图片选择组件，目标是做到类似为微信发朋友圈时的效果

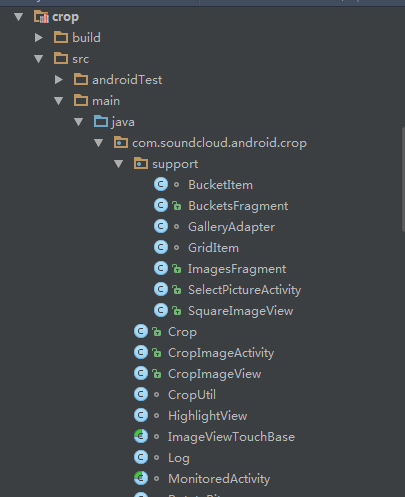
实际：

实现单张图片选择，剪切，保存功能 可用于用户更换头像

实现多张图片选择功能 提供预览功能 可用于表单数据上传

#### 过程

###### 实现单张图片选择，剪切，保存功能



名字为crop 库。借鉴网上部分代码。所以包的命名也比较随意。

Support里面为选择单张图片的代码

Crop里面提供了从Fragment和Activity中调用的方法。

比如在Fragment里面进行调用：

Crop.*pickImage*(\_mActivity,SupplyFragment.this);

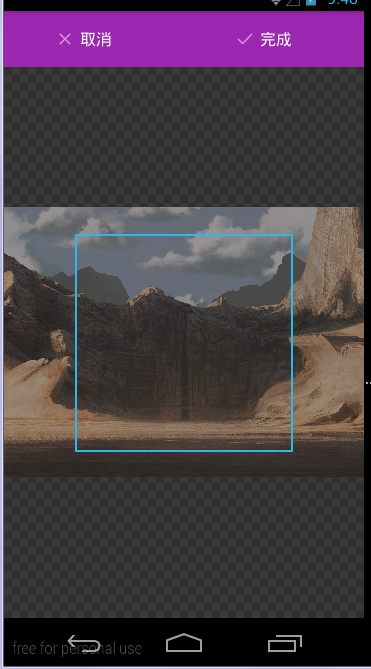
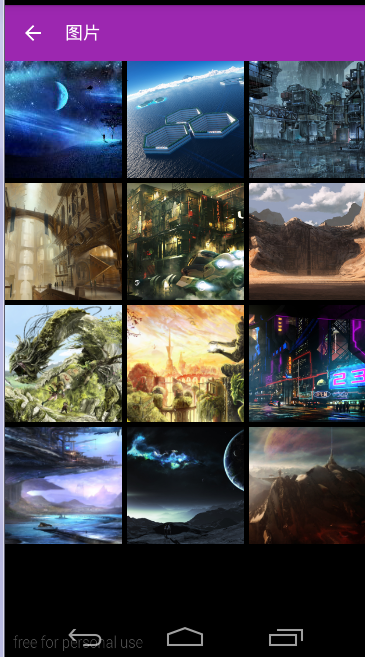
进入图片选择界面

同时在调用者的

onActivityResult

方法中 截取requestCode

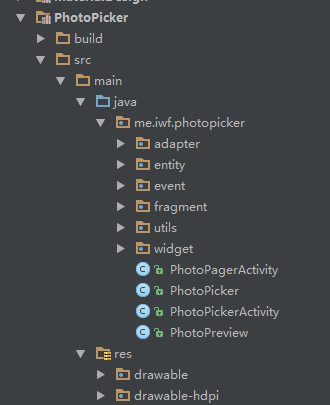
case Crop.*REQUEST\_PICK*:  
 if(resultCode == *RESULT\_OK*){  
 beginCrop(data.getData()); //开始剪切  
 }  
 break;  
case Crop.*REQUEST\_CROP*:  
 handleCrop(resultCode, data); //剪切成功，显示  
 break;

###### 实现多张图片选择功能 提供预览功能

多张图片选择器的名字为 PhotoPicker lib

参考一些开源项目的实现，优化一些不合理的地方。



Utils里面封装了数据的加载器。

其他的都比较常规。

由于综合了一些开源项目，核心代码的逻辑有些紊乱。

另外在预览大图的时候，如果图片过大，太吃内存。

使用：

选择：

PhotoPicker.*builder*()  
 .setPhotoCount(9)  
 .setGridColumnCount(3)  
 .start(\_mActivity,SupplyFragment.this);

预览：

PhotoPreview.*builder*()  
 .setPhotos(selectedPhotos)  
 .setCurrentItem(position)  
 .start(\_mActivity,SupplyFragment.this);

PhotoPicker和PhotoPreview为和外界交互的两个类

同样在

onActivityResult

方法中

case PhotoPicker.*REQUEST\_CODE*:  
case PhotoPreview.*REQUEST\_CODE*:  
 List<String> photos = null;  
 if (data != null) {  
 photos = data.getStringArrayListExtra(PhotoPicker.*KEY\_SELECTED\_PHOTOS*);  
 }  
 selectedPhotos.clear();  
  
 if (photos != null) {  
 selectedPhotos.addAll(photos);  
 }  
 photoAdapter.notifyDataSetChanged();  
 break;

