目 录

设计遍历全国省市县代码

根据创建的数据表，建立与数据表对应的 province、city 和 county 三个类，创建 db 包。

1. 实体对象的创建

1.1 City 类

```
package com.coolweather.android.db;
import org.litepal.crud.DataSupport;
public class City extends DataSupport {
    private int id;
    private String cityName;
    private int cityCode;
    private int provinceId;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getCityName() {
        return cityName;
    }
    public void setCityName(String cityName) {
        this.cityName = cityName;
    }
    public int getCityCode() {
        return cityCode;
    }
    public void setCityCode(int cityCode) {
```

```
        this.cityCode = cityCode;
    }
    public int getProvinceId() {
        return provinceId;
    }
    public void setProvinceId(int provinceId) {
        this.provinceId = provinceId;
    }
}
```

id 是主键，cityName 是市名，cityCode 是市代码，provinceId 是市所属省的 id 值。

1.2 County 类

```
package com.coolweather.android.db;
import org.litepal.crud.DataSupport;
public class County extends DataSupport {
    private int id;
    private String countyName;
    private String weatherId;
    private int cityId;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getCountyName() {
        return countyName;
    }
    public void setCountyName(String countyName) {
```

```
        this.countyName = countyName;
    }
    public String getWeatherId() {
        return weatherId;
    }
    public void setWeatherId(String weatherId) {
        this.weatherId = weatherId;
    }
    public int getCityId() {
        return cityId;
    }
    public void setCityId(int cityId) {
        this.cityId = cityId;
    }
}
```

id 是主键，countyName 是区县名，weatherId 是区县对应的天气 id，cityId 当前区县所属市的 id 值。

1.3 Province 类

```
package com.coolweather.android.db;
import org.litepal.crud.DataSupport;
public class Province extends DataSupport {
    private int id;
    private String provinceName;
    private int provinceCode;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
```

```
    }
    public String getProvinceName() {
        return provinceName;
    }
    public void setProvinceName(String provinceName) {
        this.provinceName = provinceName;
    }
    public int getProvinceCode() {
        return provinceCode;
    }
    public void setProvinceCode(int provinceCode) {
        this.provinceCode = provinceCode;
    }
}
```

id 是主键，provinceName 是省的名字，provinceCode 省的代码。每个实体类都必须要继承 DataSupport 类。

实体类声明属性后，并生成相应的 getter/setter 方法。

2. 配置 litepal.xml 文件

```
<?xml version="1.0" encoding="utf-8"?>
<litepal>
    <dbname value="cool_weather" />
    <version value="1" />
    <list>
        <mapping class="com.coolweather.android.db.Province" />
        <mapping class="com.coolweather.android.db.City" />
        <mapping class="com.coolweather.android.db.County" />
    </list>
</litepal>
```

定义数据库名为 Cool_weather，数据库版本为 1，并将 province、city 和

county 三个实体类添加到映射列表中。

在 AndroidManifest.xml 中配置 LitePalApplication。

```
<application
android:name="org.litepal.LitePalApplication"
android:allowBackup="true"
android:icon="@mipmap/logo"
android:label="@string/app_name"
android:supportsRtl="true"
android:theme="@style/AppTheme">
<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER"
/>
    </intent-filter>
</activity>
<activity android:name=".WeatherActivity" />
<service
    android:name=".service.AutoUpdateService"
    android:enabled="true"
    android:exported="true" />
</application>
```

3. 添加依赖库

使用 LitePal 管理数据库，在 build.gradle 文件中添加依赖包。

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    implementation 'com.android.support:appcompat-v7:28.0.0-rc01'
     implementation        'com.android.support.constraint:constraint-
```

```
layout:1.1.2'
    compile group: 'com.github.bumptech.glide', name: 'glide', version:
'4.8.0'
    compile group: 'org.litepal.android', name: 'core', version:
'2.0.0'
    compile group: 'com.squareup.okhttp3', name: 'okhttp', version:
'3.11.0'
    compile group: 'com.google.code.gson', name: 'gson', version:
'2.8.5'
}
```

LitePal 用于对数据库的操作，OkHttp 用于进行网络请求，GSON 用于解析 json 数据，Glide 用于加载和展示图片。

数据库和数据表在首次执行任意数据库操作时自动创建。

4. 遍历全国省市县数据

4.1 编写工具类

全国所有省市县数据都是从服务器端获取，需要编写与服务器交互代码，在 util 包下增加 HttpUtil 类，代码如下：

```
public static void sendOkHttpRequest(String address, okhttp3.Callback
callback) {
    OkHttpClient client = new OkHttpClient();
    Request request = new Request.Builder().url(address).build();
    client.newCall(request).enqueue(callback);
}
```

客户端发起 HTTP 请求只要调用 sendOkHttpRequest()方法，传入请求地址并注册一个回调来处理服务器响应就可以。

服务器返回的省市县数据都是 json 格式，编写一个工具类来解析和处理 json 数据，在 util 包下建 Utility 类，代码如下：

```
package com.coolweather.android.util;
```

```
import android.text.TextUtils;
import com.coolweather.android.db.City;
import com.coolweather.android.db.County;
import com.coolweather.android.db.Province;
import com.coolweather.android.gson.Weather;
import com.google.gson.Gson;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

public class Utility {
    /**
     * 解析和处理服务器返回的省级数据
     */
    public static boolean handleProvinceResponse(String response) {
        if (!TextUtils.isEmpty(response)) {
            try {
                JSONArray allProvinces = new JSONArray(response);
                for (int i = 0; i < allProvinces.length(); i++) {
                    JSONObject provinceObject =
allProvinces.getJSONObject(i);
                    Province province = new Province();

province.setProvinceName(provinceObject.getString("name"));

province.setProvinceCode(provinceObject.getInt("id"));
                    province.save();
                }
                return true;
```

```
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        return false;
    }
    /**
     * 解析和处理服务器返回的市级数据
     */
    public static boolean handleCityResponse(String response, int
provinceId) {
        if (!TextUtils.isEmpty(response)) {
            try {
                JSONArray allCities = new JSONArray(response);
                for (int i = 0; i < allCities.length(); i++) {
                    JSONObject cityObject =
allCities.getJSONObject(i);
                    City city = new City();
                    city.setCityName(cityObject.getString("name"));
                    city.setCityCode(cityObject.getInt("id"));
                    city.setProvinceId(provinceId);
                    city.save();
                }
                return true;
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        return false;
```

```
    }
    /**
     * 解析和处理服务器返回的县级数据
     */
    public static boolean handleCountyResponse(String response, int
cityId) {
        if (!TextUtils.isEmpty(response)) {
            try {
                JSONArray allCounties = new JSONArray(response);
                for (int i = 0; i < allCounties.length(); i++) {
                    JSONObject countyObject =
allCounties.getJSONObject(i);
                    County county = new County();

county.setCountyName(countyObject.getString("name"));

county.setWeatherId(countyObject.getString("weather_id"));
                    county.setCityId(cityId);
                    county.save();
                }
                return true;
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        return false;
    }
    /**
     * 将返回的JSON数据解析成Weather实体类
```

```
     */
    public static Weather handleWeatherResponse(String response) {
        try {
            JSONObject jsonObject = new JSONObject(response);
            JSONArray jsonArray = jsonObject.getJSONArray("HeWeather");
            String weatherContent = jsonArray.getJSONObject(0).toString();
            return new Gson().fromJson(weatherContent, Weather.class);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

handleProvinceResponse（）、handleCityResponse（）、handleCountyResponse（）三个方法分别用于处理服务器返回的省市县数据。

先使用 JSONArray 和 JSONObject 将数据解析出来，然后组装成实体类对象，再调用 save（）方法将数据存储到数据库。

4.2 编写界面

遍历全国省市县的功能在后面还会用到，因此写在碎片里面，在需要复用的地方直接在布局里引用碎片即可。

在 res/layout 下建 choose_area.xml 布局文件，代码如下：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
    android:background="#fff"
    android:fitsSystemWindows="true">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="?attr/colorPrimary">
        <TextView
            android:id="@+id/title_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:textColor="#fff"
            android:textSize="20sp"/>
        <Button
            android:id="@+id/back_button"
            android:layout_width="25dp"
            android:layout_height="25dp"
            android:layout_marginLeft="10dp"
            android:layout_alignParentLeft="true"
            android:layout_centerVertical="true"
            android:background="@drawable/ic_back"/>
    </RelativeLayout>
    <ListView
        android:id="@+id/list_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
</LinearLayout>
```

定义一个头布局作为标题栏，将布局高度设置为 actionBar 的高度，背景色设置为 colorPrimary，在头布局中使用一个 TextView 用于显示标题内容，使用

一个 Button 用于执行返回操作，使用 ic_back.pngp 图片用于按钮的背景图。

这里之所以自己定义标题栏，是因为碎片中最好不要直接使用 ActionBar 或 ToolBar，不然在复用的时候可能会出现异常。

在头布局中定义了一个 ListView，省市县的数据将显示在这里。ListView 会自动给每个子项之间添加一条分隔线，而如果使用 RecyclerView 实现同样的效果比较麻烦。

下面编写遍历省市县数据的碎片

4.3 程序编写

建 ChooseAreaFragment 继承 Fragment，代码如下：

```
package com.coolweather.android;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import com.coolweather.android.db.City;
import com.coolweather.android.db.County;
import com.coolweather.android.db.Province;
import com.coolweather.android.util.HttpUtil;
import com.coolweather.android.util.Utility;
```

```
import org.litepal.crud.DataSupport;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.Response;
public class ChooseAreaFragment extends Fragment {
    private static final String TAG = "ChooseAreaFragment";
    public static final int LEVEL_PROVINCE = 0;
    public static final int LEVEL_CITY = 1;
    public static final int LEVEL_COUNTY = 2;
    private ProgressDialog progressDialog;
    private TextView titleText;
    private Button backButton;
    private ListView listView;
    private ArrayAdapter<String> adapter;
    private List<String> dataList = new ArrayList<>();
    /**
     * 省列表
     */
    private List<Province> provinceList;
    /**
     * 市列表
     */
    private List<City> cityList;
    /**
     * 县列表
     */
```

```
    private List<County> countyList;
    /**
     * 选中的省份
     */
    private Province selectedProvince;
    /**
     * 选中的城市
     */
    private City selectedCity;
    /**
     * 当前选中的级别
     */
    private int currentLevel;
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.choose_area, container,
false);
        titleText = (TextView) view.findViewById(R.id.title_text);
        backButton = (Button) view.findViewById(R.id.back_button);
        listView = (ListView) view.findViewById(R.id.list_view);
        adapter = new ArrayAdapter<>(getContext(),
android.R.layout.simple_list_item_1, dataList);
        listView.setAdapter(adapter);
        return view;
    }
    @Override
    public void onActivityCreated(Bundle savedInstanceState) {
```

```
        super.onActivityCreated(savedInstanceState);
        listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view,
int position, long id) {
                if (currentLevel == LEVEL_PROVINCE) {
                    selectedProvince = provinceList.get(position);
                    queryCities();
                } else if (currentLevel == LEVEL_CITY) {
                    selectedCity = cityList.get(position);
                    queryCounties();
                } else if (currentLevel == LEVEL_COUNTY) {
                    String                  weatherId                  =
countyList.get(position).getWeatherId();
                    if (getActivity() instanceof MainActivity) {
                        Intent  intent  =  new  Intent(getActivity(),
WeatherActivity.class);
                        intent.putExtra("weather_id", weatherId);
                        startActivity(intent);
                        getActivity().finish();
                    }    else    if    (getActivity()    instanceof
WeatherActivity) {
                        WeatherActivity activity = (WeatherActivity)
getActivity();
                        activity.drawerLayout.closeDrawers();
                        activity.swipeRefresh.setRefreshing(true);
                        activity.requestWeather(weatherId);
                    }
```

```
                }
            }
        });
        backButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentLevel == LEVEL_COUNTY) {
                    queryCities();
                } else if (currentLevel == LEVEL_CITY) {
                    queryProvinces();
                }
            }
        });
        queryProvinces();
    }
    /**
     * 查询全国所有的省，优先从数据库查询，如果没有查询到再去服务器上查询。
     */
    private void queryProvinces() {
        titleText.setText("中国");
        backButton.setVisibility(View.GONE);
        provinceList = DataSupport.findAll(Province.class);
        if (provinceList.size() > 0) {
            dataList.clear();
            for (Province province : provinceList) {
                dataList.add(province.getProvinceName());
            }
            adapter.notifyDataSetChanged();
```

```
            listView.setSelection(0);
            currentLevel = LEVEL_PROVINCE;
        } else {
            String address = "http://guolin.tech/api/china";
            queryFromServer(address, "province");
        }
    }
    /**
     * 查询选中省内所有的市，优先从数据库查询，如果没有查询到再去服务器上查询。
     */
    private void queryCities() {
        titleText.setText(selectedProvince.getProvinceName());
        backButton.setVisibility(View.VISIBLE);
        cityList = DataSupport.where("provinceid = ?",
String.valueOf(selectedProvince.getId())).find(City.class);
        if (cityList.size() > 0) {
            dataList.clear();
            for (City city : cityList) {
                dataList.add(city.getCityName());
            }
            adapter.notifyDataSetChanged();
            listView.setSelection(0);
            currentLevel = LEVEL_CITY;
        } else {
            int provinceCode = selectedProvince.getProvinceCode();
            String address = "http://guolin.tech/api/china/" +
provinceCode;
            queryFromServer(address, "city");
```

```
        }
    }
    /**
     * 查询选中市内所有的县，优先从数据库查询，如果没有查询到再去服务器上查询。
     */
    private void queryCounties() {
        titleText.setText(selectedCity.getCityName());
        backButton.setVisibility(View.VISIBLE);
        countyList = DataSupport.where("cityid = ?",
String.valueOf(selectedCity.getId())).find(County.class);
        if (countyList.size() > 0) {
            dataList.clear();
            for (County county : countyList) {
                dataList.add(county.getCountyName());
            }
            adapter.notifyDataSetChanged();
            listView.setSelection(0);
            currentLevel = LEVEL_COUNTY;
        } else {
            int provinceCode = selectedProvince.getProvinceCode();
            int cityCode = selectedCity.getCityCode();
            String address = "http://guolin.tech/api/china/" +
provinceCode + "/" + cityCode;
            queryFromServer(address, "county");
        }
    }
    /**
     * 根据传入的地址和类型从服务器上查询省市县数据。
```

```
     */
    private void queryFromServer(String address, final String type) {
        showProgressDialog();
        HttpUtil.sendOkHttpRequest(address, new Callback() {
            @Override
            public void onResponse(Call call, Response response)
throws IOException {
                String responseText = response.body().string();
                    boolean result = false;
                if ("province".equals(type)) {
                    result =
Utility.handleProvinceResponse(responseText);
                } else if ("city".equals(type)) {
                    result = Utility.handleCityResponse(responseText,
selectedProvince.getId());
                } else if ("county".equals(type)) {
                    result =
Utility.handleCountyResponse(responseText, selectedCity.getId());
                }
                if (result) {
                    getActivity().runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            closeProgressDialog();
                            if ("province".equals(type)) {
                                queryProvinces();
                            } else if ("city".equals(type)) {
                                queryCities();
                            } else if ("county".equals(type)) {
```

```
                                queryCounties();
                            }
                        }
                    });
                }
            }
            @Override
            public void onFailure(Call call, IOException e) {
                // 通过 runOnUiThread()方法回到主线程处理逻辑
                getActivity().runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        closeProgressDialog();
                        Toast.makeText(getContext(), "加载失败",
Toast.LENGTH_SHORT).show();
                    }
                });
            }
        });
    }
    /**
     * 显示进度对话框
     */
    private void showProgressDialog() {
        if (progressDialog == null) {
            progressDialog = new ProgressDialog(getActivity());
            progressDialog.setMessage("正在加载...");
            progressDialog.setCanceledOnTouchOutside(false);
        }
```

```
            progressDialog.show();
        }
        /**
         * 关闭进度对话框
         */
        private void closeProgressDialog() {
            if (progressDialog != null) {
                progressDialog.dismiss();
            }
        }
}
```

分析:在 onCreateView()方法中获取一些控件的实例,初始化 ArrayAdapter,并将它设置为 ListView 的适配器。在 onActivityCreated()方法中给 ListView 和 Button 设置点击事件。

在 onActivityCreated（）方法最后，调用 queryProvinces（）方法，开始加载省级数据。queryProvinces（）方法首先将布局的标题设置成中国，将返回按钮隐藏起来，因为省级列表已经不能再返回了。然后调用 LitePal 的查询接口来从数据库中读取省级数据，如果读到了就直接将数据显示到界面上，如果没有读到就组装出一个请求地址，然后调用 queryFromServer（）方法来从服务器上查询数据。

queryFromServer（）方法会调用 HttpUtil 的 sendOkHttpRequest（）方法来向服务器发送请求，响应的数据会回调到 onResponse（）方法中，然后去调用 Utility 的 handleProvincesResponse()方法来解析和处理服务器返回的数据，并存储到数据库中。

在解析和处理完数据之后，再次调用 queryProvince（）方法来重新加载省级数据，由于 queryProvince（）方法牵扯到 UI 操作，因此必须要在主线程中调用，在这里借助了 runOnUIThread（）方法来实现从子线程切换到主线程。现在数据库中已经存在了数据，因此调用 queryProvince（）方法可以直接将数据显示到界面。

当点击某个省进入到 ListView 的 onItemClick（）方法中，这个时候会根据当前级别来判断是去调用 queryCities（）方法还是 queryCounties（）方法。queryCities（）方法是查询市级市级，queryCounties（）方法是查询区县级数据，这两个方法内部实现流程和 queryProvinces（）方法基本相同。

在返回按钮的点击事件里，会对当前 ListView 的列表级别进行判断，如果当前是县级列表，那么就返回到市级列表；如果当前是市级列表，那么就返回到省级列表。当返回到省级列表时，返回按钮会自动隐藏。

把遍历全国省市县功能完成，但是碎片是不能直接显示在界面上的，因此必须要把它添加到活动中才行。修改 Activity_main.xml 中的代码，如下所示：

```
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent">
<fragment
    android:id="@+id/choose_area_fragment"
    android:name="com.coolweather.android.ChooseAreaFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
</FrameLayout>
```

定义一个 FrameLayout，然后将 ChooseAreaFragment 添加进来，并让它充满整个布局。因在碎片布局中已经自定义了一个标题栏，因此不再需要原生的 ActionBar，修改 res/values/styles.xml 的代码，如下所示：

```
<resources>
<!-- Base application theme. -->
<style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
```

```
    </style>
</resources>
```

在 colors.xml 中修改如下代码：

```
    <resources>
        <color name="colorPrimary">#3F51B5</color>
        <color name="colorPrimaryDark">#303F9F</color>
        <color name="colorAccent">#FF4081</color>
    </resources>
```

因为本程序需要访问网络，因此要声明程序所需的权限，修改 AndroidManifest.xml 代码，如下所示：

```
    <manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.coolweather.android">
    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:name="org.litepal.LitePalApplication"
        android:allowBackup="true"
        android:icon="@mipmap/logo"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
    ......
     </manifest>
```