中山大学数据科学与计算机学院本科生实验报告

（2017 年秋季学期）

课程名称：手机应用平台开发　　任课教师： 刘宁

年级	15 级	专业（方向）	软件工程数媒方向
学号	15331016	姓名	陈桂燕
电话	13719346314	Email	735594896@qq.com
开始日期	2017/11/30	完成日期	2017/12/1

一、 实验题目：数据存储（一）

【目的】

1. 学习 SharedPreferences 的基本使用；

2. 学习 Android 中常见的文件操作方法；

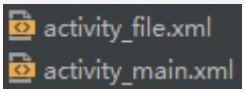
3. 复习 Android 界面编程。

二、 实现内容

实现一个密码输入 activity 和一个文件编辑 activity。具体要求略~

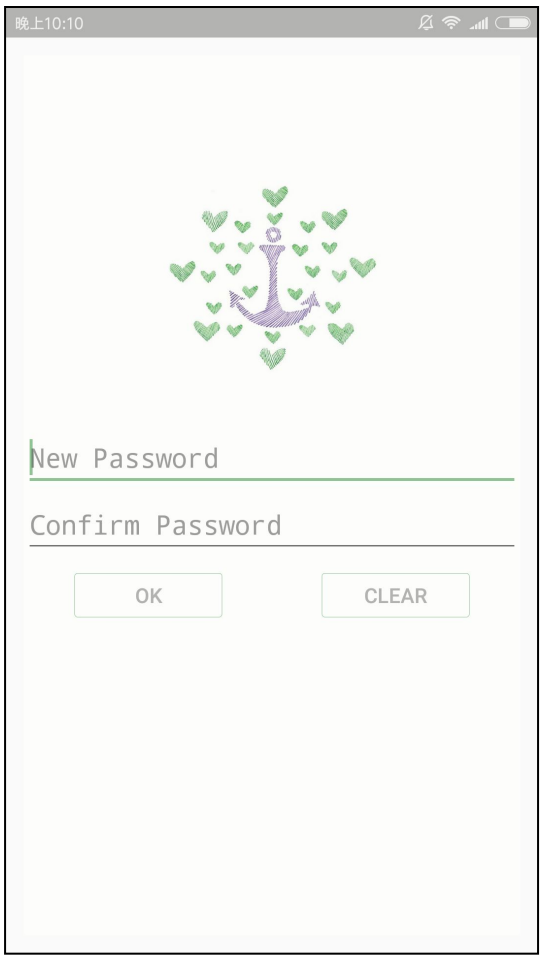
三、 实验过程

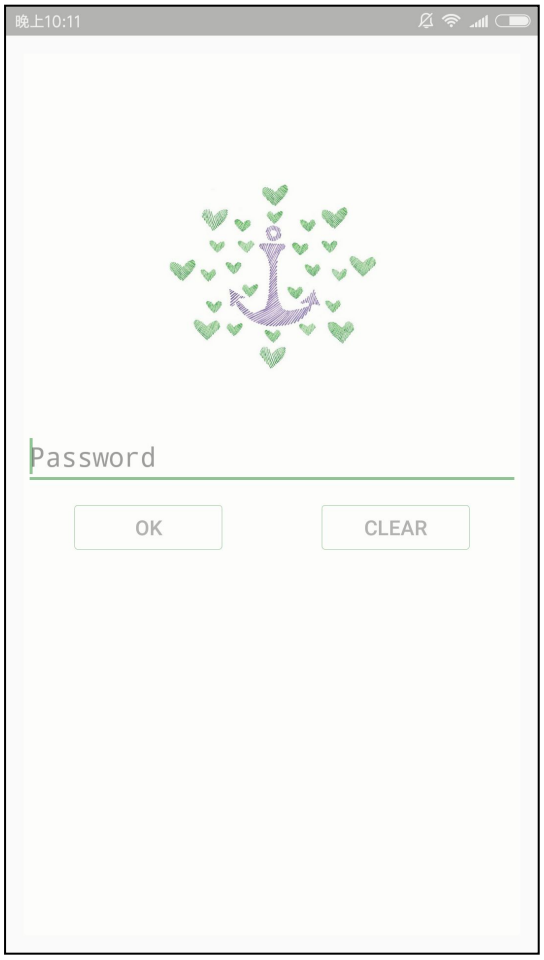
1. 界面设计

实现两个界面： 

一个为密码输入界面 activity_file.xml，如果应用首次启动，则界面呈现出两个输入框，分别为新密码输入和确认密码输入框；完成创建密码后，退出应用再进入应用，则只呈现一个密码输入框。一个为文件编辑界面 activity_file.xml，能对文件进行增删查改的基本操作。

>>密码输入界面 activity_main.xml 展示如下：





如何实现创建密码后，启动界面由左图变为右图呢？我在对应的 MainActivity 里判断密码是否已经存储，若是，则设第一个 EditText 控件不可见，代码如下：

```
private SharedPreferences sp;
```

（密码存储后会写入“rememberPassword”为 true）

```
void init() {
    //判断是否已存入密码，若已存入，则界面改变
    boolean isRemember = sp.getBoolean("rememberPassword",false);
    if (isRemember) {
        NewPassword.setVisibility(View.GONE);
        ConfirmPassword.setHint("Password");
    }
}
```

>>文件编辑界面 activity_file.xml 展示如下：

为了能使第二个 EditText（即 File Content）占据空间变大，在其控件属性里设置如下：

```
android:inputType="textMultiLine"
android:gravity="left|top"
android:minLines="17"
```

（设置显示行数以及光标初始位置）

2. SharedPreferences

按钮点击的应用逻辑不必赘述，详情见代码文件。这里直接切入主题。

>>创建 SharedPreferences

```
private SharedPreferences sp;
```

>>将要存储的内容写入 SharedPreferences

```
} else {//密码不为空且互相匹配，创建成功，转入文件编辑界面
    //存储密码
    SharedPreferences.Editor editor = sp.edit();
    editor.putBoolean("rememberPassword", true);
    editor.putString("Password", psw1);
    editor.commit();
    Intent intent = new Intent(MainActivity.this, FileActivity.class);
    startActivity(intent);
}
```

>>从 SharedPreferences 中读取所要的内容

```
//判断是否已存入密码，若已存入，则界面改变
boolean isRemember = sp.getBoolean("rememberPassword",false);
//首先判断是否已经存入密码，从而设置不同的点击事件
String psw = sp.getString("Password","");
```

3. 文件操作 Internal Storage

向 Internal Storage 写入文件（调用 write 方法）：

```
save.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
        String text1 = topic.getText().toString();
        String text2 = content.getText().toString();
        if (text1.equals("")) {
            Toast.makeText(getApplicationContext(),"Fail to save file",Toast.LENGTH_SHORT).show();
        } else {
            String FILE_NAME = text1+".txt";
            try (FileOutputStream fileOutputStream = openFileOutput(FILE_NAME, MODE_PRIVATE)) {
                fileOutputStream.write(text2.getBytes());
                fileOutputStream.flush();
                fileOutputStream.close();
                Toast.makeText(getApplicationContext(),"Save successfully",Toast.LENGTH_SHORT).show();
            } catch (IOException e) {
                Toast.makeText(getApplicationContext(),"Fail to save file",Toast.LENGTH_SHORT).show();
            }
        }
    }
});
```

读取文件里的内容（调用 read 方法）：

```
load.setOnClickListener((OnClickListener) (view) -> {
        String text1 = topic.getText().toString();
        if (text1.equals("")) {
            content.setText("");
            Toast.makeText(getApplicationContext(),"Fail to load file",Toast.LENGTH_SHORT).show();
        } else {
            String FILE_NAME = text1+".txt";
            try (FileInputStream fileInputStream = openFileInput(FILE_NAME)) {
                byte[] contents = new byte[fileInputStream.available()];
                fileInputStream.read(contents);
                String text = new String(contents);
                content.setText(text);
                Toast.makeText(getApplicationContext(),"Load successfully",Toast.LENGTH_SHORT).show();
                fileInputStream.close();
            } catch (IOException e) {
                content.setText("");
                Toast.makeText(getApplicationContext(),"Fail to load file",Toast.LENGTH_SHORT).show();
            }
        }
});
```

删除文件：

```
delete.setOnClickListener((view) -> {
        String text1 = topic.getText().toString();
        if (text1.equals("")) {
            Toast.makeText(getApplicationContext(),"Fail to delete file",Toast.LENGTH_SHORT).show();
        } else {
            String FILE_NAME = text1+".txt";
            boolean flag = deleteFile(FILE_NAME);
            if (flag) {
                Toast.makeText(getApplicationContext(),"Delete successfully",Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(getApplicationContext(),"Fail to delete file",Toast.LENGTH_SHORT).show();
            }
        }
});
```

4. 特殊要求

要求进入文件编辑的 Activity 之后，如果点击返回按钮，则直接返回 Home 界面，不再返回密码输入界面。要使 Activity 不可见时，即将其从 activity stack 中除去。
由于是在密码输入界面 MainActivity 里通过 intent 进入文件编辑界面的，因此，我们可以在 AndroidManifest.xml 中设置 noHistory 属性，如下。

```
<activity android:name=".MainActivity"
    android:noHistory="true">
```

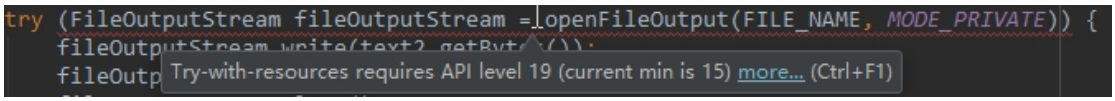
5. 阐述：Internal Storage 和 External Storage 的区别以及他们分别适用的场景

- Internal Storage ：内部存储器，数据私有。
- External Storage ：外部存储器，数据共有。

Internal Storage 把数据存储在设备内部存储器上，默认情况下在这里存储的数据为应用程序的私有数据，其它应用程序不能访问，文件管理器也寻不到。卸载 app 后，内部存储器的/data/data/<package name>目录及其下子目录和文件一同被删除。因此，Internal Storage 适用于“想确保文件数据不被用户和其他 app 所访问”的场景。
而 External Storage 是将数据作为 USB 存储模式，所有程序和用户都可以访问，文件管理器也可以找到。卸载 app 后，系统系统仅会删除 external 根目录（getExternalFilesDir）下的相关文件。（另外，使用 External Storage 前要进行权限申请，这样才可以对 SD 卡进行读取操作。）因此，External Storage 适用于“不需要严格的访问权限并且希望这些文件数据能被其他 app 所共享或者是允许用户通过文件管理器进行访问”的场景。

四、 实验思考及感想

（1）说几个实验遇到的小错误：
① 这处地方一直显示红波浪线，解决不了。但是代码能正常运行……



② 在进行文件删除时，最开始百度到的是 file.delete()，使用后一直无法成功删除，没有报错但也无 boolean 返回值，后来改成 deleteFile(FILE_NAME)才运行正确。

（2）数据存储是特别特别实用的功能，可以利用该功能来开发自己专属的备忘录&日记 app。感受到了学这门课的意义。