目录

Android 案例开发 实验报告

成绩	

实验名称 实验一 公共类和登录模块的设计与调试

专业班级 软件 **172** 班级学号 **201715030208**

姓 名 朱洪龙 实验日期 **2019/03/09**

（报告内容包括：实验题目、实验环境、实验步骤、实验结果、遇到的问题和解决办法、实验小结等。

【实验题目】

1. 创建个人理财通项目
2. 输入公共类代码
3. 创建登录窗口的布局文件，输入登录模块的 Activity 类代码，调通登录功能。

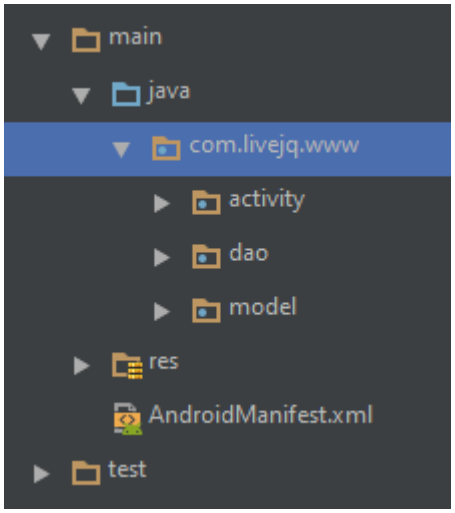
【实验环境】

OS:Windows 10
IDE:Android Studio 2.3.2
AVD:Nexus 5X API 23
minSdkVersion:21(android 5.0)
targetSdkVersion:23(android 6.0)

【实验步骤、程序调试过程中所有遇到的问题和解决办法】

(在这里不贴代码,只是详细分析)

建立项目包结构：



Activity:活动窗体文件
Dao:操作数据库文件
Model:保存数据文件

创建保存密码的模板(bean)

在 model 中创建 Tb_pwd 类文件，设置密码文本，自动获得 get,set 方法

创建数据库连接对象（DAO）

主要是继承 SqliteOpenHelper 和在构造方法中调用父类的构造方法,然后重写 onCreate()和 onUpgrade 方法，前者创建（若数据库中没有）或加载数据表，后者可写可不写.

创建操作数据库密码表对象（DAO）

在 dao 中创建 PwdDAO 类文件，其中主要是在其构造方法中调用数据库连接类并获得可操

作数据库的对象（用来执行 Sql 语句）,其后的方法都是基于 Sql 语句的增删改查（获得密码并通过密码模板保存和 return 出来的方法和查询数据表中是否存在密码的方法等）

创建登录布局文件和活动窗口

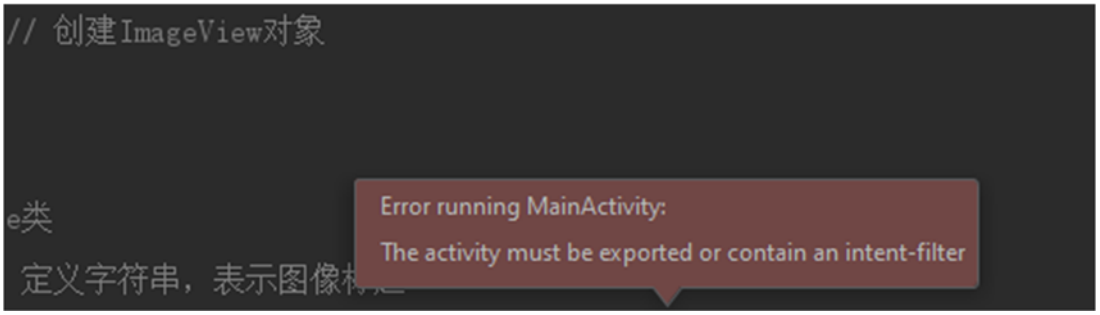
在 res 目录中的 layout 目录中，创建 login.xml 布局文件，其整体布局为 RelativeLayout 相对布局，然后是一个 TextView（输入密码文本）和一个 EditText(hint:输入密码框,layout_below:在输入密码文本下面)，然后是两个登录和取消 Button，定位都差不多（先让其中一个在输入框的下面，然后让另一个 Button 在这个 Button 的旁边）。

在 activity 中创建 Login 类文件，并设置其 R.layout.login，之后获取输入密码框和按钮并对两个按钮添加点击事件，获取密码框中的数据对其判断，这里就要用到密码操作对象里的方法，主要判断用户是否为新用户，若为新用户，则判断没有输入密码时登录到主窗体；若已有用户，则判断输入的密码是否正确，判断为正确时登录到主窗体，点击取消时调用 finish（）方法关闭窗体

修改 AndroidManifest.xml 文件并调试运行

设置 Login 窗体为启动时首先显示的窗体，还得声明 MainActivity 为一个 Activity。

第一次运行：

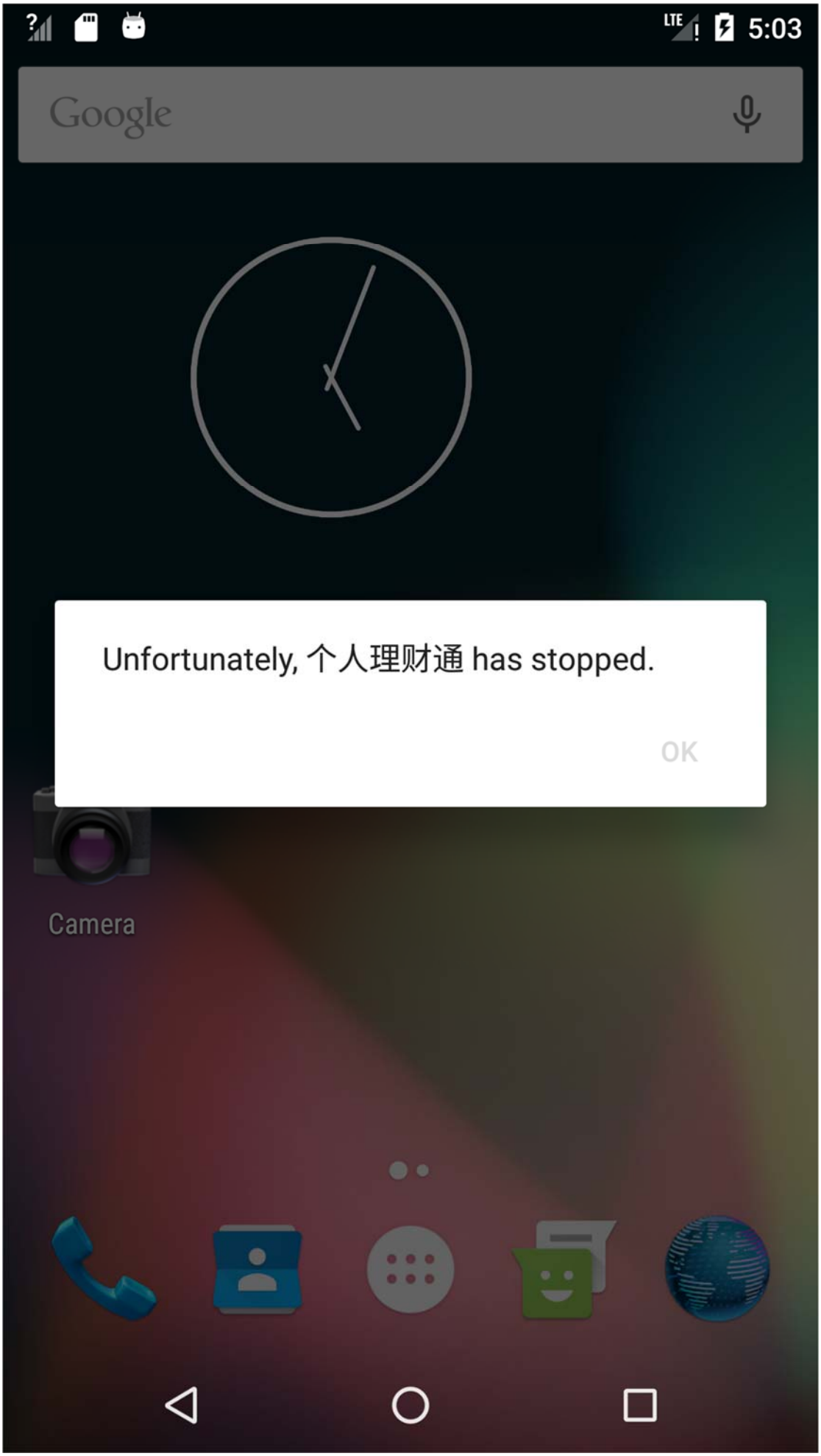


修正之后：

```
<activity android:name="com.livejq.www.activity.MainActivity"
    android:label="个人理财通">
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

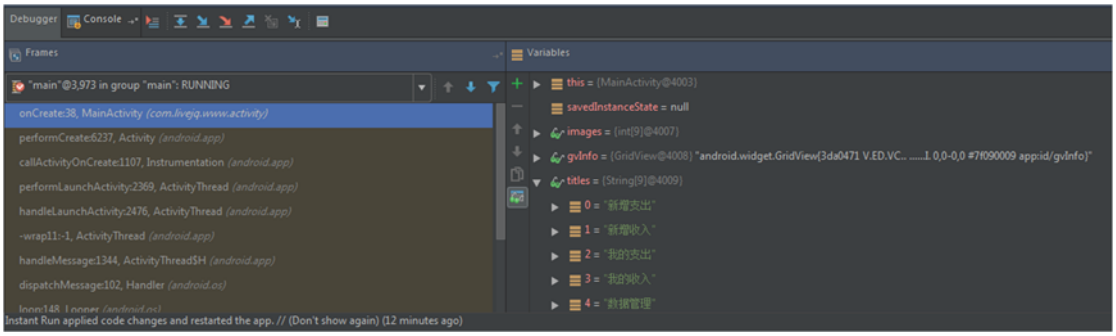
第二次运行：

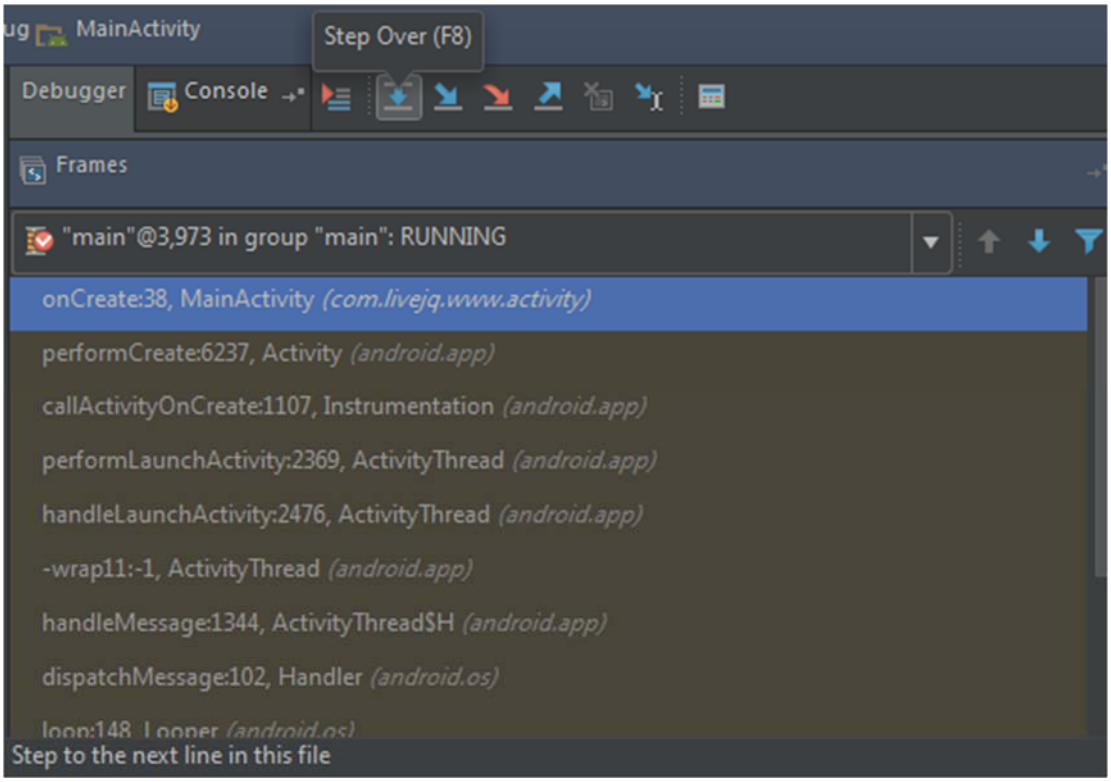

android

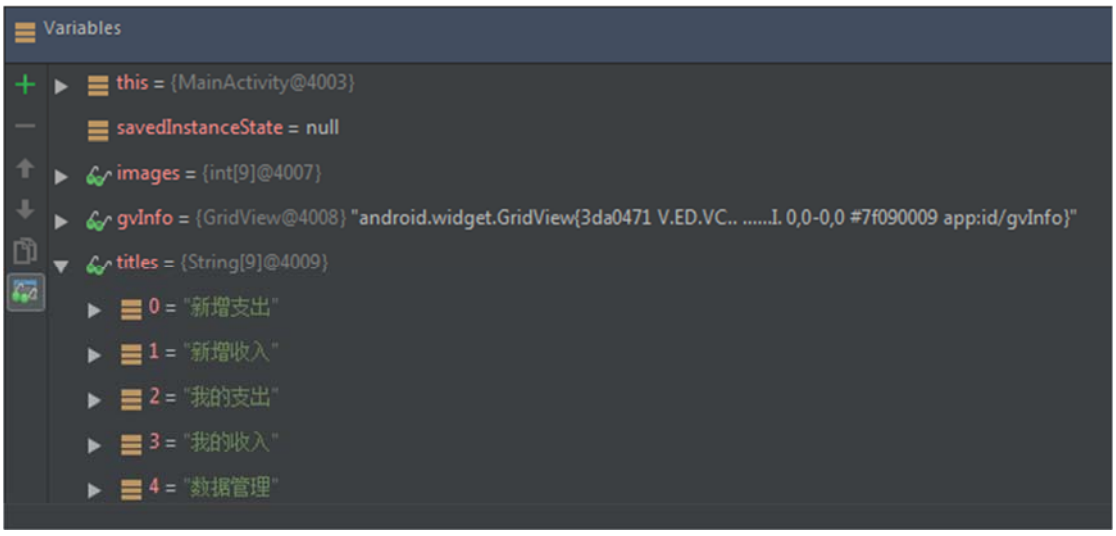

5:03
Google
Unfortunately, 个人理财通 has stopped.
OK
Camera

初步断定为 MainActivity 中的错误（因为 Login 中就添加了两个点击事件，还没点击就不行了）

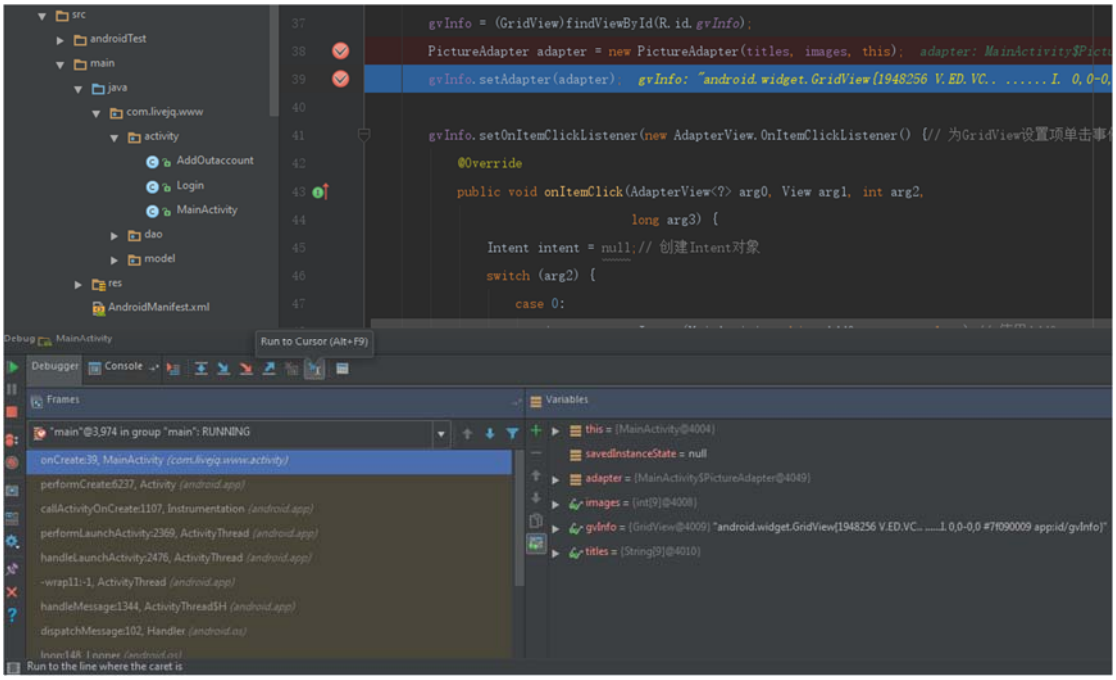
设置断点：



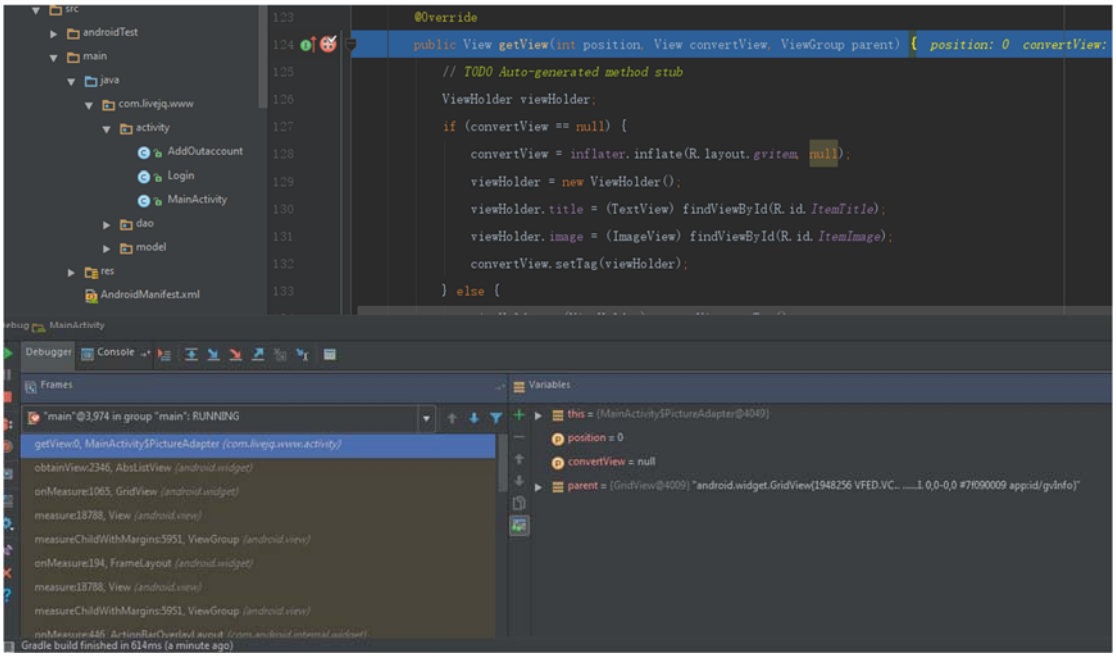

MainActivity
Step Over (F8)
Debugger
Console
Frames
"main"@3,973 in group "main": RUNNING
onCreate:38, MainActivity (com.livejq.www.activity)
performCreate:6237, Activity (android.app)
callActivityOnCreate:1107, Instrumentation (android.app)
performLaunchActivity:2369, ActivityThread (android.app)
handleLaunchActivity:2476, ActivityThread (android.app)
-wrap11:-1, ActivityThread (android.app)
handleMessage:1344, ActivityThread$H (android.app)
dispatchMessage:102, Handler (android.os)
Step to the next line in this file


Variables
this = {MainActivity@4003}
savedInstanceState = null
images = {int[9]@4007}
gvInfo = {GridView@4008} "android.widget.GridView{3da0471 V.ED.VC.. .....I. 0,0-0,0 #7f090009 app:id/gvInfo}"
titles = {String[9]@4009}
0 = "新增支出"
1 = "新增收入"
2 = "我的支出"
3 = "我的收入"
4 = "数据管理"

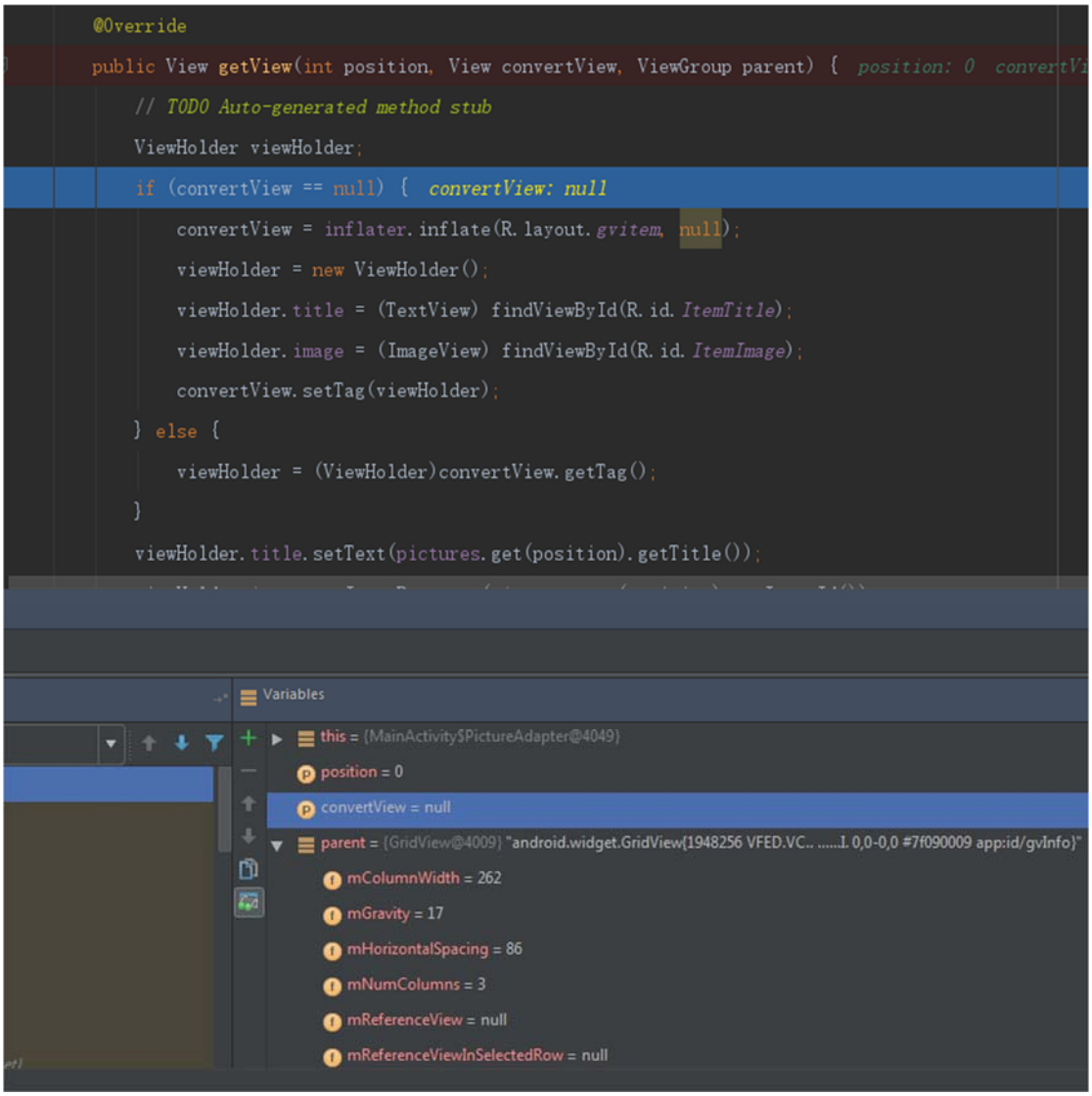
Step over 后：

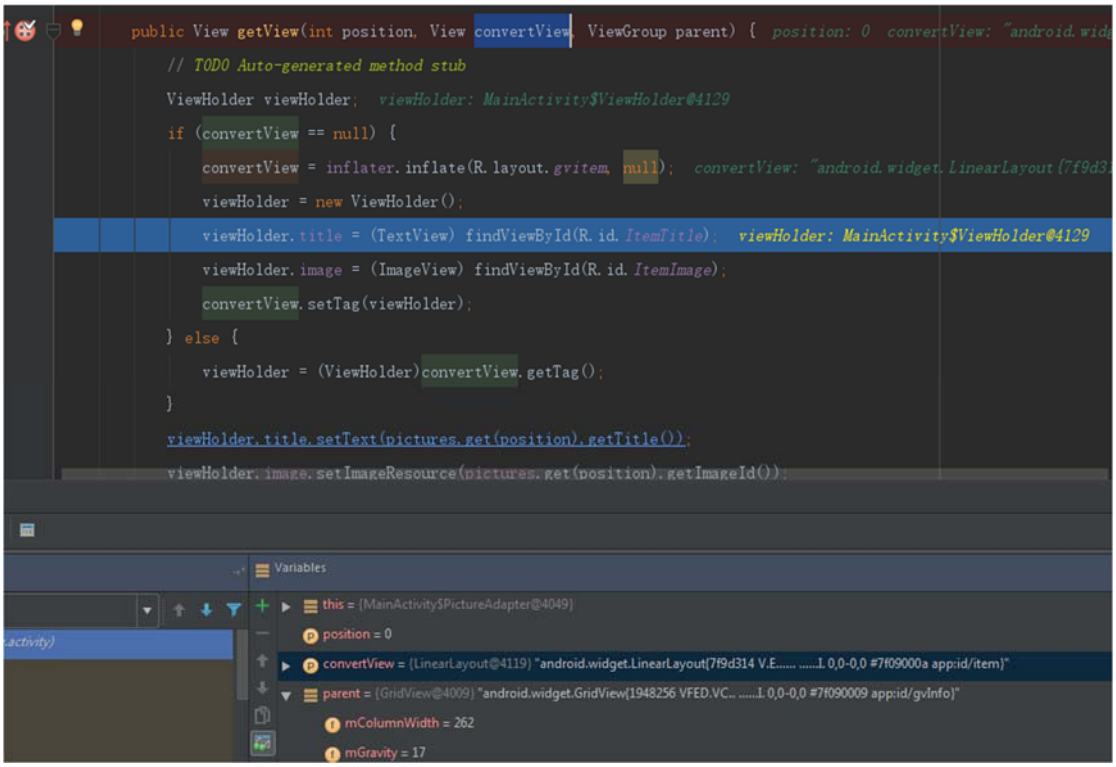


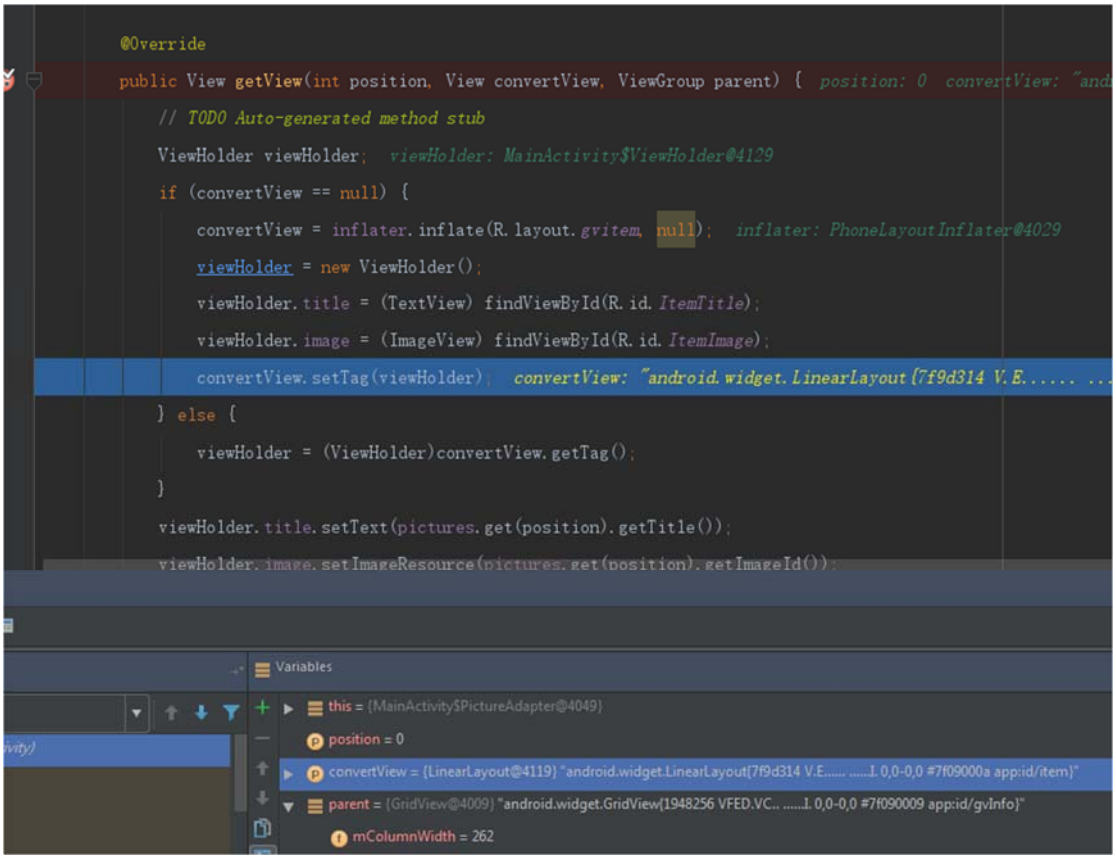
Run to cursor 后：

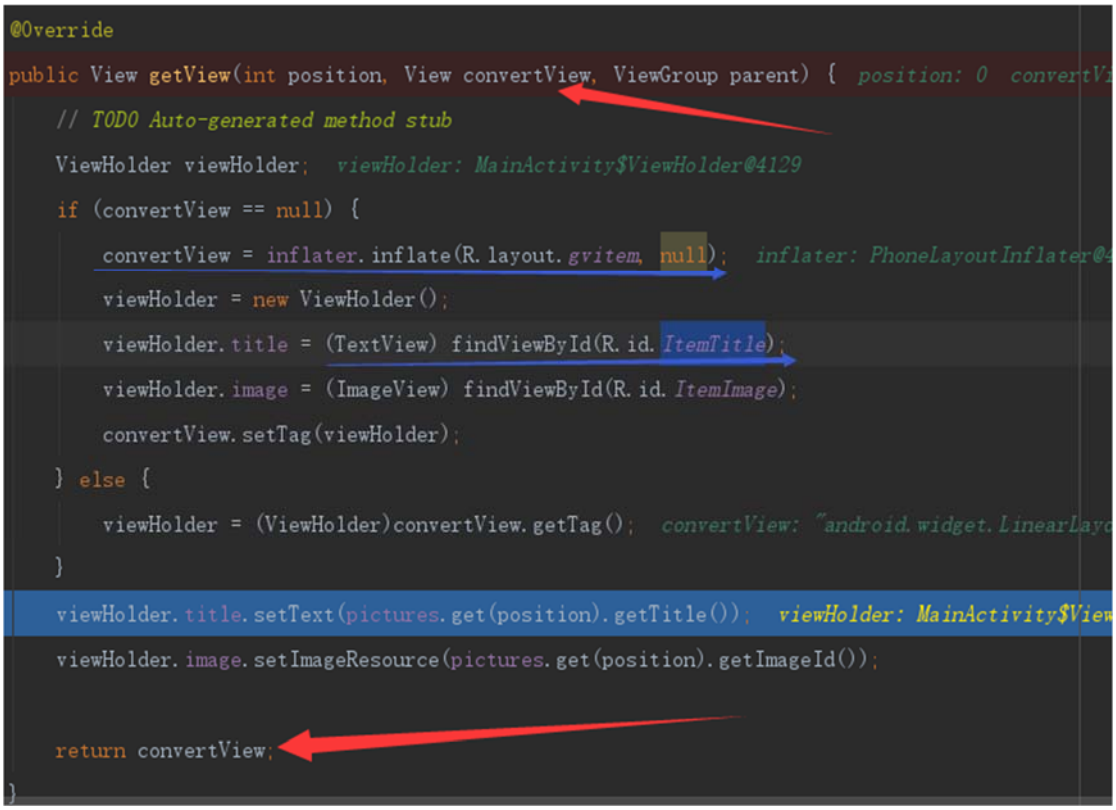


Step over 后：


@Override
public View getView(int position, View convertView, ViewGroup parent) { position: 0 convertVi
// TODO Auto-generated method stub
ViewHolder viewHolder;
if (convertView == null) { convertView: null
convertView = inflater.inflate(R.layout.gvitem, null);
viewHolder = new ViewHolder();
viewHolder.title = (TextView) findViewById(R.id.ItemTitle);
viewHolder.image = (ImageView) findViewById(R.id.ItemImage);
convertView.setTag(viewHolder);
} else {
viewHolder = (ViewHolder)convertView.getTag();
}
viewHolder.title.setText(pictures.get(position).getTitle());
Variables
this = {MainActivity$PictureAdapter@4049}
position = 0
convertView = null
parent = {GridView@4009} "android.widget.GridView{1948256 VFED.VC.. .....I. 0,0-0,0 #7f090009 app:id/gvInfo}"
mColumnWidth = 262
mGravity = 17
mHorizontalSpacing = 86
mNumColumns = 3
mReferenceView = null
mReferenceViewInSelectedRow = null


public View getView(int position, View convertView, ViewGroup parent) { position: 0 convertView: "android.wid
// TODO Auto-generated method stub
ViewHolder viewHolder; viewHolder: MainActivity$ViewHolder@4129
if (convertView == null) {
convertView = inflater.inflate(R.layout.gvitem, null); convertView: "android.widget.LinearLayout{7f9d3
viewHolder = new ViewHolder();
viewHolder.title = (TextView) findViewById(R.id.ItemTitle); viewHolder: MainActivity$ViewHolder@4129
viewHolder.image = (ImageView) findViewById(R.id.ItemImage);
convertView.setTag(viewHolder);
} else {
viewHolder = (ViewHolder)convertView.getTag();
}
viewHolder.title.setText(pictures.get(position).getTitle());
viewHolder.image.setImageResource(pictures.get(position).getImageId());
Variables
this = {MainActivity$PictureAdapter@4049}
position = 0
convertView = {LinearLayout@4119} "android.widget.LinearLayout{7f9d314 V.E..... .....I. 0,0-0,0 #7f09000a app:id/item}"
parent = {GridView@4009} "android.widget.GridView{1948256 VFED.VC.. .....I. 0,0-0,0 #7f090009 app:id/gvInfo}"
mColumnWidth = 262
mGravity = 17


@Override
public View getView(int position, View convertView, ViewGroup parent) { position: 0 convertView: "and
// TODO Auto-generated method stub
ViewHolder viewHolder; viewHolder: MainActivity$ViewHolder@4129
if (convertView == null) {
convertView = inflater.inflate(R.layout.gvitem, null); inflater: PhoneLayoutInflater@4029
viewHolder = new ViewHolder();
viewHolder.title = (TextView) findViewById(R.id.ItemTitle);
viewHolder.image = (ImageView) findViewById(R.id.ItemImage);
convertView.setTag(viewHolder); convertView: "android.widget.LinearLayout{7f9d314 V.E..... ...
} else {
viewHolder = (ViewHolder)convertView.getTag();
}
viewHolder.title.setText(pictures.get(position).getTitle());
viewHolder.image.setImageResource(pictures.get(position).getImageId());
Variables
this = {MainActivity$PictureAdapter@4049}
position = 0
convertView = {LinearLayout@4119} "android.widget.LinearLayout{7f9d314 V.E..... .....I. 0,0-0,0 #7f09000a app:id/item}"
parent = {GridView@4009} "android.widget.GridView{1948256 VFED.VC.. .....I. 0,0-0,0 #7f090009 app:id/gvInfo}"
mColumnWidth = 262

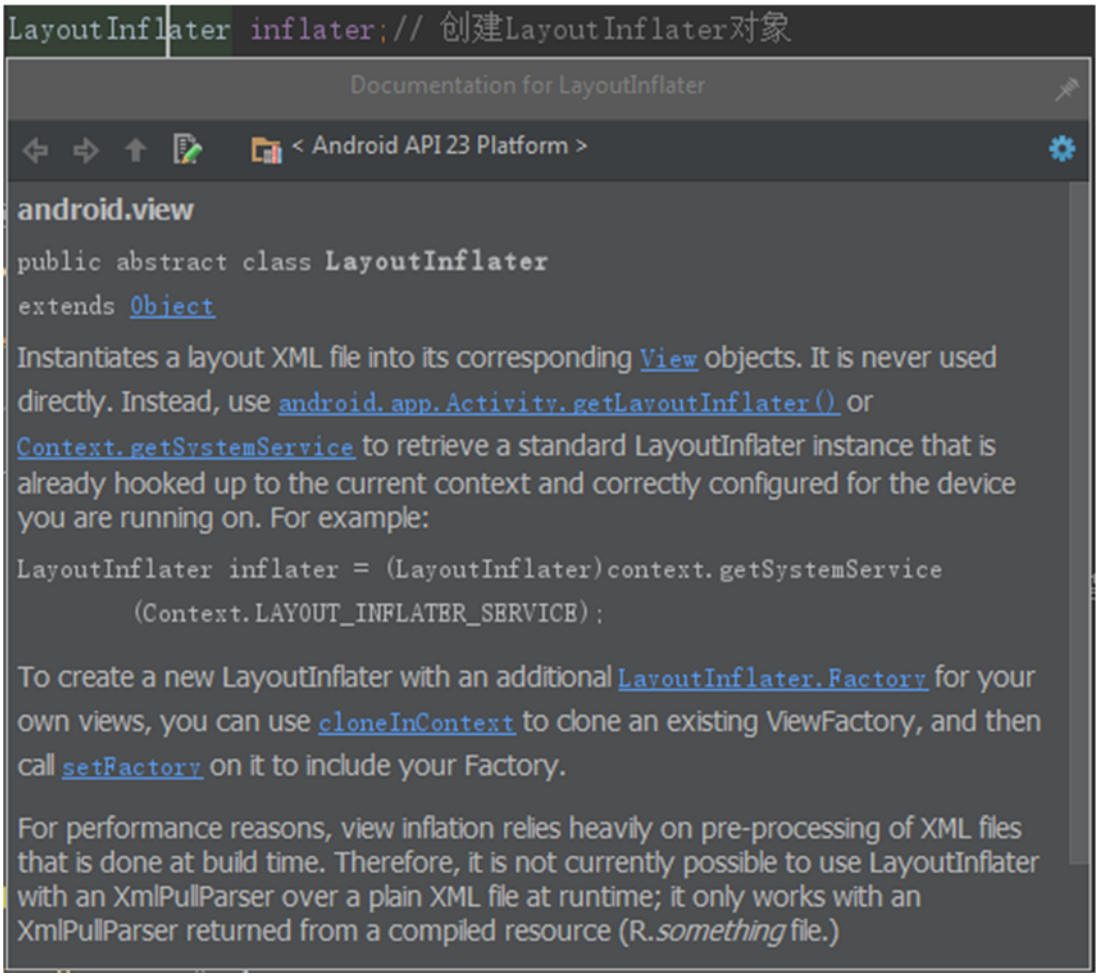


```
@Override
public View getView(int position, View convertView, ViewGroup parent) {  position: 0  convertVi
    // TODO Auto-generated method stub
    ViewHolder viewHolder;  viewHolder: MainActivity$ViewHolder@4129
    if (convertView == null) {
        convertView = inflater.inflate(R.layout.gvitem, null);  inflater: PhoneLayoutInflater@4
        viewHolder = new ViewHolder();
        viewHolder.title = (TextView) findViewById(R.id.ItemTitle);
        viewHolder.image = (ImageView) findViewById(R.id.ItemImage);
        convertView.setTag(viewHolder);
    } else {
        viewHolder = (ViewHolder)convertView.getTag();  convertView: "android.widget.LinearLayo
    }
    viewHolder.title.setText(pictures.get(position).getTitle());  viewHolder: MainActivity$View
    viewHolder.image.setImageResource(pictures.get(position).getImageId());

    return convertView;
}
```

```
gvInfo = (GridView)findViewById(R.id.gvInfo);
PictureAdapter adapter = new PictureAdapter(titles, images, this);
gvInfo.setAdapter(adapter);
```

在重写的方法 getView()中，ConvertView 是这个这个适配器最终返回的一个视图，而之后将这个适配器设置到 gvInfo 中。然而在 getView()中获取视图中保存的数据类型（标题和图片）时，是以 this 获取的（未进行特殊处理），而 gvInfo（可以认为是用来装载另一个视图中的数据）也是 this 获取的；为此这两个视图之间无法取得联系（数据通信），先来看看 LayoutInflater 的作用。



大概意思就是说可以初始化一个 xml 的布局文件来响应或与之通信的一个视图对象（不能直接使用），但可以通过已挂钩的当前上下文来获取。

So,

```
class PictureAdapter extends BaseAdapter {// 创建基于BaseAdapter的子类
    private LayoutInflater inflater;// 创建LayoutInflater对象
    private List<Picture> pictures;// 创建List泛型集合

    // 为类创建构造函数
    public PictureAdapter(String[] titles, int[] images, Context context) {
        super();
        pictures = new ArrayList<>();// 初始化泛型集合对象
        inflater = LayoutInflater.from(context);// 初始化LayoutInflater对象
        for (int i = 0; i < images.length; i++) {// 遍历图像数组
            Picture picture = new Picture(titles[i], images[i]);// 使用标题和图像生成Picture对象
            pictures.add(picture);// 将Picture对象添加到泛型集合中
        }
    }
```

也可以：

```
class PictureAdapter extends BaseAdapter {// 创建基于BaseAdapter的子类
    private LayoutInflater inflater;// 创建LayoutInflater对象
    private List<Picture> pictures;// 创建List泛型集合

    // 为类创建构造函数
    public PictureAdapter(String[] titles, int[] images, Context context) {
        super();
        pictures = new ArrayList<>();// 初始化泛型集合对象
        inflater = getLayoutInflater();// 初始化LayoutInflater对象
        for (int i = 0; i < images.length; i++) {// 遍历图像数组
            Picture picture = new Picture(titles[i], images[i]);// 使用标题和图像生成Picture对象
            pictures.add(picture);// 将Picture对象添加到泛型集合中
        }
    }
```

修正后（用 LayoutInflater 处理过的视图对象来加载需要作为适配器的视图（图片和标题））：

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
    // TODO Auto-generated method stub
    ViewHolder viewHolder;
    if (convertView == null) {
        convertView = inflater.inflate(R.layout.gvitem, null);
        viewHolder = new ViewHolder();
        viewHolder.title = (TextView) convertView.findViewById(R.id.ItemTitle);
        viewHolder.image = (ImageView) convertView.findViewById(R.id.ItemImage);
        convertView.setTag(viewHolder);
    } else {
        viewHolder = (ViewHolder)convertView.getTag();
    }
    viewHolder.title.setText(pictures.get(position).getTitle());
    viewHolder.image.setImageResource(pictures.get(position).getImageId());

    return convertView;
}
```

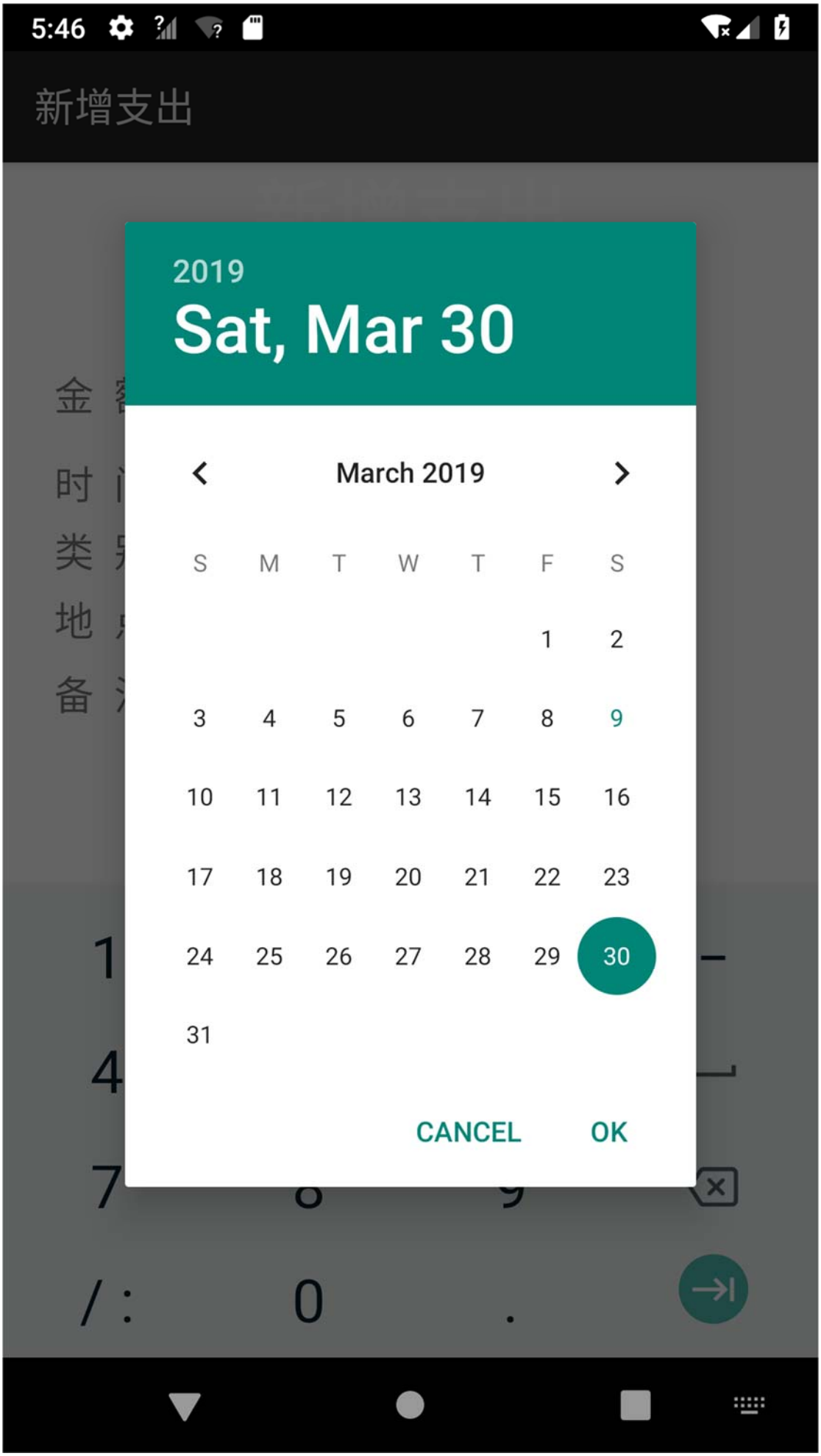
【实验结果】

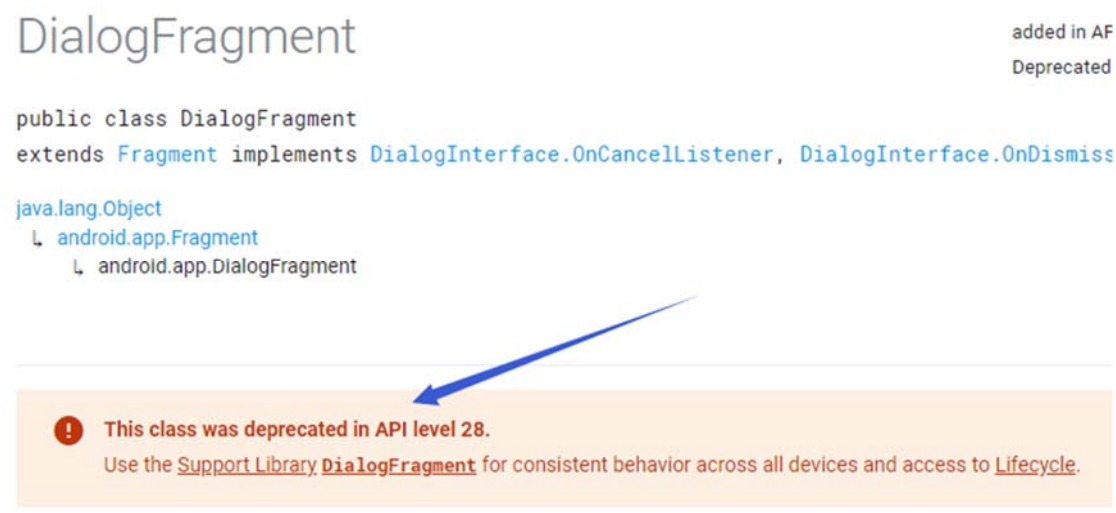

6:19
登录
请输入密码：
请输入密码
登录
取消


6:19
个人理财通
新增支出
新增收入
我的支出
我的收入
数据管理
系统设置
收支便签
帮助
欢迎!
退出


5:45
新增支出
新增支出
金 额:
0.00
时 间:
2019-3-9
类 别:
餐费
地 点:
备 注:
保存
取消


5:46
新增支出
新增支出
金 额:
0.00
时 间:
2019-3-30
类 别:
地 点:
备 注:
餐费
书本
衣服
车费
其他
保存
取消


5:46
新增支出
2019
Sat, Mar 30
March 2019
S M T W T F S
1 2
3 4 5 6 7 8 9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
31
CANCEL
OK




```
txtTime.setOnClickListener((arg0) -> {
        // TODO Auto-generated method stub
        showDialog(DATE_DIALOG_ID);// 显示日期选择对话框
}
```

'showDialog(int)' is deprecated more... (Ctrl+F1)

```
@Override
protected Dialog onCreateDialog(int id){// 重写onCreateDialog方法
    switch (id) {
        case DATE_DIALOG_ID:// 弹出日期选择对话框
            return new DatePickerDialog(this, mDateSetListener, mYear, mMonth,
                    mDay);
    }
    return null;
}
```

【实验小结】

对于调试还不太熟，日后还得多加操作。在处理新增支出时，发现在 showDialog()方法和 Activity 中的 onCreateDialog()方法已经弃用了，查了官方文档说明是改用了继承自 Fragment 的 DialogFragment（而在这篇文档顶端却说了此 API 在 android API 28 已弃用）,哎~。
将在下篇报告中详谈。