBLE);

int dis = (int) drivePath.getDistance();

int dur = (int) drivePath.getDuration();

String des = AMapUtil.getFriendlyTime(dur)+"("+AMapUtil.getFriendlyLength(dis)+")";

mRotueTimeDes.setText(des);

mRouteDetailDes.setVisibility(View.VISIBLE);

int taxiCost = (int) mDriveRouteResult.getTaxiCost();

mRouteDetailDes.setText("打车约"+taxiCost+"元");

mBottomLayout.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(mContext,

DriveRouteDetailActivity.class);

intent.putExtra("drive\_path", drivePath);

intent.putExtra("drive\_result",

mDriveRouteResult);

startActivity(intent);

}

});

} else if (result != null && result.getPaths() == null) {

Toast.makeText(RouteActivity.this,R.string.poi\_no\_result,Toast.LENGTH\_SHORT).show();

}

} else {

Toast.makeText(RouteActivity.this,R.string.poi\_no\_result,Toast.LENGTH\_SHORT).show();

}

} else {

ToastUtil.showerror(this.getApplicationContext(), errorCode);

}

}

**//HttpUtil.java 部分代码**

public static Map<String, String> synGetKaptcha() {

OkHttpClient client = new OkHttpClient();

Request request = new Request.Builder().url(CommonConfig.URL + "/request/kaptcha").build();

Map<String, String> resParams = new HashMap<>();

try {

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

if (response.isSuccessful() && response.body() != null) {

try {

Message message = JSON.parseObject(response.body().string(), Message.class);

if (message != null && message.isSuccess()) {

String kaptchaToken = message.getParams().get("kaptcha\_token");

String img = message.getParams().get("img");

if (kaptchaToken != null && img != null) {

resParams.put("kaptcha\_token", kaptchaToken);

resParams.put("img", img);

} else {

resParams = null;

}

}

} catch (Exception e) {

e.printStackTrace();

resParams = null;

}

}

} catch (IOException e) {

e.printStackTrace();

resParams = null;

}

return resParams;

}

## 3.2程序运行界面

见上面2.1.2中具体模块设计中的：

图2-2 显示地图、图2-3 注册页面显示、图2-4 登录显示、图2-5 忘记密码页面显示、图2-6 地点搜索显示、图2-7 路线按钮显示、图2-8 驾车路线显示、图2-9 路线详情显示、图2-10 公交路线显示、图2-11 步行路线显示、图2-12 导航显示。

# 4.系统的测试和评价说明

## 4.1系统测试结果

### 4.1.1注册测试数据及结果

**（1）正确的输入输出**

**第一组测试数据**

**正确的输入：**

账号：xushicheng

密码：654123

确认密码：654123

验证码：n18m

（注：验证码为图形验证码，需要输入图形验证码中的字符，点击图形验证码可更新，每次的验证码均为随机）

**正确的输出：**

点击注册按钮后：提示注册成功，并自动跳转到登录页面

**第二组测试数据**

**正确的输入：**

账号：ceshizhanghao

密码：ceshimima

确认密码：ceshimima

验证码：2pw7

（注：验证码为图形验证码，需要输入图形验证码中的字符，点击图形验证码可更新，每次的验证码均为随机）

**正确的输出：**

点击注册按钮后：提示注册成功，并自动跳转到登录页面

**第三组测试数据**

**正确的输入：**

账号：测试

密码：测试

确认密码：测试

验证码：3px3

（注：验证码为图形验证码，需要输入图形验证码中的字符，点击图形验证码可更新，每次的验证码均为随机）

**正确的输出：**

点击注册按钮后：提示注册成功，并自动跳转到登录页面

**（2）错误的输入及其输出**

**第一组测试数据**

**错误的输入：**

账号：test3

密码：test3

确认密码：tets3

验证码：ae47

（注：验证码为图形验证码，需要输入图形验证码中的字符，点击图形验证码可更新，每次的验证码均为随机）

**输出结果：**

点击注册按钮后：提示两次密码不同，仍停留在注册页面

**第二组测试数据**

**错误的输入：**

账号：test1

密码：test1

确认密码：test1

验证码：2dd2

**输出结果：**

点击注册按钮后：提示不可重复注册，仍停留在注册页面

**第三组测试数据**

**错误的输入：**

账号：test1

密码：test1

确认密码：test1

验证码：2dd

**输出结果：**

点击注册按钮后：提示Kaptcha auth error!。表示验证码输入错误，仍停留在注册页面。

测试截图见：图4-1 注册成功、图4-2 重复注册、图4-3 登录输入。



图4-1 注册成功 图4-2 重复注册 图4-3 登录输入

### 4.1.2登录测试数据及结果

**（1）正确的输入输出**

**第一组测试数据**

**正确的输入：**

账号：test1

密码：test1

**正确的输出：**

点击登录按钮后：直接跳转到个人中心页面

**第二组测试数据**

**正确的输入：**

账号：test2

密码：test2

**正确的输出：**

点击登录按钮后：直接跳转到个人中心页面

**第三组测试数据**

**正确的输入：**

账号：测试

密码：测试

**正确的输出：**

点击登录按钮后：直接跳转到个人中心页面

**（2）错误的输入及其输出**

**第一组测试数据**

**错误的输入：**

账号：test1

密码：tes1

**输出结果：**

点击登录按钮后：提示账号密码不匹配，仍停留在登录页面

**第二组测试数据**

**错误的输入：**

账号：tes2

密码：test2

**输出结果：**

点击登录按钮后：提示不存在该用户，仍停留在登录页面

**第三组测试数据**

**错误的输入：**

账号：tes2

密码：test2

**输出结果：**

点击登录按钮后：提示不存在该用户，仍停留在登录页面

测试截图：见图4-4 个人中心、图4-5 登录失败、图4-6 登录失败。

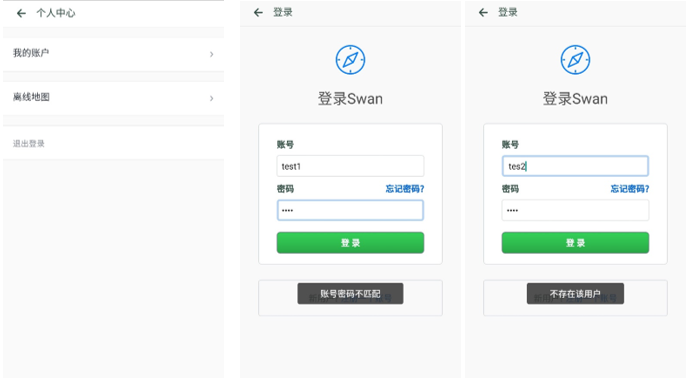


图4-4 个人中心 图4-5 登录失败 图4-6 登录失败

### 4.1.3地点搜索测试数据及结果

**第一组测试数据**

**输入：**桂林理工大学

**显示：**如 图4-7 搜索地点

**第二组测试数据**

**输入：银行**

**显示：**如 图4-8 搜索银行

**第三组测试数据**

**输入：景区**

**显示：**如 图4-9 搜索景区



图4-7 搜索地点 图4-8 搜索银行 图4-9 搜索景区

## 4.2系统的特点

（1）基本实现了一个完整的导航应用的基础功能，完成了最初制定的功能需求。

（2）页面简洁美观，操作方便。

（3）运行稳定，导航方便。

## 4.3系统的缺点

（1）仍有部分功能存在未知bug。

（2）代码较冗杂，代码结构亟待优化。

（3）用户交互仍待优化。

# 5.实习总结

## 5.1遇到的问题及解决

（1）**问题：**页面的部分TopBar的样式调整

页面的部分TopBar使用了QMUI的组件，但是QMUI的开发文档非常的简略，在没有文档的情况下修改样式是比较麻烦的一件事，有一些组件的样式调整起来较困难。

**解决：**通过查看QMUI相关组件的源码并且查看相关技术博客，逐步解决相关问题，达到想要的样式效果。

（2）**问题：**登录、注册页面的设计

登录、注册页面的设计与样式调整较为复杂和细致。登录、注册页面元素丰富，但要实现样式简洁清晰美观，需要付出较多的时间，在一些样式和细节的调整上是较为困难的。

**解决：**登录、注册页面参考GitHub的登录、注册页面进行设计，充分利用线性布局的优势，并结合ImageView、TextView、EditText、Button等组件的特点进行布局上的优化和美化。利用长达10多个小时的时间进行不断的调整，终于实现了想要的效果。

（3）**问题：**注册页面图形验证码的样式调整

由于图形验证码是由服务器端返回的位图，在尺寸上不符合要求，所以需要对返回的位图进行格式的调整，这一部分较为困难。

**解决：**通过查阅相关文档，了解了Android中对位图样式的调整方法，最终通过一些细节上的微调，达到理想效果。

（4）**问题：**搜索页面搜索框的样式调整

由于搜索框使用了Android本身提供的SearchView组件，而该组件的样式又被写死了，想要自定义修改是很麻烦的一件事，查阅了很多的解决办法都无法理想的实现。

**解决：**通过详细阅读关于SearchView组件的介绍文档，以及尝试多种解决办法，最后综合的进行参考设计，综合使用XML文档中相关属性的设计调整和使用java代码的相关操作方法，综合调整，最终达到理想效果。

（5）**问题：**历史记录的ListView展示。

由于历史记录的内容不固定，当历史记录较多，超过一个页面的内容时，清空历史记录的按钮就被挤掉了，无法正常显示。想要实现当历史记录较少时，清空历史记录的按钮紧跟在历史记录的后面，当历史记录较多时，清空历史记录固定在页面底部，清空历史记录的按钮随历史记录的内容进行变化是比较不好处理的。

**解决：**将ListView与TextView放在线性布局中，设置LinerLayout的layout\_height=”wrap\_content”，orientation=”vertical”。ListView的layout\_height=”0dp”,layout\_weight=”1”。TextView的layout\_height=”wrap\_content。充分利用线性布局的layout\_weight属性实现效果。

（6）**问题：**其它部分细节的样式调整问题。

相关显示的页面还是比较多的，有很多细节上的样式需要慢慢的调整。需要耗费较多时间。

**解决：**反复调整，通过多次真机实测，查看效果，逐步调整。

## 5.2负责的模块

主要是登录、注册、忘记密码、地点搜索、搜索历史记录、页面布局与设计。

## 5.3收获体会

经过为期一个月的生产实习，使我在技能方面、代码阅读、代码改错、规范检查等方面有了一定的提高，训练了自己编写程序的熟练度和规范性，积累了一定的项目开发经验，通过完成项目案例和阶段项目实战，增加了自己对实际软件项目开发的体验，同时，项目组角色分配和技术研讨，培养我养成良好的职业习惯，逐步达到熟练掌握移动应用程序的开发。

在实习期间，也是遇到了各种各样的问题，感觉像是每天都在写Bug，写完Bug又改Bug。由于对知识掌握得不系统、不熟练、不扎实，在写的过程中就不停的在出问题，但是，通过不断的去努力想办法解决这些问题，自己也有了很大的提高。一些问题的解决过程很痛苦，但是，当解决了的时候也会感到非常的高兴。在这一次的实习中，我主要是负责页面的设计与美化，在Android开发中，页面的呈现主要是通过.xml文档来进行设计，在xml文件中，利用Android本身提供的相关View组件和一些ViewGroup及相关布局，结合特定的属性，来实现想要的效果。这个过程是有趣但繁琐的，同时也存在着很多的技术难点，比如自定义SearchView的样式，较好的展示出ListView的效果等等，通过刻苦的钻研和理解以及不断的尝试和实践，最终也都克服了相关的困难，使页面能够达到理想的效果，友好的展示出来。

总之，解决了这些问题后，自己也掌握了很多的技巧与方法，对Android页面的常用布局以及常用组件的控制调整也更加的熟练，掌握了更多的相关属性。在这个过程中也总结了很多的经验技巧，比如自定义SearchView的样式，对位图的样式调整，ListView的使用等等。Android开发是一个漫长且艰辛的学习过程，后续还需要不断的学习和实践，多学多练才能更好的掌握Android开发的技能。