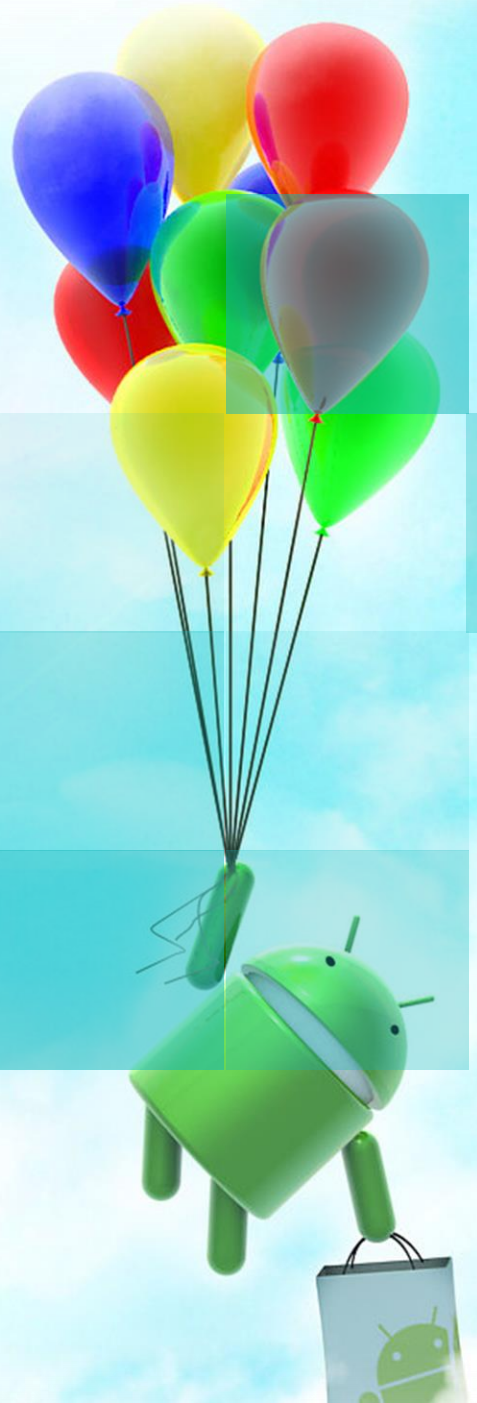
Android
云存储客户端开发



项目8

文件操作模块

任务01 文件夹的创建

任务02 文件夹和文件重命名

任务03 复制和粘贴

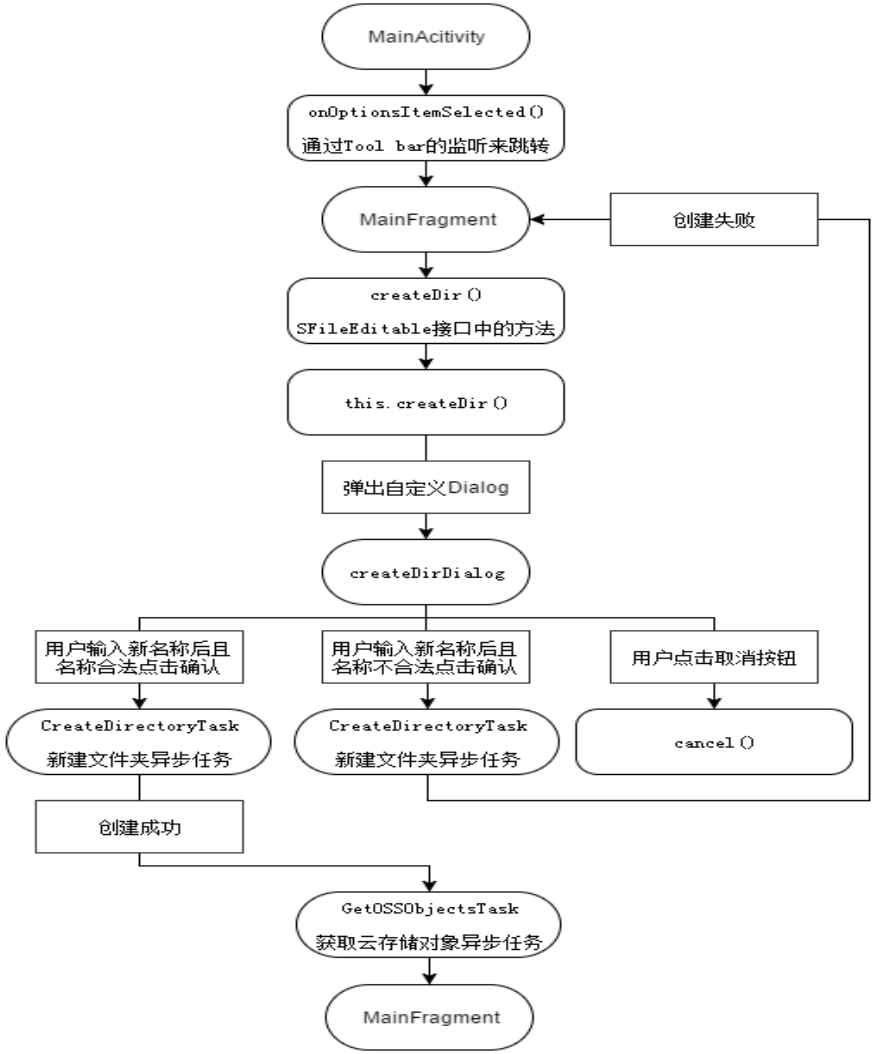
文件操作模块

在前面项目的界面基础上，根据项目5所学知识，利用Openstack Swift SDK完成已注册用户登录到Swift服务和用户注册的功能。

1

文件夹的创建

功能流程图



1 文件夹的创建

功能需求

1）视图层，界面实现

根据原型图设计实现新建文件夹窗口View。

1

文件夹的创建

功能需求

2）控制层，用户点击新建后出现对话框

点击菜单栏（menu）下的新建目录，页面弹出对话框（Dialog），输入新建目录的名称，点击确定按钮。

3）服务层，用户输入目录名称后的操作

读取输入的目录的名称，调用Swift服务创建目录

4）控制层，返回结果处理(MVC/MVP)

返回值处理：认证可能的情况包含：

（1）文件名称不重复，不含特殊字符，新建目录在View中显示；

（2）已存在文件名，提示用户文件名重复；

（3）文件名包含不符合规定的特殊字符，提示用户文件名不合法

1 文件夹的创建

实现步骤

1）选择File\Open..，点击弹出选择project75目录下面的项目“swiftstorage”。

2）界面实现

菜单栏新建目录点击弹出对话框（Dalog），布局文件为input_text_edit_dialog.xml其中包含一个输入框（Edittext），两个按钮（Button），一个确定一个取消。

组件	作用	ID
EditText	输入目录名称	edit_text
Button	确认按钮	btnEnter
Button	取消按钮	btnCancel

1 文件夹的创建

实现步骤

3）在MainActivity中对新建文件夹按钮进行处理

在MainActivity中对菜单（menu）的新建目录选项进行监听

```
if (id == R.id.action_create_dir) {
    //弹出输入框
    this.currentFragment.createDir(null);
}
```

调用SFileEditable接口下的createDir（）方法

1 文件夹的创建

实现步骤

4）在MainFragment中完成对新建目录的处理

在这个操作中，主要完成2个操作，首先弹出一个输入新目录的对话框，用户可以输入名称；其次，创建实现SFileEditable接口下的createDir（）的方法

```
 //创建目录，弹出对话框
private AlertDialog createDirDialog;
private void createDir() {
//创建弹出对话框
}
@Override
public void createDir(String filePath) {
    this.createDir();
}
```

this.createDir();调用createDir()方法

1 文件夹的创建

实现步骤

实现说明:

（1）弹出对话框（Dialog），在输入框中（Edittext）填入目录名

```
AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
LayoutInflater inflater = getActivity().getLayoutInflater();
```

（2）对确定按钮（Button）进行监听，用于启动异步认为

```
CreateDirectoryTask createDirectoryTask = new CreateDirectoryTask(dirName);
createDirectoryTask.execute();
```

（3）对取消按钮（Button）监听，关闭对话框（Dialog）。

```
createDirDialog.cancel();
```

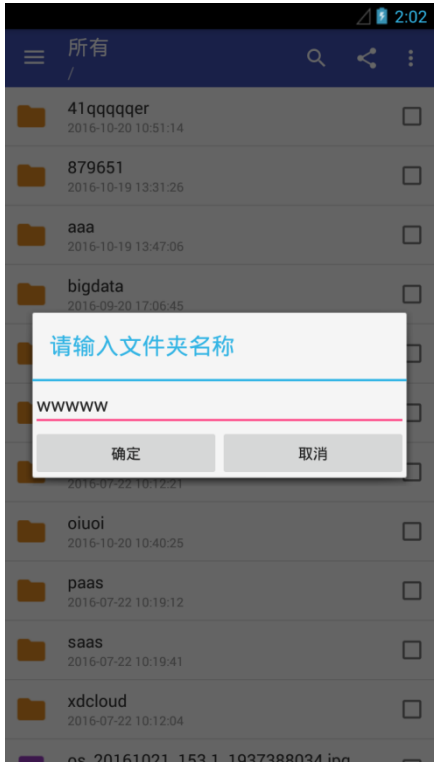
（4）完成操作后返回提示结果，刷新视图，新建的目录在View中显示，到此我们完成了新建目录的全部操作；

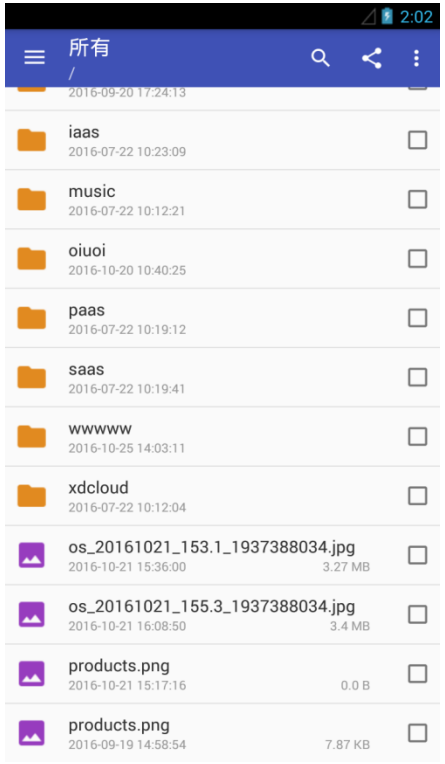
1 文件夹的创建

功能执行测试

1）执行效果

选择项目Android的“app”，点击工具条中运行Run “app”，执行的效果如图6-5所示，填入已经注册好的用户名和密码，进行登录验证。

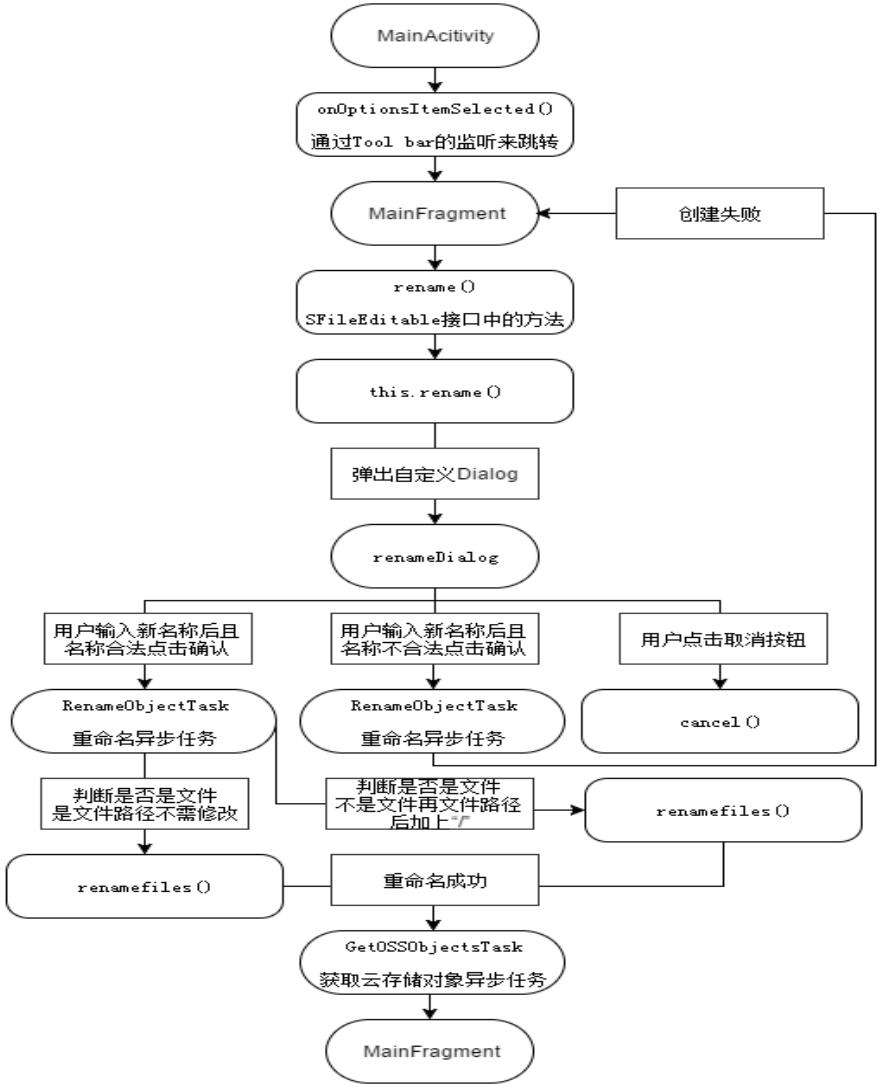




2

文件夹和文件重命名

功能流程图



2

文件夹和文件重命名

功能需求

1）视图层，界面实现

根据原型图设计实现重命名窗口View。

2 文件夹和文件重命名

功能需求

2）控制层，用户在实现文件夹和文件重命名的处理

登入后选中一个文件或文件夹，点击下拉菜单（menu）中的重命名，弹出对话框，对话框中有原本文件的名称，用户在输入框（Edittext）中修改后点击确定。

3）服务层，用户输入文件夹名后的操作

读取修改完成的文件名称，调用swift服务对文件重命名。

4）控制层，返回结果处理(MVC/MVP)

返回值处理：返回结果的情况包含：

（1）文件的新名称不重复，不含特殊字符，修改成功页面显示出现；

（2）文件的新名称重复，不含特殊字符，提示创建失败，文件名已存在；

（3）文件的新名称不重复，含特殊字符，提示创建失败，文件名不合法；

（4）未选择文件点击重命名，对话框不会弹出；

2 文件夹和文件重命名

实现步骤

1）导入项目

选择File\Open..，点击弹出选择project81目录下面的项目“swiftstorage”

2）界面实现

选中文件后点击菜单栏下的重命名，弹出对话框（Dalog），布局文件为input_text_edit_dialog.xml其中包含一个写着原文件名的输入框（Edittext），两个按钮（Button），一个确定一个取消。

组件	作用	ID
EditText	文件夹名称修改	edit_text
Button	确认按钮	btnEnter
Button	取消按钮	btnCancel

此xlm文件中所涉及的控件的使用，具体请参考前面项目4Android基础。

2 文件夹和文件重命名

实现步骤

3）在MainActivity中对重命名按钮进行处理

在MainActivity中对下拉菜单（menu）中的重命名进行监听.

```
else if (id == R.id.action_rename) {
    this.currentFragment.rename(null, null);
}
```

调用SFileEditable接口下的rename（）方法。

4）在MainFragment中完成对重命名的处理

在这里，依然是弹出改名对话框，然后实现SFileEditable接口下的rename（）的方法。

实现步骤说明：

（1）获取用户选中的文件,如果没有选中程序就不会往下运行，如果选中多个则取第一个文件。

（2）取得文件后，弹出对话框（Dialog），在对话框（Dialog）中的输入框（Edittext）填有原文件的名称，用户在此基础上进行修改。

（3）分别对对话框（Dialog）中的确定按钮（button）与取消按钮（button）进行监听，确定则启动异步任务，取消关闭对话框。

（4）如果是文件夹则要修改文件夹路径下的所有文件路径，如果是文件就直接调用swift服务进行修改。

2 文件夹和文件重命名

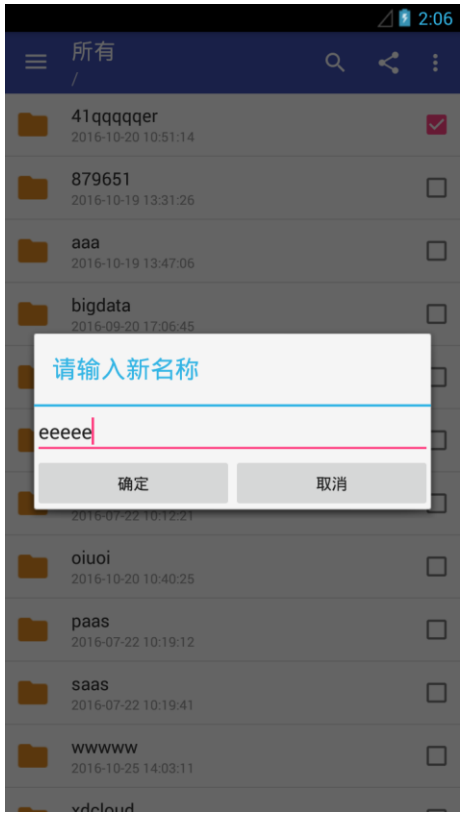
实现步骤

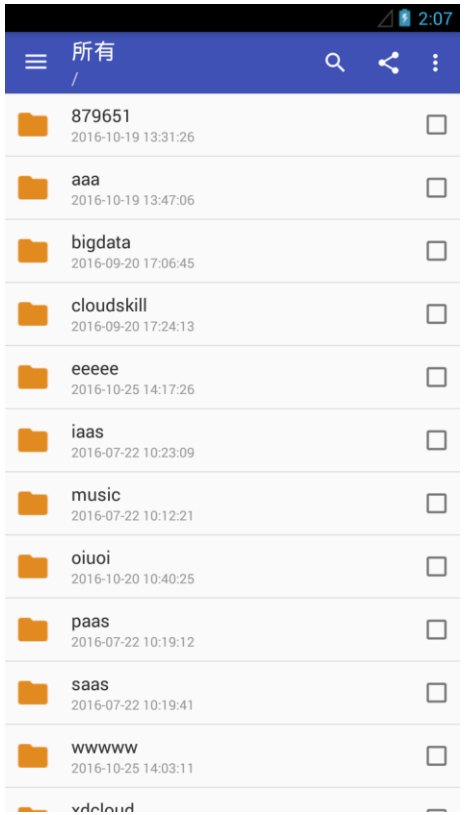
```
private void renameFiles(String cName, String path2, SFile sFile) {
    // 重命名
    if (sFile.isFile()) {
        getService().rename(cName, sFile.getName(), path2, sFile.getContentType());
    } else {//目录
        getService().rename(cName, sFile.getName(), path2, "text/directory");
    }
    // 重命名目录下文件
    for (SFile file : sFile.listFiles()) {
        getService().rename(cName, file.getName(), path2 + cleanName(file.getName()), file.getContentType());
    }
    // 遍历目录,递归重命名目录和文件
    for (SFile dir : sFile.listDirectories()) {
        renameFiles(cName, path2 + cleanName(dir.getName()) + "/", dir);
    }
}
```

2 文件夹和文件重命名

功能执行测试

选择项目Android的“app”，点击工具条中运行Run “app”，执行的效果如图所示，填入已经注册好的用户名和密码，进行登录验证。

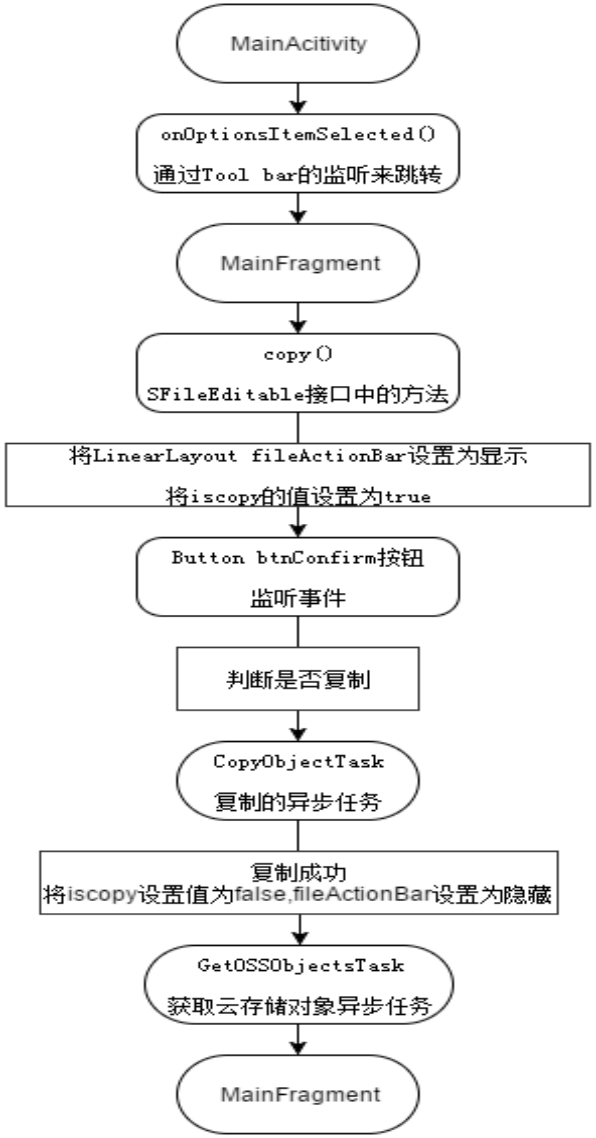




3

复制和黏贴

功能流程图

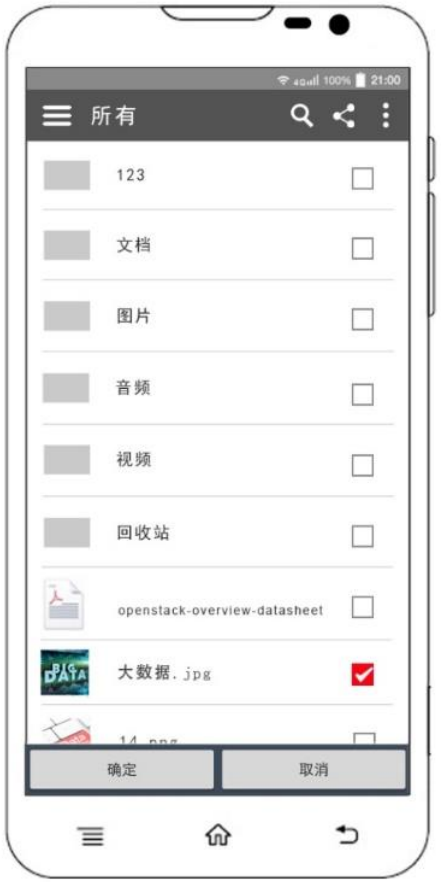


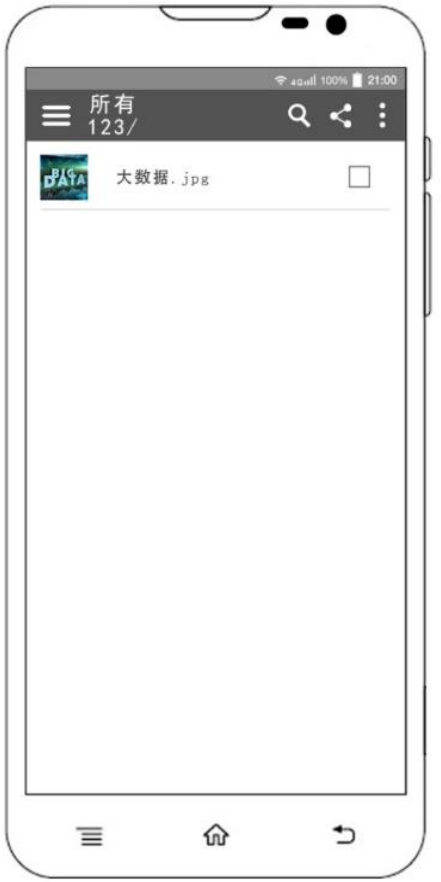
3 复制和黏贴

功能需求

1）视图层，界面实现

根据原型图设计实现复制和黏贴窗口View。





3

复制和黏贴

功能需求

2）控制层，用户在实现复制和粘贴的处理

登入后选中一个文件，点击下拉菜单（menu）中的复制，底部会出现两个按钮（Button），确定与取消，用户，改后点击确定。

3）服务层，用户输入文件夹名后的操作

读取修改完成的文件信息，调用swift服务对文件进行复制和粘贴。

4）控制层，返回结果处理(MVC/MVP)

返回值处理：返回结果的情况包含：

（1）复选中文件后复制到其他目录，文件在当前目录下显示

（2）就在当前文件目录下进行复制黏贴，文件未做任何改变

（3）复制文件夹，提示不可以复制文件夹

3 复制和黏贴

功能需求

1）导入项目

选择File\Open..，点击弹出选择project82目录下面的项目“swiftstorage”

2）界面实现

说先通过android:visibility="gone"把faragment_main.xml下的id为layout_operation_bar的线行布局（LinearLayout）隐藏掉。其中有两按钮（Button），确定与取消。

组件	作用	ID
Button	确认按钮	btnEnter
Button	取消按钮	btnCancel

此xlm文件中所涉及的控件的使用，具体请参考前面项目4Android基础。

3 复制和黏贴

实现步骤

3）在MainActivity中对重命名按钮进行处理

在MainActivity中对下拉菜单中的复制进行监听

```
else if (id == R.id.action_copy) {
    this.currentFragment.copy(null, null);
}
```

调用SFileEditable接口下的copy（）方法

3 复制和黏贴

实现步骤

4）在MainFragment中完成对复制和黏贴的处理

在复制功能中需要新建的变量：

代码说明：

（1）getFirstSelected()获取用户选中的第一个文件信息，前文之中以涉及。

（2）fileActionBar.setVisibility(View.VISIBLE);将隐藏的控件显示出来

```
//是否是复制
private boolean iscopy = false;
//复制的文件名称
String copyFileName = null;
//复制到的文件地址
String copyToFileName = null;
//复制文件的类型
String copyFileType = null;
实现SFileEditable接口下的copy（）的方法
@Override
public void copy(String fromPath, String toPath) {
    if (getFirstSelected() != null) {
        this.iscopy = true;
        this.copyFileName = getFirstSelected().getName();
        this.copyFileType = getFirstSelected().getContentType();
        fileActionBar.setVisibility(View.VISIBLE);
    }
}
```

3 复制和黏贴

实现步骤

之前我们在onCreateView构造视图方法中已经对隐藏好的按钮进行了监听下面我们来完善这两个监听事件。

```
btnConfirm.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (iscopy) {
            //是复制
            copyToFileName =
getAppState().getSelectedDirectory().getName() +
cleanName(copyFileName);
                    CopyObjectTask copyObjectTask = new
CopyObjectTask(copyFileName, copyToFileName, copyFileType);
            copyObjectTask.execute();
        }
    }
});
```

```
btnCancel = (Button) rootView.findViewById(R.id.btnCancel);
btnCancel.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        fileActionBar.setVisibility(View.GONE);
    }
});
```

代码说明：

（1）确定按钮（Button）的监听：如果iscopy为就将copyFileName, copyToFileName, copyFileType这三个参数传到异步任务之中。

（2）取消按钮（Button）的监听，点击后底部的布局隐藏。

（3）调用swift服务完成复制粘贴的工作，并且隐藏底部的布局。

3 复制和黏贴

功能执行测试

选择项目Android的“app”，点击工具条中运行Run “app”，执行的效果如图所示，填入已经注册好的用户名和密码，进行登录验证。