# 《移动应用开发技术》

# 课程设计报告

学院名称： 信息技术系

班级名称： ZB计科194

学生姓名： 朱桐

学 号： 201901012428

日 期： 2020年6月

目录

[《移动应用开发技术》 0](#_Toc43884373)

[课程设计报告 0](#_Toc43884374)

[项目一、天气预报App 3](#_Toc43884375)

[背景意义 3](#_Toc43884376)

[1 项目需求分析 3](#_Toc43884377)

[1.1 需求分析 3](#_Toc43884378)

[1.2 功能需求 3](#_Toc43884379)

[2 总体设计 4](#_Toc43884380)

[2.1系统架构设计 4](#_Toc43884381)

[2.2系统功能体系 4](#_Toc43884382)

[3 系统详细设计 4](#_Toc43884383)

[3.1 数据库设计 4](#_Toc43884384)

[3.2系统界面设计 5](#_Toc43884385)

[3.2.1 主界面 5](#_Toc43884386)

[3.2.2 添加城市界面 10](#_Toc43884387)

[3.2.3 删除城市界面 13](#_Toc43884388)

[3.2.4 更多设置界面 15](#_Toc43884389)

[3.3 数据库存储设计 20](#_Toc43884390)

[4 系统编码实现 23](#_Toc43884391)

[4.1 项目框架 23](#_Toc43884392)

[4.2 交互实现 23](#_Toc43884393)

[4.3 单元测试 26](#_Toc43884394)

[5 系统测试发布 26](#_Toc43884395)

[项目二、理财App 28](#_Toc43884396)

[背景意义 28](#_Toc43884397)

[1 项目需求分析 28](#_Toc43884398)

[1.1 需求分析 28](#_Toc43884399)

[1.2 功能需求 28](#_Toc43884400)

[2 总体设计 29](#_Toc43884401)

[2.1系统架构设计 29](#_Toc43884402)

[2.2系统功能体系 29](#_Toc43884403)

[3 系统详细设计 30](#_Toc43884404)

[3.1 数据库设计 30](#_Toc43884405)

[3.2系统界面设计 31](#_Toc43884406)

[3.2.1 启动界面 31](#_Toc43884407)

[3.2.2 注册界面 32](#_Toc43884408)

[3.2.3 登录界面 35](#_Toc43884409)

[3.2.4 新增收入界面 38](#_Toc43884410)

[3.2.5 收入明细界面 41](#_Toc43884411)

[3.2.6新增支出界面 44](#_Toc43884412)

[3.2.7支出明细界面 47](#_Toc43884413)

[3.2.8数据分析界面 49](#_Toc43884414)

[3.2.9系统设置界面 56](#_Toc43884415)

[3.3 数据库存储设计 60](#_Toc43884416)

[4 系统编码实现 61](#_Toc43884417)

[4.1 项目框架 61](#_Toc43884418)

[4.2 交互实现 61](#_Toc43884419)

[4.3 单元测试 68](#_Toc43884420)

[5 系统测试发布 68](#_Toc43884421)

[参考文献 69](#_Toc43884422)

# 项目一、天气预报App

## 背景意义

随着移动终端的迅速普及，Android系统平台引用软件的需求随之增大。伴随着Android智能手机与平板电脑已经出现在我们生活的大量的使用，越来越多的基于Android开发平台也随之而出，为丰富人们使用Android智能产品的用途，使其可以帮人们提前了解未来的天气情况。本设计开发通过研究Android体系结构和百度天气API方面的知识，设计并实现了个人看天气系统。能够显示未来三天的天气情况，并且可以展示诸事事宜等功能。

## 1 项目需求分析

### 1.1 需求分析

物质和科技的飞速发展，人们的生活水平也不断的在提高，对于经常外出出门的人来说，未来的天气是一个必不可少的关注点，因为天气情况很有可能就会影响到他的外出，正因如此，人们很需要一个可以提前知道未来天气情况的App，在这个App上能够看到未来几天的天气情况，已做好相关的外出计划。以下是本软件的一些功能：

1. 查看界面：点击App进去后就可以看到自己添加的地区的天气情况
2. 添加城市：通过搜索功能添加想要的地区的天气
3. 删除城市：删除自己不想看到的地区的天气
4. 更多设置：可以修改背景壁纸，删除缓存等功能

### 1.2 功能需求

目前国内外查看天气软件已有上百种之多，如墨迹天气、爱天气等软件，虽然他们的软件也相当不错，但是界面不够简洁，操作复杂。本软件是针对个人用户的一款Android软件，主要对个人想要迅速了解未来的天气、出门注意事项等等情况。本软件该具备以下功能：

1. 功能操作要方便、易懂，不要有多余或复杂的操作。
2. 对天气城市的增加删除修改等操作。
3. 系统的功能复合本人的实际情况。
4. 界面直观简洁，方便用户更好的使用。
5. 保证软件的安全性。

## 2 总体设计

### 2.1系统架构设计

更多设置

删除城市

查看界面

添加城市

看天气APP

图 2-1 系统架构图

### 2.2系统功能体系

系统总体功能主要包括系统各功能模块的详细划分和定义，以及各子功能模块具体实现细节，通过详细划分和定义细节可以将系统各功能进行有效区分，防止功能重叠，功能冗余等现象，系统总体功能分析如表2-1所示.

|  |  |
| --- | --- |
| 用户功能 | 具体描述 |
| 查看界面 | 点击App进去后就可以看到自己添加的地区的天气情况 |
| 添加城市 | 通过搜索功能添加想要的地区的天气 |
| 删除城市 | 删除自己不想看到的地区的天气 |
| 更多设置 | 可以修改背景壁纸，删除缓存等功能 |

图 2-2 总体功能分析

## 3 系统详细设计

### 3.1 数据库设计

看天气App是一款运行在Android系统上的程序，在Android系统中，集成了一种轻量型的数据库，即SQLite，该数据库是使用C语言编写的开源嵌入式数据库，支持的数据库大小为2TB，使用该数据库，用户可以像使用SQL Server数据库或者Oracle数据库那样来存储、管理和维护数据，本系统采用了SQLite数据库，并且命名为forecast.db，该数据库中用到了1个数据表，即info表。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 大小 | 是否主键 | 是否为空 | 备注 |
| \_id | int | autoincrement | 是 | 否 | 数量多少 |
| city | varchar | 20 | 否 | 否 | 城市 |
| content | varchar | autoincrement | 否 | 否 | 天气信息 |

图 3-1 info表

### 3.2系统界面设计

#### 3.2.1 主界面

主界面清新脱俗，并且界面简洁大方，一眼就能识别出当天天气，并且有图标展示与之相呼应，中间部分是一个炸死你hi未来三天天气的布局，下面就是外出各种事宜的提醒。



图 3-2-1 展示界面

界面设计分四个部分

1. 第一部分：使用RelativeLayout（相对布局）设计上面天气以、地区、温度以及图标。
2. 第二部分：使用LinearLayout（线性布局）设计每一个未来天气的显示，共三个，内部采用垂直布局的方式。
3. 第三部分：使用LinearLayout（线性布局）设计，使用LmageView控件来分割LinearLayout，分成6分。
4. 第四部分：最底部的“+”与“更多”使用了RelativeLayout布局。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***MainActivity.java***

**public class** MainActivity **extends** AppCompatActivity **implements** View.OnClickListener {  
 ImageView **addCityIv**,**moreIv**;  
 LinearLayout **pointLayout**;  
 ViewPager **mainVp**;  
 RelativeLayout **outLayout**;  
 *//ViewPager的数据源* List<Fragment>**fragmentList**;  
 *//表示需要显示的城市的集合* List<String>**cityList**;  
 *//表示ViewPager的页数指示器显示集合* List<ImageView>**imgList**;  
 **private** CityFragmentPagerAdapter **adapter**;  
 **private** SharedPreferences **pref**;  
 **private int bgNum**;

@Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **addCityIv**=findViewById(R.id.***main\_iv\_add***);  
 **moreIv**=findViewById(R.id.***main\_iv\_more***);  
 **pointLayout**=findViewById(R.id.***main\_layout\_point***);  
 **outLayout**=findViewById(R.id.***main\_out\_layput***);  
 exchangeBg();  
 **mainVp**=findViewById(R.id.***main\_vp***);

*//添加点击事件* **addCityIv**.setOnClickListener(**this**);  
 **mainVp**.setOnClickListener(**this**);  
 **moreIv**.setOnClickListener(**this**);

*//初始化* **fragmentList** = **new** ArrayList<>();  
 **cityList** = DBManager.*queryALLCityNmae*(); *//获取数据库包含的城市信息列表* **imgList** =**new** ArrayList<>();  
 **if**(**cityList**.size()==0 ){  
 **cityList**.add(**"信阳"**);  
 }  
 */\*因为可能搜索界面点击跳转到此界面，所以次数获取对象\*/* Intent intent = getIntent();  
 String city = intent.getStringExtra(**"city"**);  
 **try** {  
 **if** (!**cityList**.contains(city)&&!TextUtils.*isEmpty*(city)) {  
 **cityList**.add(city);  
 }  
 }**catch** (Exception e){  
 Log.*i*(**"TT"**,**"程序书问题了"**);  
 }  
 *//初始化ViewPager页面的方法* initPager();  
 **adapter** = **new** CityFragmentPagerAdapter(getSupportFragmentManager(), **fragmentList**);  
 **mainVp**.setAdapter(**adapter**);  
 *//创建小圆点指示器* initPoint();  
 *//设置最后一个城市信息* **mainVp**.setCurrentItem(**fragmentList**.size()-1);  
 *//设置ViewPager页面监听* setPagerListener();  
 }  
 **private void** setPagerListener() {  
 *//设置监听事件* **mainVp**.addOnPageChangeListener(**new** ViewPager.OnPageChangeListener() {  
 @Override  
 **public void** onPageScrolled(**int** position, **float** positionOffset, **int** positionOffsetPixels) {  
 }  
 @Override  
 **public void** onPageSelected(**int** position) {  
 **for** (**int** i =0;i<**imgList**.size();i++){  
 **imgList**.get(i).setImageResource(R.mipmap.***a1***);  
 }  
 **imgList**.get(position).setImageResource(R.mipmap.***a2***);  
 }  
 @Override  
 **public void** onPageScrollStateChanged(**int** state) {  
 }  
 });  
 }  
 **private void** initPoint() {  
 *//创建小圆点 ViewPager页面指示器的函数* **for** (**int** i = 0;i<**fragmentList**.size();i++){  
 ImageView pIv = **new** ImageView(**this**);  
 pIv.setImageResource(R.mipmap.***a1***);  
 pIv.setLayoutParams(**new** LinearLayout.LayoutParams(LinearLayout.LayoutParams.***WRAP\_CONTENT***, LinearLayout.LayoutParams.***WRAP\_CONTENT***));  
 LinearLayout.LayoutParams lp = (LinearLayout.LayoutParams) pIv.getLayoutParams();  
 lp.setMargins(0,0,20,0);  
 **imgList**.add(pIv);  
 **pointLayout**.addView(pIv);  
 }  
 **imgList**.get(**imgList**.size()-1).setImageResource(R.mipmap.***a2***);  
 }  
 **private void** initPager() {  
 *//创建Fragment对象，添加到ViewPager数据源中* **for** (**int** i=0;i<**cityList**.size();i++){  
 CityWewtherFragment cwFrag = **new** CityWewtherFragment();  
 Bundle bundle = **new** Bundle();  
 bundle.putString(**"city"**,**cityList**.get(i));  
 cwFrag.setArguments(bundle);  
 **fragmentList**.add(cwFrag);  
 }  
 }  
 @Override  
 **public void** onClick(View v) {  
 Intent intent = **new** Intent();  
 **switch** (v.getId()){  
 **case** R.id.***main\_iv\_add***:  
 intent.setClass(**this**, CityManagerActivity.**class**);  
 **break**;  
 **case** R.id.***main\_iv\_more***:  
 intent.setClass(**this**, MoreActivity.**class**);  
 **break**;  
 }  
 startActivity(intent);  
 }  
 *//换壁纸的函数* **public void** exchangeBg(){  
 **pref** = getSharedPreferences(**"bg\_pref"**, ***MODE\_PRIVATE***);  
 **bgNum** = **pref**.getInt(**"bg"**, 2);  
 **switch** (**bgNum**) {  
 **case** 0:  
 **outLayout**.setBackgroundResource(R.mipmap.***bg***);  
 **break**;  
 **case** 1:  
 **outLayout**.setBackgroundResource(R.mipmap.***bg2***);  
 **break**;  
 **case** 2:  
 **outLayout**.setBackgroundResource(R.mipmap.***bg3***);  
 **break**;  
 }  
 }  
 */\*页面重新加载时回调用的函数，这个函数在页面获取焦点前进行调用，此处完成ViewPager页数的更新\*/* @Override  
 **protected void** onRestart() {  
 **super**.onRestart();  
 *//获取数据库中还剩下的城市列表* List<String> list = DBManager.*queryALLCityNmae*();  
 **if** (list.size()==0) {  
 list.add(**"信阳"**);  
 }  
 **cityList**.clear(); *//重写加载之前，清空原本的数据源* **cityList**.addAll(list);  
 *//剩下的城市也要创建对应的fragment页面* **fragmentList**.clear();  
 initPager();  
 **adapter**.notifyDataSetChanged();  
 *//页面数量发生改变，指示器的数量也会发生变化，重新设置添加指示器* **imgList**.clear();  
 **pointLayout**.removeAllViews(); *//将布局当中所有的元素全部移除* initPoint();  
 **mainVp**.setCurrentItem(**fragmentList**.size()-1);  
 }  
}

#### 3.2.2 添加城市界面

添加城市设计更加简洁且方便操作，在搜索框下面有几排热门城市的显示，如果热门城市中有所需的城市直接点击就可以添加，如果没有则需要在搜索框中搜索。



图 3-2-2 添加城市界面

界面设计分两个部分

1. 顶部由一个ImageView与一个EditText控件做构成。
2. 下面的热门城市则是使用GridView控件。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***SearchCityActivity.java***

**public class** SearchCityActivity **extends** BaseActivity **implements** View.OnClickListener {  
 EditText **searchEt**;  
 ImageView **submitIv**;  
 GridView **searchGv**;  
 String[]**hotCitys** = {**"北京"**,**"上海"**,**"广州"**,**"深圳"**,**"珠海"**,**"佛山"**,**"南京"**,**"苏州"**,**"厦门"**,**"长沙"**,**"成都"**,**"福州"**,  
 **"杭州"**,**"武汉"**,**"青岛"**,**"西安"**,**"太原"**,**"沈阳"**,**"重庆"**,**"天津"**,**"南宁"**};  
 **private** ArrayAdapter<String> **adapter**;  
 String **url1**=**"http://api.map.baidu.com/telematics/v3/weather?location="**;  
 String **url2**=**"&output=json&ak=FkPhtMBK0HTIQNh7gG4cNUttSTyr0nzo"**;  
 String **city**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_search\_city***);  
 **searchEt** = findViewById(R.id.***search\_et***);  
 **submitIv** = findViewById(R.id.***search\_iv\_submit***);  
 **searchGv** = findViewById(R.id.***search\_gv***);  
 **submitIv**.setOnClickListener(**this**);  
  
 *//设置适配器* **adapter** = **new** ArrayAdapter<>(**this**, R.layout.***item\_hotcity***, **hotCitys**);  
 **searchGv**.setAdapter(**adapter**);  
 setListener();  
 }  
*/\*设置监听事件\*/* **private void** setListener() {  
 **searchGv**.setOnItemClickListener(**new** AdapterView.OnItemClickListener() {  
 @Override  
 **public void** onItemClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 **city** = **hotCitys**[position];  
 String url = **url1**+**city**+**url2**;  
 loadData(url);  
 }  
 });  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **switch** (v.getId()) {  
 **case** R.id.***search\_iv\_submit***:  
 **city** = **searchEt**.getText().toString();  
 **if** (!TextUtils.*isEmpty*(**city**)) {  
 *//判断时候能找到这个城市* String url = **url1**+**city**+**url2**;  
 loadData(url);  
 }**else**{  
 Toast.*makeText*(**this**,**"输入内容不能为空"**,Toast.***LENGTH\_SHORT***).show();  
 }  
 **break**;  
 }  
 }  
 @Override  
 **public void** onSuccess(String result) {  
 WeatherBean weatherBean = **new** Gson().fromJson(result, WeatherBean.**class**);  
 **if** (weatherBean.getError()==0) {  
 Intent intent = **new** Intent(**this**, MainActivity.**class**);  
 intent.setFlags(Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***|Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 intent.putExtra(**"city"**,**city**);  
 startActivity(intent);  
 }**else**{  
 Toast.*makeText*(**this**,**"暂时未收入此城市信息"**,Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
}

#### 3.2.3 删除城市界面

删除界面操作简介方便，点击列表中的删除图标，即可删除需要删除的城市项。 左上角的“×”操作是可以撤销刚才所作的删除操作，点击右上角的向下按钮，即可返回城市管理界面。



图 3-2-1 删除城市界面

界面设计分三个部分

1. 第一部分：顶部是RelativeLayout（相对布局），左边是一个ImageView控件的“×”。
2. 第二部分：下面是一个ListView控件，用来显示自己已经添加过的城市列表。
3. 第三部分：一个城市列表项中由左边的删除，右边的TextView组成。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***DeleteCityActivity.java***

**public class** DeleteCityActivity **extends** AppCompatActivity **implements** View.OnClickListener {  
 ImageView **errorIv**,**rightIv**;  
 ListView **deleteLv**;  
 List<String>**mDatas**; *//ListView的数据源* List<String>**deleteCitys**; *//表示存储了删除的城市信息* **private** DeleteCityAdapter **adapter**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_delete\_city***);  
 **errorIv**=findViewById(R.id.***delete\_iv\_error***);  
 **rightIv**=findViewById(R.id.***delete\_iv\_right***);  
 **deleteLv**=findViewById(R.id.***delete\_lv***);  
 **mDatas** = DBManager.*queryALLCityNmae*();  
 **deleteCitys** = **new** ArrayList<>();  
 *//设置监听事件* **errorIv**.setOnClickListener(**this**);  
 **rightIv**.setOnClickListener(**this**);  
 *//适配器的设置* **adapter** = **new** DeleteCityAdapter(**this**, **mDatas**, **deleteCitys**);  
 **deleteLv**.setAdapter(**adapter**);  
  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **switch** (v.getId()) {  
 **case** R.id.***delete\_iv\_error***:  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(**this**);  
 builder.setTitle(**"提示信息"**).setMessage(**"您确定舍弃更改吗？"**).setPositiveButton(**"舍弃更改"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 finish(); *//关闭当前的Activity* }  
 });  
 builder.setNegativeButton(**"取消"**,**null**);  
 builder.create().show();  
 **break**;  
 **case** R.id.***delete\_iv\_right***:  
 **for**( **int** i=0;i<**deleteCitys**.size();i++){  
 String city = **deleteCitys**.get(i);  
 *//调用删除城市的函数* DBManager.*deleteInfoByCity*(city);  
 }  
 *//删除成功后返回上一级页面* finish();  
 **break**;  
 }  
 }

#### 3.2.4 更多设置界面

更多设置界面相对来说很普通了，共有四个选项：第一个是改变壁纸，当点击“改变壁纸”的时候，在他的下面就会弹出三个可供选择的壁纸。第二个是版本号，显示当前App的版本。第三个是清楚缓存。第四个是分享软件。

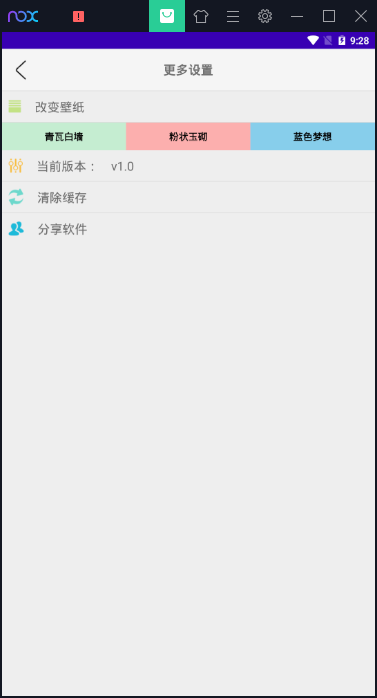


图 3-2-4 更多设置界面

图 3-2-1 删除城市界面

界面设计分两个部分

1. 第一部分：顶部是RelativeLayout（相对布局），用来返回前后一个界面。
2. 第二部分：在RelativeLayout下面由4个ImageView用来显示4个功能。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***MoreAcitvity.java***

**public class** MoreActivity **extends** AppCompatActivity **implements** View.OnClickListener {  
 TextView **bgTv**,**cacheTv**,**versionTv**,**shareTv**;  
 RadioGroup **exbgRg**;  
 ImageView **backIv**;  
 **private** SharedPreferences **pref**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_more***);  
 **bgTv**=findViewById(R.id.***more\_tv\_exchangebg***);  
 **cacheTv**=findViewById(R.id.***more\_tv\_cache***);  
 **versionTv**=findViewById(R.id.***more\_tv\_version***);  
 **shareTv**=findViewById(R.id.***more\_tv\_share***);  
 **backIv**=findViewById(R.id.***more\_iv\_back***);  
 **exbgRg**=findViewById(R.id.***more\_rg***);  
 *//设置监听* **bgTv**.setOnClickListener(**this**);  
 **cacheTv**.setOnClickListener(**this**);  
 **shareTv**.setOnClickListener(**this**);  
 **backIv**.setOnClickListener(**this**);  
  
  
 **pref** = getSharedPreferences(**"bg\_pref"**, ***MODE\_PRIVATE***);  
  
 String versionName = getVersionName();  
 **versionTv**.setText(**"当前版本： v"**+versionName);  
 setRGListener();  
 }  
  
 **private void** setRGListener() {  
 *//设置该变背景图片的单选按钮的监听* **exbgRg**.setOnCheckedChangeListener(**new** RadioGroup.OnCheckedChangeListener() {  
 @Override  
 **public void** onCheckedChanged(RadioGroup group, **int** checkedId) {  
 *//获取目前的默认壁纸* **int** bg = **pref**.getInt(**"bg"**,0);  
 SharedPreferences.Editor editor = **pref**.edit();  
 Intent intent = **new** Intent(MoreActivity.**this**,MainActivity.**class**);  
 intent.setFlags(Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***|Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 **switch** (checkedId) {  
 **case** R.id.***more\_rb\_green***:  
 **if** (bg==0) {  
 Toast.*makeText*(MoreActivity.**this**,**"您选择的是当前背景"**,Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 }  
 editor.putInt(**"bg"**,0);  
 editor.commit();  
 **break**;  
 **case** R.id.***more\_rb\_pink***:  
 **if** (bg==1) {  
 Toast.*makeText*(MoreActivity.**this**,**"您选择的是当前背景"**,Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 }  
 editor.putInt(**"bg"**,1);  
 editor.commit();  
 **break**;  
 **case** R.id.***more\_rb\_blue***:  
 **if** (bg==2) {  
 Toast.*makeText*(MoreActivity.**this**,**"您选择的是当前背景"**,Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 }  
 editor.putInt(**"bg"**,2);  
 editor.commit();  
 **break**;  
 }  
 startActivity(intent);  
 }  
 });  
 }  
  
 **private** String getVersionName() {  
 */\*获取应用的版本名称\*/* PackageManager manager = getPackageManager();  
 String versionName =**null**;  
 **try** {  
 PackageInfo info = manager.getPackageInfo(getPackageName(),0);  
 versionName = info.**versionName**;  
 } **catch** (PackageManager.NameNotFoundException e) {  
 e.printStackTrace();  
 }  
 **return** versionName;  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **switch** (v.getId()) {  
 **case** R.id.***more\_iv\_back***:  
 finish();  
 **break**;  
 **case** R.id.***more\_tv\_cache***:  
 clearCache();  
 **break**;  
 **case** R.id.***more\_tv\_share***:  
 shareSoftwareMsg(**"看天气App值得你的下载！"**);  
 **break**;  
 **case** R.id.***more\_tv\_exchangebg***:  
 **if** (**exbgRg**.getVisibility() == View.***VISIBLE***) {  
 **exbgRg**.setVisibility(View.***GONE***);  
 }**else**{  
 **exbgRg**.setVisibility(View.***VISIBLE***);  
 }  
 **break**;  
 }  
 }  
  
 **private void** shareSoftwareMsg(String s) {  
 */\*这是一个分享软件的函数\*/* Intent intent = **new** Intent(Intent.***ACTION\_SEND***);  
 intent.setType(**"text/plain"**); *//文本类型分享* intent.putExtra(Intent.***EXTRA\_TEXT***,s);  
 startActivity(Intent.*createChooser*(intent,**"看天气"**));  
  
 }  
  
 **private void** clearCache() {  
 */\*清除缓存函数\*/* AlertDialog.Builder builder = **new** AlertDialog.Builder(**this**);  
 builder.setTitle(**"提示信息"**).setMessage(**"确定要删除缓存吗？"**);  
 builder.setPositiveButton(**"确定"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 DBManager.*deleteALLInfo*();  
 Toast.*makeText*(MoreActivity.**this**,**"已清除全部缓存！"**,Toast.***LENGTH\_SHORT***).show();  
 Intent intent = **new** Intent(MoreActivity.**this**,MainActivity.**class**);  
 intent.setFlags(Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***|Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 startActivity(intent);  
  
 }  
 }).setNegativeButton(**"取消"**,**null**);  
 builder.create().show(); *//清空所有数据库操作* }  
}

### 3.3 数据库存储设计

创建一个forecast.db的数据库，并创建一个名为info的表用来存储数据信息。

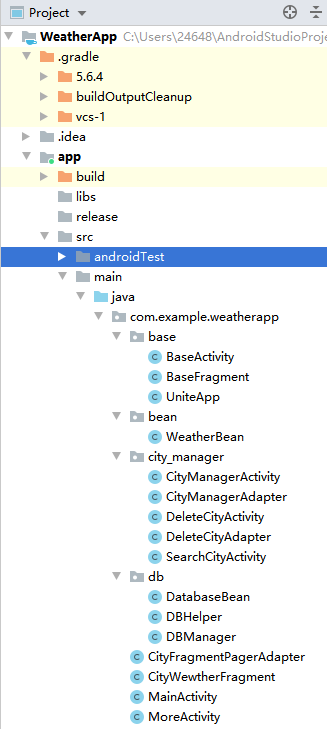
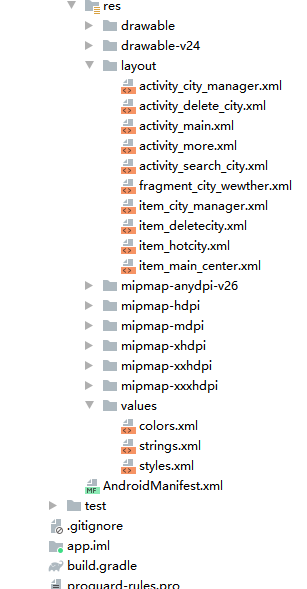
**public class** DBHelper **extends** SQLiteOpenHelper {  
 **public** DBHelper(Context context){  
 **super**(context,**"forecast.db"**,**null**,1);  
 }  
 @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 *//数据库第一次被创建的方法  
 //创建表的操作* String sql=**"create table info (\_id integer primary key autoincrement,city varchar(20) unique not null,content text not null)"**;  
 db.execSQL(sql);  
 }  
 @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {}  
}

创建一个DBManager.java用来对数据库中的数据进行操作。

**public class** DBManager {  
 **public static** SQLiteDatabase *database*;  
  
 *//初始化数据库信息* **public static void** initDB(Context context) {  
 DBHelper dbHelper = **new** DBHelper(context);  
 *database* = dbHelper.getWritableDatabase();  
 }  
 *//查找数据库当中的城市列表* **public static** List<String>queryALLCityNmae(){  
 Cursor cursor = *database*.query(**"info"**, **null**, **null**, **null**, **null**, **null**,**null**);  
 List<String>cityList=**new** ArrayList<>();  
 **while** (cursor.moveToNext()) {  
 String city = cursor.getString(cursor.getColumnIndex(**"city"**));  
 cityList.add(city);  
 }  
 **return** cityList;  
 }  
 */\*根据城市名称替换信息内容\*/* **public static int** updateInfoByCity(String city,String content) {  
 ContentValues values = **new** ContentValues();  
 values.put(**"content"**,content);  
 **return** *database*.update(**"info"**,values,**"city=?"**,**new** String[]{city});  
 }  
 */\*新增一条城市记录\*/* **public static long** addCityInfo(String city,String content){  
 ContentValues values = **new** ContentValues();  
 values.put(**"city"**,city);  
 values.put(**"content"**,content);  
 **return** *database*.insert(**"info"**,**null**,values);  
 }  
 */\*根据城市名字查询数据库当中的内容\*/* **public static** String queryInfoByCity(String city){  
 Cursor cursor = *database*.query(**"info"**,**null**,**"city=?"**,**new** String[]{city},**null**,**null**,**null**);  
 **if** (cursor.getCount()>0) {  
 cursor.moveToFirst();  
 String content = cursor.getString(cursor.getColumnIndex(**"content"**));  
 **return** content;  
 }  
 **return null**;  
 }  
 *//存储城市天气，最多存储5个 超过5个 就不能存储  
 //定义一个方法 获取现在存储的数量* **public static int** getCityCount() {  
 Cursor cursor = *database*.query(**"info"**,**null**,**null**,**null**,**null**,**null**,**null**);  
 **int** count = cursor.getCount();  
 **return** count;  
 }  
  
 */\*查寻数据库当中的全部信息\*/* **public static** List<DatabaseBean>queryALLInfo(){  
 Cursor cursor = *database*.query(**"info"**,**null**,**null**,**null**,**null**,**null**,**null**);  
 List<DatabaseBean>list = **new** ArrayList<>();  
 **while** (cursor.moveToNext()) {  
 **int** id = cursor.getInt(cursor.getColumnIndex(**"\_id"**));  
 String city = cursor.getString(cursor.getColumnIndex(**"city"**));  
 String content = cursor.getString(cursor.getColumnIndex(**"content"**));  
 DatabaseBean bean = **new** DatabaseBean(id,city,content);  
 list.add(bean);  
 }  
 **return** list;  
 }  
 */\*根据城市名称，删除这个城市在数据库中的信息\*/* **public static int** deleteInfoByCity(String city ){  
 **return** *database*.delete(**"info"**,**"city=?"**,**new** String[]{city});  
 }  
 */\*删除表当中所有的数据信息\*/* **public static void** deleteALLInfo(){  
 String sql=**"delete from info"**;  
 *database*.execSQL(sql);  
 }  
}

## 4 系统编码实现

### 4.1 项目框架

### 4.2 交互实现

由于布局文件代码众多，再此只选择一部分显示：

***activity\_main.xml***

<**RelativeLayout  
 android:id="@+id/main\_bottom\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:paddingBottom="15dp"  
 android:layout\_alignParentBottom="true"**>

<**ImageView  
 android:id="@+id/main\_iv\_add"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"  
 android:layout\_marginLeft="10dp"  
 android:src="@mipmap/icon\_add"**/>  
  
 <**ImageView  
 android:id="@+id/main\_iv\_more"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginRight="10dp"  
 android:src="@mipmap/icon\_more"** />  
  
 <**LinearLayout  
 android:id="@+id/main\_layout\_point"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerInParent="true"  
 android:orientation="horizontal"** />  
</**RelativeLayout**>

<**androidx.viewpager.widget.ViewPager  
 android:id="@+id/main\_vp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_above="@+id/main\_bottom\_layout"**>  
</**androidx.viewpager.widget.ViewPager**>

***activity\_search\_city.xml***

<**ImageView  
 android:id="@+id/search\_iv\_submit"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginLeft="5dp"  
 android:src="@mipmap/icon\_search"** />  
  
<**EditText  
 android:id="@+id/search\_et"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_toLeftOf="@+id/search\_iv\_submit"  
 android:background="@drawable/searchbg"  
 android:hint="搜索国内城市"  
 android:padding="3dp"** />

<**TextView  
 android:id="@+id/search\_tv"  
 android:layout\_below="@id/search\_et"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="20dp"  
 android:textSize="16dp"  
 android:textStyle="bold"  
 android:text="热门城市"**/>

<**GridView  
 android:id="@+id/search\_gv"  
 android:layout\_below="@id/search\_tv"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:numColumns="5"**/>

***activity\_delete\_city.xml***

<**RelativeLayout  
 android:id="@+id/delete\_top\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="60dp"  
 android:background="#F5F5F5"**>

<**ImageView  
 android:id="@+id/delete\_iv\_error"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"  
 android:src="@mipmap/icon\_error"  
 android:layout\_marginLeft="10dp"**/>

<**ImageView  
 android:id="@+id/delete\_iv\_right"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"  
 android:src="@mipmap/icon\_right"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginRight="10dp"**/>

<**TextView  
 android:id="@+id/delete\_tv\_title"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="城市管理"  
 android:textSize="18sp"  
 android:textStyle="bold"  
 android:layout\_centerInParent="true"**/>  
</**RelativeLayout**>  
*<!-- 分割线-->*<**ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="1dp"  
 android:background="#DCDCDC"**/>  
<**ListView  
 android:id="@+id/delete\_lv"  
 android:divider="@null"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**></**ListView**>

### 4.3 单元测试

设备检测：OPPO R9

API检测：没有问题

功能测试：每个功能都能实现，达到预期目标

## 5 系统测试发布

手机实测图：



图5.1 图5.2



图5.3 图5.4

# 项目二、理财App

## 背景意义

随着移动终端的迅速普及，Android系统平台引用软件的需求随之增大。伴随着Android智能手机与平板电脑已经出现在我们生活的大量的使用，越来越多的基于Android开发平台也随之而出，为丰富人们使用Android智能产品的用途，使其可以帮人们记录一些事情。本设计开发通过研究Android体系结构和个人理财管理方面的知识，设计并实现了个人理财通系统。能够对理财信息进行获取、汇总、整理、计算等功能，从而实现随身随时随地地进行日常的理财活动。

## 1 项目需求分析

### 1.1 需求分析

物质和科技的飞速发展，人们的生活水平也不断的在提高，往往有很多人在快节奏的生活中迷失和迷茫，很多人觉得自己没钱，但每个月的工资也不是很低，却往往不知道钱花在哪，为什么每到月底自己的钱包会空空如也，正因为这样，人们才需要一款个人理财软件，简单的界面，易懂的操作，十分便携直观的理财方式，可以让人们更好的进行个人理财。以下是本软件的一些功能：

1. 登录界面：输入账户密码验证登录
2. 注册界面：添加账号密码以及相关信息
3. 新增支出：添加支出金额、时间、类别和地点等信息
4. 支出明细：显示支出金额、时间、类别和付款方等信息
5. 新增收入：添加收入金额、时间、类别和付款方等信息
6. 收入明细：显示收入金额、时间、类别和付款方等信息
7. 数据分析：支出汇总，收入汇总
8. 系统设置：可以修改密码

### 1.2 功能需求

目前国内外理财软件已有上百种之多，如美国的直觉公司QUICKEN软件为美国13个州及加拿大的客户提供金融管理和预算等财务问题。国内在财务管理方面做的比较突出的当属金蝶公司。然而，在手机理财软件方面做的很突出的还没有，本软件是针对个人用户的一款Android软件，主要对个人理财收入、支出做一个记录和统计，可以对用户的收入、支出记录做添加、删除、查询和修改的管理，本软件该具备以下功能：

①功能操作要方便、易懂，不要有多余或复杂的操作。

②对用户收入支出信息做添加、删除、查询和修改。

③系统的功能复合本人的实际情况。

④界面直观简洁，方便用户更好的使用。

⑤保证软件的安全性。

## 2 总体设计

### 2.1系统架构设计

理财App

启动界面

注册界面

新增支出

新增收入

支出明细

收入明细

数据分析

系统设置

登陆界面

### 2.2系统功能体系

系统总体功能主要包括系统各功能模块的详细划分和定义，以及各子功能模块具体实现细节，通过详细划分和定义细节可以将系统各功能进行有效区分，防止功能重叠，功能冗余等现象，系统总体功能分析如表2-1所示.

|  |  |
| --- | --- |
| **用户功能** | **具体描述** |
| 注册界面 | 输入账号密码邮件电话等信息注册 |
| 登录界面 | 输入账号密码登录 |
| 新增支出 | 用户对新增支出财产进行详细的记录 |
| 新增收入 | 用户对新增收入财产进行详细的记录 |
| 支出明细 | 用户对我的支出财产记录进行详细的查询 |
| 收入明细 | 用户对我的收入财产进行详细的查询 |
| 数据分析 | 支出汇总，收入汇总 |
| 系统设置 | 修改登陆密码 |

表2-2 总体功能分析

## 3 系统详细设计

### 3.1 数据库设计

个人理财通是一款运行在Android系统上的程序，在Android系统中，集成了一种轻量型的数据库，即SQLite，该数据库是使用C语言编写的开源嵌入式数据库，支持的数据库大小为2TB，使用该数据库，用户可以像使用SQL Server数据库或者Oracle数据库那样来存储、管理和维护数据，本系统采用了SQLite数据库，并且命名为financial.db，，该数据库中用到了3个数据表，分别是tb\_userinfo、in\_come和pay\_out。

表3-1-1 tb\_userinfo

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 大小 | 是否主键 | 是否为空 | 备注 |
| id | int | autoincrement | 是 | 否 | 序号 |
| name | varchar | 10 | 否 | 否 | 姓名 |
| pwd | varchar | 15 | 否 | 否 | 密码 |
| email | varchar | 50 | 否 | 否 | 邮件 |
| phone | varchar | 11 | 否 | 否 | 电话 |

表3-1-2 in\_come

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 大小 | 是否主键 | 是否为空 | 备注 |
| id | int | autoincrement | 是 | 否 | 序号 |
| inmoney | double |  | 否 | 否 | 收入钱数 |
| intime | varchar | 20 | 否 | 否 | 进账时间 |
| intype | varchar | 30 | 否 | 否 | 收入的类型 |
| inpayer | varchar | 100 | 否 | 否 | 谁支付给我 |
| inremark | varchar | 500 | 否 | 否 | 备注 |

表3-1-3 pay\_out

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 类型 | 大小 | 是否主键 | 是否为空 | 备注 |
| id | int | autoincrement | 是 | 否 | 序号 |
| outmoney | double |  | 否 | 否 | 支出钱数 |
| outtime | varchar | 20 | 否 | 否 | 支出时间 |
| outtype | varchar | 30 | 否 | 否 | 支出的类型 |
| outpayee | varchar | 100 | 否 | 否 | 我支付给谁 |
| outremark | varchar | 500 | 否 | 否 | 备注 |

### 3.2系统界面设计

#### 3.2.1 启动界面

启动界面：清新简洁的启动界面，标识着各种消费示意图，提升整体软件美观程度，点击下方“知道了”进入主程序。



图3-2-1

界面设计分两个部分

1. 第一部分：上面是一个图片，使用的是ImageView控件
2. 第二部分：下面是三句话，其中“知道了”是一个Button按钮，点击进入主程序

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***WelcomeActivity.java***

**public class** WelcomeActivity **extends** AppCompatActivity {  
*//定义对象* Button **bt\_know\_wel**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_welcome***);  
 *//绑定控件* **bt\_know\_wel**=findViewById(R.id.***bt\_know\_wel***);  
 *//按钮单击事件* **bt\_know\_wel**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent intent=**new** Intent(WelcomeActivity.**this**,LoginActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }  
}

#### 3.2.2 注册界面

注册界面清洁简单，输入用户名等信息点击注册即可注册成功，主要是讲各个控件中的信息保存到了数据库中。



图3-2-2

界面设计分六个部分

1. 第一至四部分：是4个LinearLayout布局，分别是“用户名”，“密码”，“邮箱”，“电话”。
2. 第五至六部分： 是两个Button按钮，分别是注册，取消。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***RegisterActivity.java***

**public class** RegisterActivity **extends** AppCompatActivity {  
*//3定义对象* EditText **et\_name**,**et\_pwd**,**et\_email**,**et\_phone**;  
 Button **btn\_register**,**btn\_cancel**;  
 MyDBHelper **mhelper**;*//创建一个数据库类文件* SQLiteDatabase **db**;*//创建一个可以操作的数据库对象* @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_register***);  
 *//4 绑定控件* initView();

*//5 注册按钮功能的实现* btnRegister();

*//6 取消按钮功能的实现* btnCancel();  
 }  
 *//4 绑定控件--------代码* **private void** initView() {  
 **et\_name**=findViewById(R.id.***et\_name\_rg***);  
 **et\_pwd**=findViewById(R.id.***et\_pwd\_rg***);  
 **et\_email**=findViewById(R.id.***et\_email\_rg***);  
 **et\_phone**=findViewById(R.id.***et\_phone\_rg***);  
 **btn\_register**=findViewById(R.id.***bt\_ok\_rg***);  
 **btn\_cancel**=findViewById(R.id.***bt\_cancel\_rg***);  
 **mhelper**=**new** MyDBHelper(RegisterActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();  
 }

*//5 注册按钮功能的实现--------------------代码* **private void** btnRegister() {  
 **btn\_register**.setOnClickListener(**new** View.OnClickListener() {

@Override  
 **public void** onClick(View v) {  
 *//创建一个对象，用来封装一行数据* ContentValues values=**new** ContentValues();  
 values.put(**"name"**,**et\_name**.getText().toString());*//将输入的用户名放到name列*

values.put(**"pwd"**,**et\_pwd**.getText().toString());*//将输入的密码放到pwd列* values.put(**"email"**,**et\_email**.getText().toString());*//将输入的邮箱放到email列* values.put(**"phone"**,**et\_phone**.getText().toString());*//将输入的电话放到phone列  
 //将封装好的一行数据保存到数据库的tb\_userinfo表中* **db**.insert(**"tb\_userinfo"**,**null**,values);  
 Toast.*makeText*(RegisterActivity.**this**,**"注册成功"**,Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 }

*//6 取消按钮功能的实现-------------------代码* **private void** btnCancel() {  
 **btn\_cancel**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent intent=**new** Intent(RegisterActivity.**this**, LoginActivity.**class**);

startActivity(intent);  
 finish();

}  
 });  
 }

}

#### 3.2.3 登录界面

登陆界面背景简介大方，输入刚注册好的账号密码点击登录即可进入主程序，主要是将在控件中输入的信息去到数据库中进行比对，正确则进入主程序。



图3-2-3

界面设计分四个部分

1. 第一至二部分：是2个LinearLayout布局，分别是用户名，密码。
2. 第三至四部分： 是两个Button按钮，分别新用户注册，取消。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***LoginActivity.java***

**public class** LoginActivity **extends** AppCompatActivity {  
*//1 定义对象* EditText **et\_name**,**et\_pwd**;  
 Button **btn\_newregister**,**btn\_login**;  
 MyDBHelper **mhelper**;  
 SQLiteDatabase **db**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_login***);  
 *//2 绑定控件* initView();  
  
 *//3 登录按钮功能的实现* btnLogin();  
  
 *//4 新用户注册按钮功能的实现* btnNewRegister();  
 }  
  
 *//2 绑定控件--------------------代码* **private void** initView() {  
 **et\_name**=findViewById(R.id.***et\_name\_lg***);  
 **et\_pwd**=findViewById(R.id.***et\_pwd\_lg***);  
 **btn\_newregister**=findViewById(R.id.***bt\_newregister\_lg***);  
 **btn\_login**=findViewById(R.id.***bt\_login\_lg***);  
 **mhelper**=**new** MyDBHelper(LoginActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();  
 }  
  
  
 *//3 登录按钮功能的实现---------------------------代码* **private void** btnLogin() {  
 **btn\_login**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 *//首先：获取输入的用户名和密码* String inputname=**et\_name**.getText().toString();  
 String inputpwd=**et\_pwd**.getText().toString();  
 *//其次：对获取的用户名和密码进行判断* **if**(inputname.equals(**""**)||inputpwd.equals(**""**)){*//用户名或密码为空* Toast.*makeText*(LoginActivity.**this**,**"用户名或密码不能为空"**,Toast.***LENGTH\_SHORT***).show();  
 }**else**{*//用户名或密码不为空时，我们再对输入的正确性进行判断。  
 // 根据输入的用户名和密码从数据库中查询* Cursor cursor =**db**.rawQuery(**"select \* from tb\_userinfo where name=? and pwd=?"**,**new** String[]{inputname,inputpwd});  
 *//根据查询到的结果进行判断* **if** (cursor.moveToNext()){*//查询到时* String getname=cursor.getString(cursor.getColumnIndex(**"name"**));  
 String getpwd=cursor.getString(cursor.getColumnIndex(**"pwd"**));  
 **if**(inputname.equalsIgnoreCase(getname)&&inputpwd.equalsIgnoreCase(getpwd)){  
 SharedPreferences.Editor editor=getSharedPreferences(**"userinfo"**,0).edit();  
 editor.putString(**"username"**,inputname);  
 editor.putString(**"userpwd"**,inputpwd);  
 editor.commit();  
 Toast.*makeText*(LoginActivity.**this**,**"用户名和密码正确，欢迎登陆"**,Toast.***LENGTH\_SHORT***).show();  
 Intent intent=**new** Intent(LoginActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 }**else**{*//没有查询到结果时* Toast.*makeText*(LoginActivity.**this**,**"用户名或密码错误，请重新输入"**,Toast.***LENGTH\_SHORT***).show();  
 **et\_name**.setText(**""**);  
 **et\_pwd**.setText(**""**);  
 }  
 }  
 }  
 });  
  
 }  
  
 *//4 新用户注册按钮功能的实现------------------------代码* **private void** btnNewRegister() {  
 **btn\_newregister**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent intent=**new** Intent(LoginActivity.**this**, RegisterActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }  
}

#### 3.2.4 新增收入界面

新增收入界面背景简介大方。主要由一下几个控件，分别是金额，日期，类型，付款方，备注。在其中输入相对应的信息即可保存。



图3-2-4

界面设计分七个部分

1. 第一至五部分：是5个LinearLayout布局，分别是金额，日期，类型，付款方，备注。
2. 第六至七部分： 是两个Button按钮，分别保存，取消。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***NewInComeActivity.java***

**public class** NewInComeActivity **extends** AppCompatActivity {  
*//1 定义对象* EditText **et\_money**,**et\_time**,**et\_payer**,**et\_remark**;  
 Spinner **sp\_type**;  
 Button **bt\_sava**,**bt\_cancel**;  
 MyDBHelper **mhelper**;  
 SQLiteDatabase **db**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_new\_in\_come***);  
 *//2 绑定控件* initView();

*//3保存按钮功能的实现* btnSave();

*//4 取消按钮功能的实现* btnCancel();

}

*//2 绑定控件-------------------代码* **private void** initView() {  
 **et\_money**=findViewById(R.id.***et\_money\_newin***);  
 **et\_time**=findViewById(R.id.***et\_time\_newin***);  
 **sp\_type**=findViewById(R.id.***sp\_type\_newin***);  
 **et\_payer**=findViewById(R.id.***et\_payer\_newin***);  
 **et\_remark**=findViewById(R.id.***et\_remark\_newin***);  
 **bt\_sava**=findViewById(R.id.***bt\_save\_newin***);  
 **bt\_cancel**=findViewById(R.id.***bt\_cancel\_newin***);  
 **mhelper**=**new** MyDBHelper(NewInComeActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();

}  
 *//3保存按钮功能的实现--------代码* **private void** btnSave() {  
 **bt\_sava**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {

*//获取输入的内容保存到数据库的收入表中* ContentValues values=**new** ContentValues();  
 values.put(**"inmoney"**,**et\_money**.getText().toString());  
 values.put(**"intime"**,**et\_time**.getText().toString());  
 values.put(**"intype"**,**sp\_type**.getSelectedItem().toString());  
 values.put(**"inpayer"**,**et\_payer**.getText().toString());  
 values.put(**"inremark"**,**et\_remark**.getText().toString());  
 **db**.insert(**"in\_come"**,**null**,values);  
 Toast.*makeText*(NewInComeActivity.**this**,**"保存成功"**,Toast.***LENGTH\_SHORT***).show();

*//刷新本页面* Intent intent=**new** Intent(NewInComeActivity.**this**,NewInComeActivity.**class**);  
 startActivity(intent);  
 finish();

}  
 });  
 }

*//4 取消按钮功能的实现--------代码* **private void** btnCancel() {  
 **bt\_cancel**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {

Intent intent=**new** Intent(NewInComeActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 finish();

}

});

}  
}

#### 3.2.5 收入明细界面

收入明细界面给人的感觉就是要一眼了解收入多少钱，在做放还有收入的时间，付款方，类型等信息。主要是通过一个RecyclerView控件从数据库中读取信息实现。

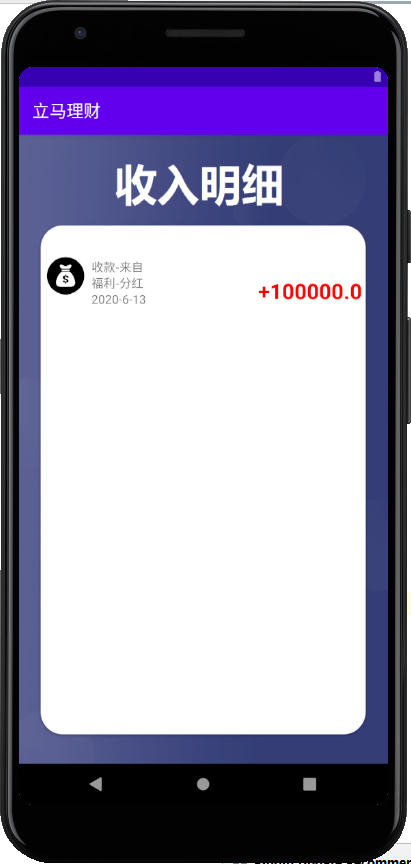


图3-2-5

界面设计分一个部分

1. 在中间的空白地方内部有个RecycleView控件，用于显示收入明细，功能实现方法是从数据库的in\_come表中读取数显示在此界面。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***InComeDetailActivity.java***

**public class** InComeDetailActivity **extends** AppCompatActivity {  
*//1 定义对象* RecyclerView **recy\_view**;  
 MyDBHelper **mhelper**;  
 SQLiteDatabase **db**;  
 List<IncomeBean> **arr1**=**new** ArrayList();  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_in\_come\_detail***);  
 *//2 绑定控件* initView();  
  
 *//3 准备数据* initData();  
  
 *//4 设计每一行的子布局  
  
 //5定义适配器： 数据和子布局关联起来（桥梁的作用）* IncomeAdapter adapter=**new** IncomeAdapter(InComeDetailActivity.**this**,**arr1**);  
  
 *//6将适配器和布局管理器加载到控件当中* StaggeredGridLayoutManager st=**new** StaggeredGridLayoutManager(StaggeredGridLayoutManager.***VERTICAL***,1);  
 **recy\_view**.setLayoutManager(st);  
 **recy\_view**.setAdapter(adapter);  
 }  
  
 *//2 绑定控件----------------代码* **private void** initView() {  
 **recy\_view**=findViewById(R.id.***recy\_view\_indetail***);  
 **mhelper**=**new** MyDBHelper(InComeDetailActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();  
 }  
  
 *//3 准备数据----------------代码* **private void** initData() {  
 *//从数据库查询所有的新增收入信息,取出数据* Cursor cursor=**db**.rawQuery(**"select \* from in\_come"**,**null**);  
 **while**(cursor.moveToNext()){  
 **int** myid=cursor.getInt(cursor.getColumnIndex(**"id"**));  
 **double** mymoney=cursor.getDouble(cursor.getColumnIndex(**"inmoney"**));  
 String mytime=cursor.getString(cursor.getColumnIndex(**"intime"**));  
 String mytype=cursor.getString(cursor.getColumnIndex(**"intype"**));  
 String mypayer=cursor.getString(cursor.getColumnIndex(**"inpayer"**));  
 String myremark=cursor.getString(cursor.getColumnIndex(**"inremark"**));  
 IncomeBean incomeBean=**new** IncomeBean( myid,mymoney,mytime,mytype,mypayer,myremark);  
 **arr1**.add(incomeBean);  
 }  
  
 }  
  
}

#### 3.2.6新增支出界面

支出界面简介大方有条理，能够详细的记录新增指出的数目及相关信息，以便于用户自己查看，界面主要是用了TextView，EditText控件来布局。如下图：



图3-2-6

界面设计分七个部分

1. 第一至五部分：是5个LinearLayout布局，分别是金额，日期，类型，付款方，备注。
2. 第六至七部分： 是两个Button按钮，分别保存，取消。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***NewPayActivity.java***

**public class** NewPayActivity **extends** AppCompatActivity {  
*//定义对象*EditText **et\_money**,**et\_time**,**et\_payer**,**et\_remark**;  
 Spinner **sp\_type**;  
 Button **bt\_sava**,**bt\_cancel**;  
 MyDBHelper **mhelper**;  
 SQLiteDatabase **db**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_new\_pay***);  
  
 *//2 绑定控件* initView();  
 *//3保存按钮功能的实现* btnSave();  
  
 *//4 取消按钮功能的实现* btnCancel();  
 }  
  
 *//2 绑定控件-------------------代码* **private void** initView() {  
 **et\_money**=findViewById(R.id.***et\_money\_newout***);  
 **et\_time**=findViewById(R.id.***et\_time\_newout***);  
 **sp\_type**=findViewById(R.id.***sp\_type\_newout***);  
 **et\_payer**=findViewById(R.id.***et\_payer\_newout***);  
 **et\_remark**=findViewById(R.id.***et\_remark\_newout***);  
 **bt\_sava**=findViewById(R.id.***bt\_save\_newout***);  
 **bt\_cancel**=findViewById(R.id.***bt\_cancel\_newout***);  
 **mhelper**=**new** MyDBHelper(NewPayActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();  
 }  
  
 *//3保存按钮功能的实现--------代码* **private void** btnSave() {  
 **bt\_sava**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 *//获取输入的内容保存到数据库的收入表中* ContentValues values=**new** ContentValues();  
 values.put(**"outmoney"**,**et\_money**.getText().toString());  
 values.put(**"outtime"**,**et\_time**.getText().toString());  
 values.put(**"outtype"**,**sp\_type**.getSelectedItem().toString());  
 values.put(**"outpayee"**,**et\_payer**.getText().toString());  
 values.put(**"outremark"**,**et\_remark**.getText().toString());  
 **db**.insert(**"pay\_out"**,**null**,values);

Toast.*makeText*(NewPayActivity.**this**,**"保存成功"**,Toast.***LENGTH\_SHORT***).show();  
 *//刷新本页面* Intent intent=**new** Intent(NewPayActivity.**this**,NewPayActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 });

}  
  
 *//4 取消按钮功能的实现--------代码* **private void** btnCancel() {  
 **bt\_cancel**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent intent=**new** Intent(NewPayActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 finish();  
 }

});  
 }  
  
  
}

#### 3.2.7支出明细界面

支出明细界面给人的感觉就是要一眼了解支出多少钱，在做放还有收入的时间，付款方，类型等信息。主要是通过一个RecyclerView控件从数据库中读取信息实现。

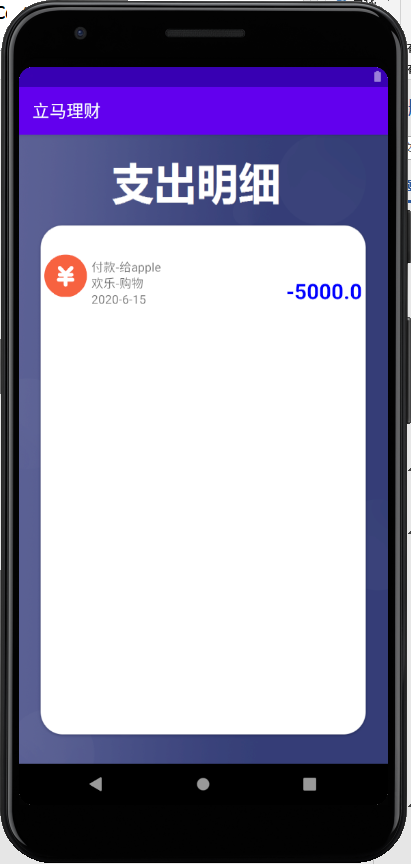


图3-2-7

界面设计分一个部分

1. 在中间的空白地方内部有个RecycleView控件，用于显示收入明细，功能实现方法是从数据库的in\_come表中读取数显示在此界面。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***PayDetailActivity.java***

**public class** PayDetailActivity **extends** AppCompatActivity {  
*//1定义对象*RecyclerView **recy\_view**;  
 MyDBHelper **mhelper**;  
 SQLiteDatabase **db**;  
 List<OutpayBean> **arr1**=**new** ArrayList();  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_pay\_detail***);  
  
 *//2 绑定控件* initView();  
  
 *//3 准备数据* initData();  
  
 *//4 设计每一行的子布局  
  
 //5 创建适配器* OutpayAdapter adapter=**new** OutpayAdapter(PayDetailActivity.**this**,**arr1**);  
  
 *//6将适配器和布局管理器加载到控件当中* StaggeredGridLayoutManager st=**new** StaggeredGridLayoutManager(StaggeredGridLayoutManager.***VERTICAL***,1);  
 **recy\_view**.setLayoutManager(st);  
 **recy\_view**.setAdapter(adapter);  
  
 }  
  
 *//2 绑定控件----------------代码* **private void** initView() {  
 **recy\_view**=findViewById(R.id.***recy\_view\_outdetail***);  
 **mhelper**=**new** MyDBHelper(PayDetailActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();  
 }  
  
 *//3 准备数据------------------代码* **private void** initData() {  
 *//从数据库查询所有的新增收入信息,取出数据* Cursor cursor=**db**.rawQuery(**"select \* from pay\_out"**,**null**);  
 **while**(cursor.moveToNext()){  
 **int** myid=cursor.getInt(cursor.getColumnIndex(**"id"**));  
 **double** mymoney=cursor.getDouble(cursor.getColumnIndex(**"outmoney"**));  
 String mytime=cursor.getString(cursor.getColumnIndex(**"outtime"**));  
 String mytype=cursor.getString(cursor.getColumnIndex(**"outtype"**));  
 String mypayer=cursor.getString(cursor.getColumnIndex(**"outpayee"**));  
 String myremark=cursor.getString(cursor.getColumnIndex(**"outremark"**));  
 OutpayBean outpayBean=**new** OutpayBean( myid,mymoney,mytime,mytype,mypayer,myremark);  
 **arr1**.add(outpayBean);  
 }  
  
 }  
  
}

#### 3.2.8数据分析界面

数据分析界面的主要功能是给用户显示自己当前的支出，收入新的汇总，使用折线图来表现出用户的收入支出情况。

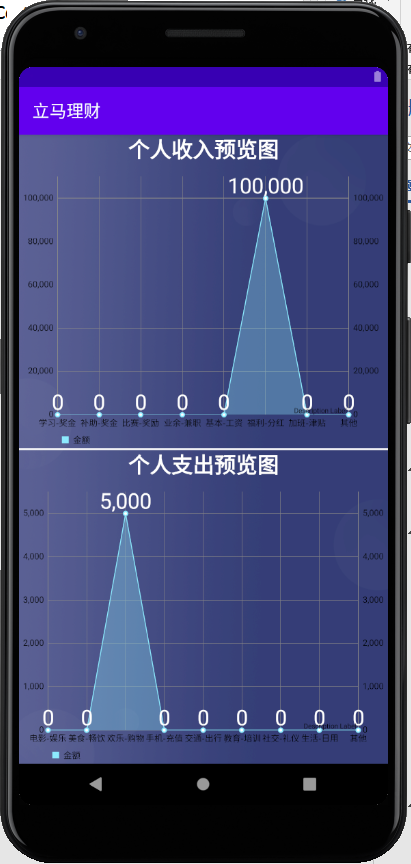


图3-2-8

界面设计分两个部分

1. 上面的一个部分是收入信息图，使用了LineChart控件进行页面布局，收据来源于数据库中的in\_come表。
2. 下面的一个部分是支出信息图，同样使用了LineChart控件进行页面布局，收据来源于数据库中的pay\_out表。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***DataAnalyseActivity.java***

**public class** DataAnalyseActivity **extends** AppCompatActivity {  
*//1 定义对象* LineChart **income\_chart**,**outpay\_chart**;  
 MyDBHelper **mhelper**;  
 SQLiteDatabase **db**;  
 String[] **indata**={**"学习-奖金"**,**"补助-奖金"**,**"比赛-奖励"**,**"业余-兼职"**,**"基本-工资"**,**"福利-分红"**,**"加班-津贴"**,**"其他"**};  
 *//收入类型数据统计的初始值* **int xxjjmoney**=0;  
 **int bzjjmoney**=0;  
 **int bsjlmoney**=0;  
 **int yyjzmoney**=0;  
 **int jbgzmoney**=0;  
 **int flfhmoney**=0;  
 **int jbjtmoney**=0;  
 **int qtmoney**=0;  
 String[] **outdata**={**"电影-娱乐"**,**"美食-畅饮"**,**"欢乐-购物"**,**"手机-充值"**,**"交通-出行"**,**"教育-培训"**,**"社交-礼仪"**,**"生活-日用"**,**"其他"**};  
 *//收入类型数据统计的初始值* **int dyylmoney**=0;  
 **int mscymoney**=0;  
 **int hlgwmoney**=0;  
 **int sjczmoney**=0;  
 **int jtcxmoney**=0;  
 **int jypxmoney**=0;  
 **int sjlymoney**=0;  
 **int shrymoney**=0;  
 **int othermoney**=0;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_data\_analyse***);  
 *//2 绑定控件* initView();  
  
 *//3 收入汇总分析* inComeData();  
  
 *//4 支出汇总分析* outComeData();  
  
 }  
 *//2 绑定控件---------------代码* **private void** initView() {  
 **income\_chart**=findViewById(R.id.***income\_chart\_data***);  
 **outpay\_chart**=findViewById(R.id.***outpay\_chart\_data***);  
 **mhelper**=**new** MyDBHelper(DataAnalyseActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();  
 }  
 *//3 收入汇总分析-------------------代码* **private void** inComeData() {  
 *//第一部分：获取数据* Cursor cursor =**db**.rawQuery(**"select \* from in\_come"**,**null**);  
 **while**(cursor.moveToNext()){  
 Double mymoney=cursor.getDouble(cursor.getColumnIndex(**"inmoney"**));  
 String mytype=cursor.getString(cursor.getColumnIndex(**"intype"**));  
 **if**(mytype.equals(**"学习-奖金"**)){  
 **xxjjmoney**+=mymoney;  
 }**else if**(mytype.equals(**"补助-奖金"**)){  
 **bzjjmoney**+=mymoney;  
 }**else if**(mytype.equals(**"比赛-奖励"**)){  
 **bzjjmoney**+=mymoney;  
 }**else if**(mytype.equals(**"业余-兼职"**)){  
 **yyjzmoney**+=mymoney;  
 }**else if**(mytype.equals(**"基本-工资"**)){  
 **jbgzmoney**+=mymoney;  
 }**else if**(mytype.equals(**"福利-分红"**)){  
 **flfhmoney**+=mymoney;  
 }**else if**(mytype.equals(**"加班-津贴"**)){  
 **jbjtmoney**+=mymoney;  
 }**else if**(mytype.equals(**"其他"**)){  
 **qtmoney**+=mymoney;  
 }  
 }  
 *//第二部分：LineChart图表初始化设置---Xy轴的设置* XAxis xAxis=**income\_chart**.getXAxis();*//获取此图表的x轴轴线* YAxis yAxisleft =**income\_chart**.getAxisLeft();*//获取此图表的Y轴左侧轴线* YAxis yAxisright =**income\_chart**.getAxisRight();*//获取此图表的Y轴右侧轴线* xAxis.setPosition(XAxis.XAxisPosition.***BOTTOM***);*//设置X轴线的位置为底部* yAxisleft.setAxisMinimum(0f);*//保证Y轴从0开始，不然会上移一点。* yAxisright.setAxisMinimum(0f);  
 xAxis.setValueFormatter(**new** IAxisValueFormatter() {*//x轴自定义标签的设置* @Override  
 **public** String getFormattedValue(**float** v, AxisBase axisBase) {  
 **return indata**[(**int**) v];  
 }  
 });  
 *//第三部分:LineDataSet曲线初始化设置* List<Entry> inentries=**new** ArrayList<>();*//Y轴的数据* inentries.add(**new** Entry(0,**xxjjmoney**));  
 inentries.add(**new** Entry(1,**bzjjmoney**));  
 inentries.add(**new** Entry(2,**bsjlmoney**));  
 inentries.add(**new** Entry(3,**yyjzmoney**));  
 inentries.add(**new** Entry(4,**jbgzmoney**));  
 inentries.add(**new** Entry(5,**flfhmoney**));  
 inentries.add(**new** Entry(6,**jbjtmoney**));  
 inentries.add(**new** Entry(7,**qtmoney**));  
 LineDataSet lineDataSet=**new** LineDataSet(inentries,**"金额"**);*//代表一条线,“金额”是曲线名称* lineDataSet.setValueTextSize(25);*//曲线上文字的大小* lineDataSet.setValueTextColor(Color.***WHITE***);*//曲线上文字的颜色* lineDataSet.setDrawFilled(**true**);*//设置折线图填充  
 //第四部分：曲线展示* LineData data=**new** LineData(lineDataSet);*//创建LineData对象 属于LineChart折线图的数据集合* **income\_chart**.setData(data);*// 添加到图表中* }  
 *//4 支出汇总分析--------------------代码* **private void** outComeData() {  
 *//第一部分：获取数据* Cursor cursor =**db**.rawQuery(**"select \* from pay\_out"**,**null**);  
 **while**(cursor.moveToNext()){  
 Double mymoney=cursor.getDouble(cursor.getColumnIndex(**"outmoney"**));  
 String mytype=cursor.getString(cursor.getColumnIndex(**"outtype"**));  
 **if**(mytype.equals(**"电影-娱乐"**)){  
 **dyylmoney**+=mymoney;  
 }**else if**(mytype.equals(**"美食-畅饮"**)){  
 **mscymoney**+=mymoney;  
 }**else if**(mytype.equals(**"欢乐-购物"**)){  
 **hlgwmoney**+=mymoney;  
 }**else if**(mytype.equals(**"手机-充值"**)){  
 **sjczmoney**+=mymoney;  
 }**else if**(mytype.equals(**"交通-出行"**)){  
 **jtcxmoney**+=mymoney;  
 }**else if**(mytype.equals(**"教育-培训"**)){  
 **jypxmoney**+=mymoney;  
 }**else if**(mytype.equals(**"社交-礼仪"**)){  
 **sjlymoney**+=mymoney;  
 }**else if**(mytype.equals(**"生活-日用"**)){  
 **shrymoney**+=mymoney;  
 }**else if**(mytype.equals(**"其他"**)){  
 **othermoney**+=mymoney;  
 }  
 }  
  
 *//第二部分：LineChart图表初始化设置---Xy轴的设置* XAxis xAxis=**outpay\_chart**.getXAxis();*//获取此图表的x轴轴线* YAxis yAxisleft =**outpay\_chart**.getAxisLeft();*//获取此图表的Y轴左侧轴线* YAxis yAxisright =**outpay\_chart**.getAxisRight();*//获取此图表的Y轴右侧轴线* xAxis.setPosition(XAxis.XAxisPosition.***BOTTOM***);*//设置X轴线的位置为底部* yAxisleft.setAxisMinimum(0f);*//保证Y轴从0开始，不然会上移一点。* yAxisright.setAxisMinimum(0f);  
 xAxis.setValueFormatter(**new** IAxisValueFormatter() {*//x轴自定义标签的设置* @Override  
 **public** String getFormattedValue(**float** v, AxisBase axisBase) {  
 **return outdata**[(**int**) v];  
 }  
 });  
  
 *//第三部分:LineDataSet曲线初始化设置* List<Entry> outentries=**new** ArrayList<>();*//Y轴的数据* outentries.add(**new** Entry(0,**dyylmoney**));  
 outentries.add(**new** Entry(1,**mscymoney**));  
 outentries.add(**new** Entry(2,**hlgwmoney**));  
 outentries.add(**new** Entry(3,**sjczmoney**));  
 outentries.add(**new** Entry(4,**jtcxmoney**));  
 outentries.add(**new** Entry(5,**jypxmoney**));  
 outentries.add(**new** Entry(6,**sjlymoney**));  
 outentries.add(**new** Entry(7,**shrymoney**));  
 outentries.add(**new** Entry(8,**othermoney**));  
 LineDataSet lineDataSet=**new** LineDataSet(outentries,**"金额"**);*//代表一条线,“金额”是曲线名称* lineDataSet.setValueTextSize(25);*//曲线上文字的大小* lineDataSet.setValueTextColor(Color.***WHITE***);*//曲线上文字的颜色* lineDataSet.setDrawFilled(**true**);*//设置折线图填充  
 //第四部分：曲线展示* LineData data=**new** LineData(lineDataSet);*//创建LineData对象 属于LineChart折线图的数据集合* **outpay\_chart**.setData(data);*// 添加到图表中* }  
  
}

#### 3.2.9系统设置界面



图3-2-9

界面设计分五个部分

1. 上面三个部分是采用LinearLayout包含TextView，EditText两个控件来显示需要输入的信息。
2. 下面两个部分是两个Button按钮，主要用来保存当前的修改和取消输入的信息。

##### 逻辑代码：（代码过于繁多，只展示一部分重要的逻辑代码）

***SysSettingActivity.java***

**public class** SysSettingActivity **extends** AppCompatActivity {  
 *//1 定义对象* TextView **txt\_user**;*//创建一个显示用户名的文本对象* EditText **et\_ypwd**, **et\_xpwd**, **et\_zxpwd**;*// 创建三个EditText对象* Button **bt\_modify**, **bt\_cancel**;*// 创建两个Button对象* MyDBHelper **mhelper**;  
 SQLiteDatabase **db**;  
 String **name**;  
 String **pwd**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_sys\_setting***);  
 *//第二步：绑定控件* initView();  
 *//第三步：显示当前登录的用户名* displayInfo();  
 *//第四步：修改按钮功能* btnModify();  
 *//第五步：取消按钮功能* btncancel();  
 }  
 *//第二步：绑定控件-----------------代码* **private void** initView() {  
 **txt\_user**=findViewById(R.id.***txt\_name\_sys***);  
 **et\_ypwd**=findViewById(R.id.***et\_ypwd\_sys***);  
 **et\_xpwd**=findViewById(R.id.***et\_xpwd\_sys***);  
 **et\_zxpwd**=findViewById(R.id.***et\_zxpwd\_sys***);  
 **bt\_modify**=findViewById(R.id.***bt\_modify\_sys***);  
 **bt\_cancel**=findViewById(R.id.***bt\_cancel\_sys***);  
 **mhelper**=**new** MyDBHelper(SysSettingActivity.**this**);  
 **db**=**mhelper**.getWritableDatabase();  
 }  
 *//第三步：显示当前登录的用户名-------代码* **private void** displayInfo() {  
 **name**=getSharedPreferences(**"userinfo"**,0).getString(**"username"**,**""**);  
 **pwd**=getSharedPreferences(**"userinfo"**,0).getString(**"userpwd"**,**""**);  
 **txt\_user**.setText(**name**);  
 }  
 *//第四步：修改按钮功能----------------------代码* **private void** btnModify() {  
 **bt\_modify**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 *//获取三个输入框中的内容* String ypwd = **et\_ypwd**.getText().toString();*//获取输入的原密码* String xpwd = **et\_xpwd**.getText().toString();*//获取输入的新密码* String zxpwd = **et\_zxpwd**.getText().toString();*//获取第二次输入的新密码  
 //对每个密码进行逻辑判断* **if**(ypwd.equals(**""**)){  
 Toast.*makeText*(SysSettingActivity.**this**, **"请输入原始密码"**, Toast.***LENGTH\_SHORT***).show();  
 }**else if**(!ypwd.equalsIgnoreCase(**pwd**)){  
 Toast.*makeText*(SysSettingActivity.**this**, **"输入的密码与原密码不一致"**, Toast.***LENGTH\_SHORT***).show();  
 }**else if**(xpwd.equals(**""**)){  
 Toast.*makeText*(SysSettingActivity.**this**, **"请输入新密码"**, Toast.***LENGTH\_SHORT***).show();  
 }**else if**(xpwd.equalsIgnoreCase(ypwd)){  
 Toast.*makeText*(SysSettingActivity.**this**, **"所输入的新密码与原密码不能相同"**, Toast.***LENGTH\_SHORT***).show();  
 }**else if**(zxpwd.equals(**""**)){  
 Toast.*makeText*(SysSettingActivity.**this**, **"请再次输入新密码"**, Toast.***LENGTH\_SHORT***).show();  
 }**else if**(!zxpwd.equalsIgnoreCase(xpwd)){  
 Toast.*makeText*(SysSettingActivity.**this**, **"两次输入的新密码不一致"**, Toast.***LENGTH\_SHORT***).show();  
 }**else**{  
 ContentValues values =**new** ContentValues();  
 values.put(**"pwd"**,xpwd);  
 **db**.update(**"tb\_userinfo"**,values,**"name=?"**,**new** String[]{**name**});  
 Toast.*makeText*(SysSettingActivity.**this**, **"密码修改成功"**, Toast.***LENGTH\_SHORT***).show();  
 Intent intent=**new** Intent(SysSettingActivity.**this**, LoginActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 }  
 });  
 }  
 *//第五步：取消按钮功能-----------------------代码* **private void** btncancel() {  
 **bt\_cancel**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent intent=**new** Intent(SysSettingActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 finish();  
 }  
 });  
 }  
}

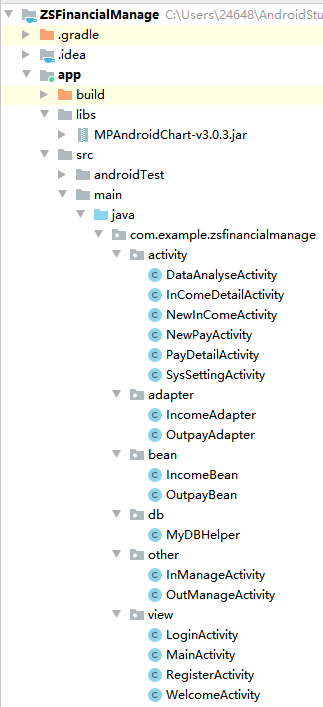
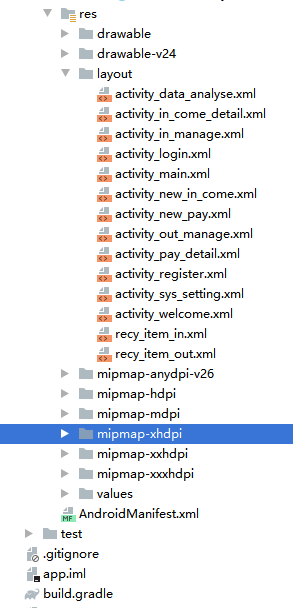
### 3.3 数据库存储设计

创建financial.db数据库，创建tb\_userinfo表，in\_come表，pay\_out表可供其他页面的功能调用。

**public class** MyDBHelper **extends** SQLiteOpenHelper {  
**private static final** String ***DBNAME***=**"financial.db"**;  
**private static final int *VERSION***=1;  
 **public** MyDBHelper(Context context) {  
 **super**(context, ***DBNAME***, **null**, ***VERSION***);  
 }  
 *//1 创建数据库* @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 *//创建用户表* db.execSQL(**"create table tb\_userinfo(id integer primary key autoincrement,name varchar(10),pwd varchar(15),email varchar(50),phone varchar(11))"**);  
 *//创建收入表* db.execSQL(**"create table in\_come(id integer primary key autoincrement,inmoney double,intime varchar(20),intype varchar(30),inpayer varchar(100),inremark varchar(500))"**);  
 *//创建支出表* db.execSQL(**"create table pay\_out(id integer primary key autoincrement,outmoney double,outtime varchar(20),outtype varchar(30),outpayee varchar(100),outremark varchar(500))"**);  
 }  
 *//2 升级数据库* @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  
 }  
}

## 4 系统编码实现

### 4.1 项目框架

### 4.2 交互实现

由于布局文件代码众多，再此只选择一部分显示：

***activity\_main.xml***

<**ImageButton  
 android:layout\_width="match\_parent"  
 android:layout\_height="250dp"  
 android:background="@drawable/main\_top"**/>  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="40dp"  
 android:gravity="center"**>  
 <**Button  
 android:id="@+id/bt\_newincome\_main"  
 android:layout\_width="90dp"  
 android:layout\_height="90dp"  
 android:background="@drawable/new\_pay\_in"  
 android:layout\_marginRight="50dp"**/>  
 <**Button  
 android:id="@+id/bt\_incomedetail\_main"  
 android:layout\_width="90dp"  
 android:layout\_height="90dp"  
 android:background="@drawable/my\_pay\_in"  
 android:layout\_marginRight="50dp"**/>  
 <**Button  
 android:id="@+id/bt\_newpay\_main"  
 android:layout\_width="90dp"  
 android:layout\_height="90dp"  
 android:background="@drawable/new\_pay\_out"** />  
 </**LinearLayout**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="20dp"  
 android:gravity="center"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="新增收入"  
 android:textSize="15sp"  
 android:textColor="#ffffff"  
 android:layout\_marginRight="80dp"**/>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="收入明细"  
 android:textSize="15sp"  
 android:textColor="#ffffff"  
 android:layout\_marginRight="80dp"**/>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="新增支出"  
 android:textSize="15sp"  
 android:textColor="#ffffff"**/>  
 </**LinearLayout**>  
  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="40dp"  
 android:gravity="center"**>  
 <**Button  
 android:id="@+id/bt\_paydetail\_main"  
 android:layout\_width="90dp"  
 android:layout\_height="90dp"  
 android:background="@drawable/my\_pay\_out"  
 android:layout\_marginRight="50dp"**/>  
 <**Button  
 android:id="@+id/bt\_dataanalyse\_main"  
 android:layout\_width="90dp"  
 android:layout\_height="90dp"  
 android:background="@drawable/data\_administration"  
 android:layout\_marginRight="50dp"**/>  
 <**Button  
 android:id="@+id/bt\_syssetting\_main"  
 android:layout\_width="90dp"  
 android:layout\_height="90dp"  
 android:background="@drawable/sys\_setting"** />  
 </**LinearLayout**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="20dp"  
 android:gravity="center"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="支出明细"  
 android:textSize="15sp"  
 android:textColor="#ffffff"  
 android:layout\_marginRight="80dp"**/>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="数据分析"  
 android:textSize="15sp"  
 android:textColor="#ffffff"  
 android:layout\_marginRight="80dp"**/>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="系统设置"  
 android:textSize="15sp"  
 android:textColor="#ffffff"**/>  
 </**LinearLayout**>

***activity\_new\_in\_come.xml***

<**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="250dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_marginRight="30dp"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="金 额:"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"**/>  
 <**EditText  
 android:id="@+id/et\_money\_newin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="0.00"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"  
 android:gravity="center"**/>  
</**LinearLayout**>  
<**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_marginRight="30dp"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="日 期:"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"**/>  
 <**EditText  
 android:id="@+id/et\_time\_newin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="2020-05-12"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"  
 android:gravity="center"**/>  
</**LinearLayout**>  
<**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_marginRight="30dp"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="类 型:"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"**/>  
 <**Spinner  
 android:id="@+id/sp\_type\_newin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:entries="@array/incometype"**/>  
</**LinearLayout**>  
<**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_marginRight="30dp"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="付款方:"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"**/>  
 <**EditText  
 android:id="@+id/et\_payer\_newin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="海明有限公司"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"  
 android:gravity="center"**/>  
</**LinearLayout**>  
<**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_marginRight="30dp"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="备 注:"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"**/>  
 <**EditText  
 android:id="@+id/et\_remark\_newin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="远程技术指导费"  
 android:textColor="#000000"  
 android:textSize="20sp"  
 android:textStyle="bold"  
 android:gravity="center"**/>  
</**LinearLayout**>  
<**Button  
 android:id="@+id/bt\_save\_newin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="保存"  
 android:textSize="20sp"  
 android:textColor="#ffffff"  
 android:layout\_marginLeft="60dp"  
 android:layout\_marginRight="60dp"  
 android:background="@drawable/btn\_style\_one"  
 android:layout\_marginTop="40dp"**/>  
<**Button  
 android:id="@+id/bt\_cancel\_newin"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="取消"  
 android:textSize="20sp"  
 android:textColor="#000000"  
 android:layout\_marginLeft="60dp"  
 android:layout\_marginRight="60dp"  
 android:background="@drawable/btn\_style\_two"  
 android:layout\_marginTop="20dp"**/>

***activity\_in\_come\_detail.xml***

<**androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recy\_view\_indetail"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="140dp"  
 android:layout\_marginLeft="30dp"  
 android:layout\_marginRight="30dp"  
 android:layout\_marginBottom="40dp"**/>

### 4.3 单元测试

设备检测：OPPO R9

API检测：没有问题

功能测试：每个功能都能实现，达到预期目标

## 5 系统测试发布

手机实测图：

图5-1 图5-2 图5-3

图5-4 图5-5 图5-6

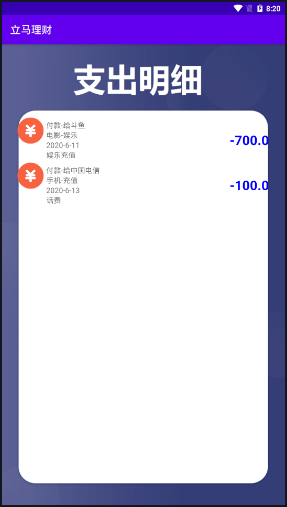
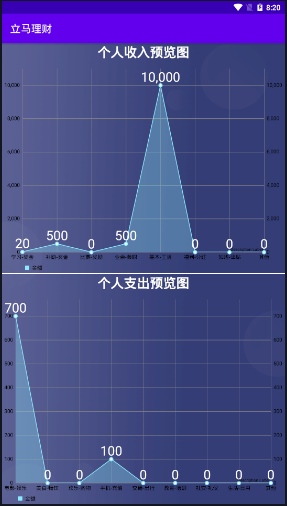
  

图5-7 图5-8 图5-9



图5-10

## 参考文献

* 兰红、李淑芝编著，《Android Studio移动开发从入门到实战》，清华大学出版社，2018年11月
* 郭宏志，Android应用开发详解[M]，电子工业出版社，2010年1月
* 吴亚峰 等编著，Android核心技术与实例详解[M]，人民邮电出版社，2010年10月
* 孙翠改等编著，《Android移动开发案例教程》，MOOC
* 盖索林编著，《Android开发入门指南》，人民邮电出版社
* 写Bug的狐狸，Bilibili.com