中山大学移动信息工程学院本科生实验报告

（2017 年秋季学期）

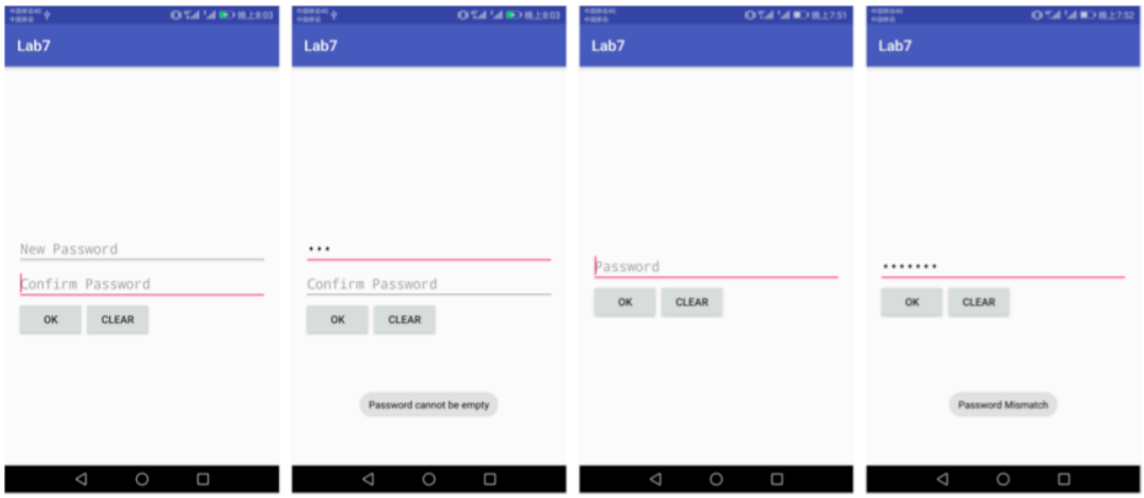
课程名称：移动应用开发　　任课教师： 郑贵锋

年级	2015 级	专业（方向）	移动信息工程（互联网方向）
学号	1535251	姓名	柯博
电话	13671412922	Email	kebo@mail2.sysu.edu.cn
开始日期		完成日期	

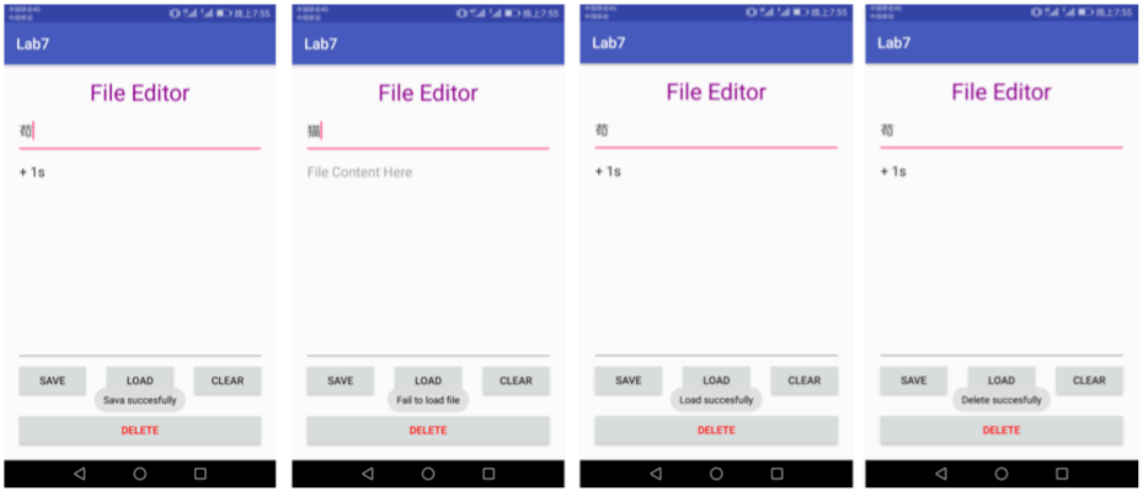
一、 实验题目

实验七：数据存储（一）

二、 实现内容



从左至右，依次为：初始密码界面、密码为空提示、密码匹配后重新进入界面、密码错误提示。



从左至右，依次为：保存成功提示、写入失败提示、写入成功提示、删除成功提示。

1、 如图所示，本次实验需要实现两个 activity;

2、 首先，需要实现一个密码输入 activity:

a、 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; b、

b、输入框下方有两个按钮:

- OK 按钮，点击之后:
  - 若 new password 为空，则弹出密码为空的提示;
  - 若 new password 与 comfirm password 不匹配，则弹出不匹配的提示; - 若密码不为空且互相匹配，则保存密码，进入文件编辑界面。
- CLEAR 按钮，点击之后清除所有输入框的内容。

c、 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;

- 点击 OK 按钮后，如果输入的密码与保存的密码不匹配，则弹出 Toast 提示;
- 点击 CLEAR 按钮后，清除密码输入框的内容。

d、出于学习的目的，我们使用 SharedPreferences 来保存密码，但是在实际应用中我们会用更加 安全的机制来保存这些隐私信息，更多可以参考开发文档。

3、 然后，实现一个文件编辑 activity:

a、 界面底部有两行四个按钮，第一行三个按钮高度一致，顶对齐，按钮水平均匀分布。按钮上方除 了 ActionBar 和 StatusBar 之外的空间由标题和两个 EditText 占据，文件内容编辑的 EditText 需 要占据除去其他控件的全部屏幕空间，且内部文字竖直方向置顶，左对齐;

b、 在文件名输入框内输入文件名，在文件内容编辑区域输入任意内容，点击 SAVE 按钮后能够保存 到指定文件，成功保存后弹出 Toast 提示;

c、 点击 CLEAR 按钮，能够清空文件内容编辑区域内的内容;

d、点击 LOAD 按钮，能够按照文件名从内存中读取文件内容，并将文件内容写入到编辑框中。如 果成功导入，则弹出成功的 Toast 提示，如果导入失败(例如:文件不存在)，则弹出读取失败 的 Toast 提示。

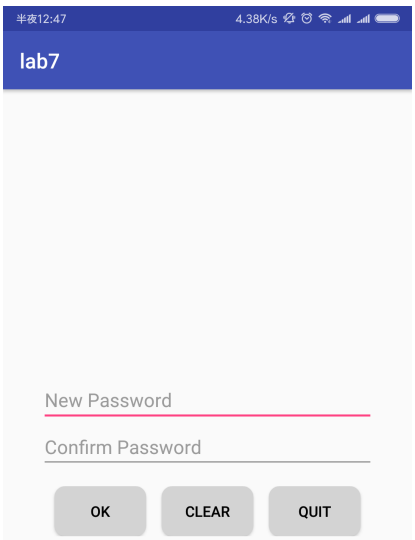
e、 点击 DELETE 按钮，能够按照文件名从内容中删除文件，删除文件后再载入文件，弹出导入失败 的 Toast 提示。

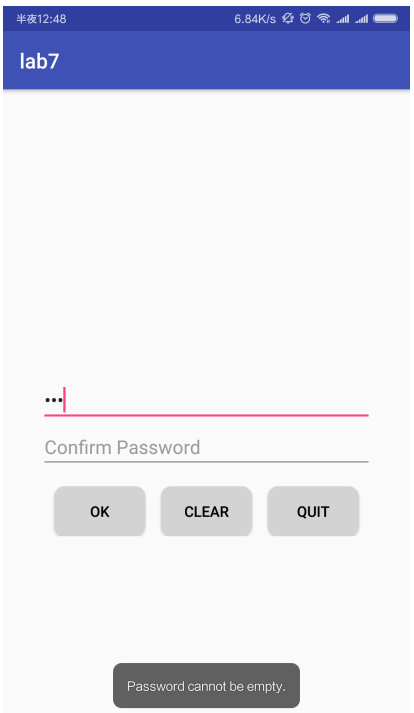
4、 特殊要求:进入文件编辑的 Activity 之后，如果点击返回按钮，则直接返回 Home 界面，不再返回 密码输入界面。

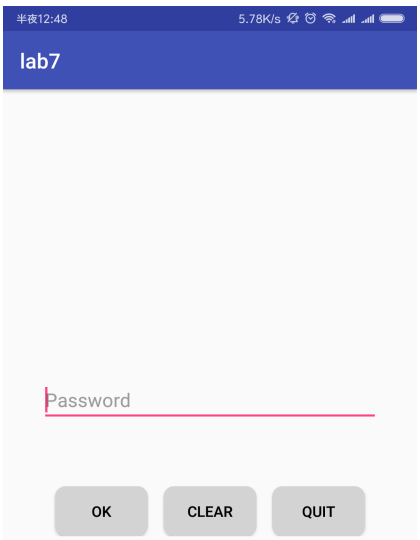
三、 课堂实验结果

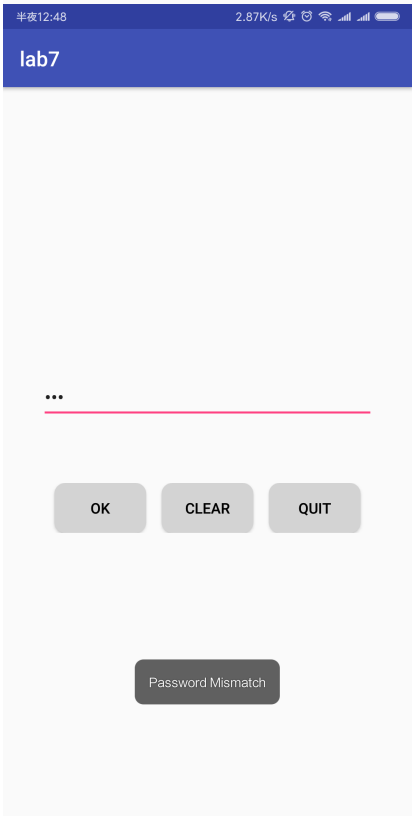
（1） 实验截图

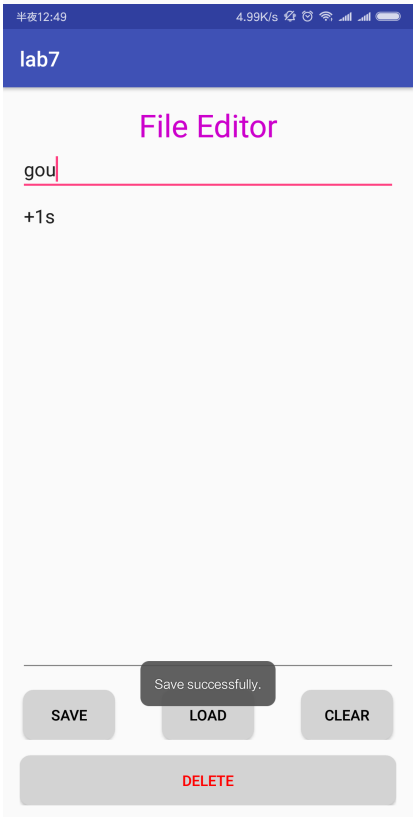
截图顺序与实现内容中的一致。

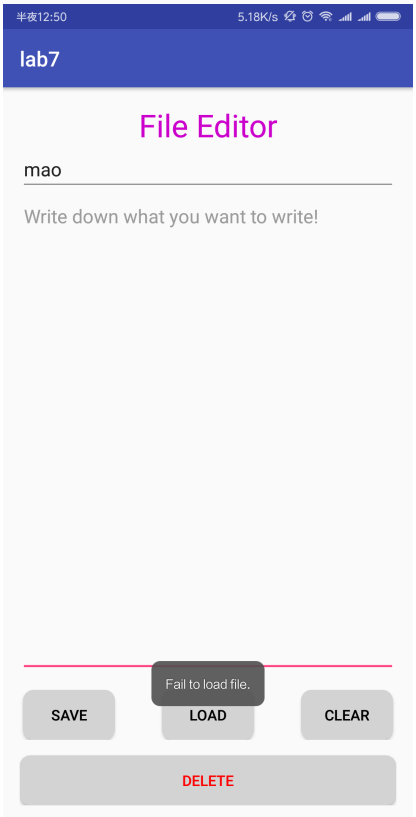

半夜12:47
4.38K/s
lab7
New Password
Confirm Password
OK
CLEAR
QUIT

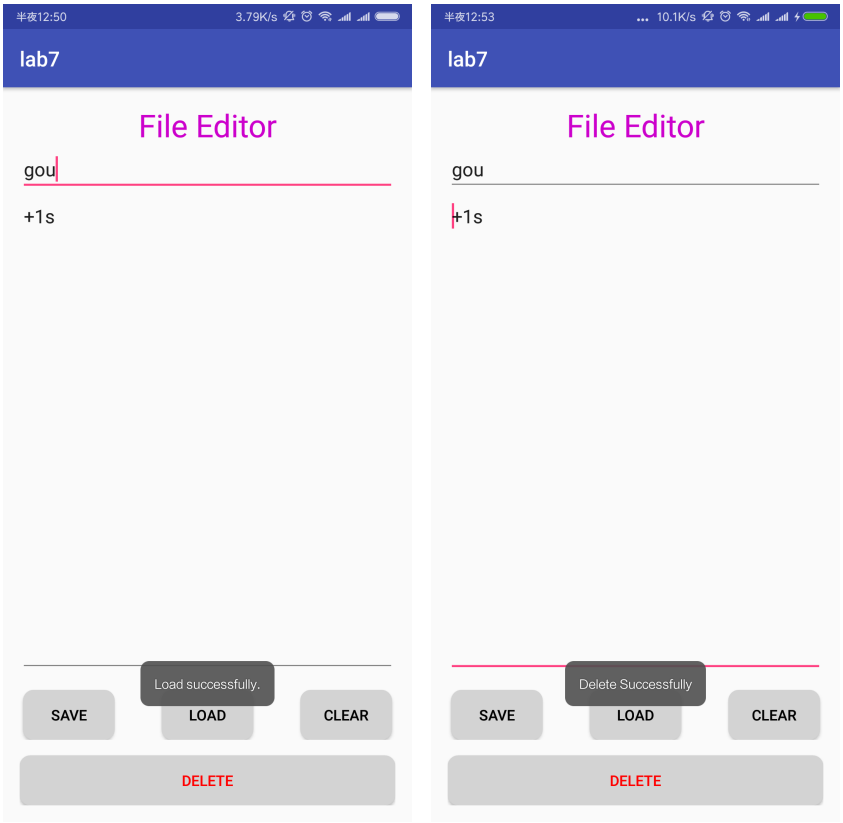

半夜12:48
6.84K/s
lab7
Confirm Password
OK
CLEAR
QUIT
Password cannot be empty.


半夜12:48
5.78K/s
lab7
Password
OK
CLEAR
QUIT


半夜12:48
2.87K/s
lab7
OK
CLEAR
QUIT
Password Mismatch


半夜12:49
4.99K/s
lab7
File Editor
gou
+1s
Save successfully.
SAVE
LOAD
CLEAR
DELETE


半夜12:50
5.18K/s
lab7
File Editor
mao
Write down what you want to write!
Fail to load file.
SAVE
LOAD
CLEAR
DELETE



（2） 实验步骤以及关键代码

activity_main.xml 这次实验采用了线性布局，从上到下的控件分别是密码框 1、密码框 2、排在同一行的 OK、CLEAR、QUIT 三个按钮：

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center"
    android:gravity="center_vertical"
    android:orientation="vertical"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp">


    <EditText...>

    <EditText...>


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:gravity="center_horizontal"
        android:orientation="horizontal">

        <Button...>

        <Button...>

        <Button...>
    </LinearLayout>

</LinearLayout>
```

activity_edit.xml 同样采用线性布局，自上而下的布局元素分别为：标题、文件名、文件内容、排在同一行的 Save、Load、Clear 按钮和最下方的 Delete 按钮：

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center"
    android:gravity="center_vertical"
    android:orientation="vertical"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp">

    <TextView...>

    <EditText...>

    <EditText...>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15dp"
        android:gravity="center_horizontal"
        android:orientation="horizontal">

        <Button...>

        <Button...>

        <Button...>
    </LinearLayout>

    <Button
        android:id="@+id/BtnDel"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@drawable/shape"
        android:layout_marginTop="15dp"
        android:text="DELETE"
        android:textColor="@color/colorRed"/>

</LinearLayout>
```

MainActivity.java 中 OnCreate 函数，先执行 readAccount 函数，再设置三个按钮的点击事件：

```
//读取保存在本地的用户名和密码
public void readAccount() {

    //创建SharedPreferences对象
    SharedPreferences sp = getSharedPreferences( name: "info", MODE_PRIVATE);

    //获得保存在SharedPredPreferences中的用户名和密码
    String password1 = sp.getString( key: "password1",  defValue: "");
    String password2 = sp.getString( key: "password2",  defValue: "");

    //在用户名和密码的输入框中显示用户名和密码
    et_password.setText(password1);
    et_password2.setText(password2);

    if(!"".equals(password2)){
        et_password2.setVisibility(View.INVISIBLE);
        et_password.setHint("Password");
        et_password.setText("");
        tag = false;
    }

}
```

```
ok.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //获得用户输入的用户名和密码
        String password1 = et_password.getText().toString();
        String password2 = et_password2.getText().toString();
        //创建sharedPreference对象，info表示文件名，MODE_PRIVATE表示访问权限为私有的
        SharedPreferences sp = getSharedPreferences( name: "info", MODE_PRIVATE);
        //获得sp的编辑器
        SharedPreferences.Editor ed = sp.edit();
        //以键值对的显示将用户名和密码保存到sp中
        ed.putString("password1", password1);
        ed.putString("password2", password2);
        //提交用户名和密码
        ed.commit();
        if ("".equals(password1)||"".equals(password2)){
            Toast.makeText( context: MainActivity.this,  text: "Password cannot be empty.", Toast.LENGTH_LONG).show();
            return ;
        }
        if (!password1.equals(password2)) {
            Toast.makeText( context: MainActivity.this,  text: "Password Mismatch", Toast.LENGTH_LONG).show();
            return ;
        }
        if(password1.equals(password2)) {
            Intent intent = new Intent();
            intent.setClass( packageContext: MainActivity.this, FileEditorActivity.class);
            MainActivity.this.startActivity(intent);
        }
    }
});

clear1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (tag == true) {
            et_password.setText("");
            et_password2.setText("");
        } else {
            et_password.setText("");
        }
    }
});

quit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try {
            MainActivity.this.finish();
        } catch (Exception e) {

        }
    }
});
```

AndroidManifest.xml 文件中修改 MainActivity 的存活模式，并声明 FileEditActivity：

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.kebo.lab7">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="lab7"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity" android:noHistory="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".FileEditorActivity"/>
    </application>

</manifest>
```

FileEditActivity.java 中 OnCreate 函数，分别设置四个按钮的点击事件，其中包含文件操作：

```
save.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try (FileOutputStream fileOutputStream = openFileOutput(et1.getText().toString(), MODE_PRIVATE)) {
            String str = "";
            str = et2.getText().toString();
            fileOutputStream.write(str.getBytes());
            Toast.makeText( context: FileEditorActivity.this,  text: "Save successfully.", Toast.LENGTH_LONG).show();
            Log.i( tag: "TAG",  msg: "Successfully saved file.");
            return ;
        } catch (IOException ex) {
            Log.e( tag: "TAG",  msg: "Fail to save file.");
        }
    }
});

load.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try (FileInputStream fileInputStream = openFileInput(et1.getText().toString())) {
            byte[] contents = new byte[fileInputStream.available()];

            fileInputStream.read(contents);
            et2.setText(new String(contents));
            Toast.makeText( context: FileEditorActivity.this,  text: "Load successfully.", Toast.LENGTH_LONG).show();
            return ;
        } catch (IOException ex) {
            Log.e( tag: "TAG",  msg: "Fail to read file.");
            Toast.makeText( context: FileEditorActivity.this,  text: "Fail to load file.", Toast.LENGTH_LONG).show();
            return ;
        }
    }
});

clear2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        et2.setText("");
    }
});

del.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (getApplicationContext().deleteFile(et1.getText().toString())){
            Toast.makeText( context: FileEditorActivity.this,  text: "Delete Successfully",Toast.LENGTH_LONG).show();
        }else{
            Toast.makeText( context: FileEditorActivity.this,  text: "Fail to Delete", Toast.LENGTH_LONG).show();
        }
    }
});
```

（3） 实验遇到困难以及解决思路

本次实验内容相对简单，没有遇到什么困难，唯一卡住的地方在于删除按钮的点击事件。一开始不知道应该使用哪种方法去删除文件，使用 File.delete( )方法发现不可行，又百度了好久文件流的内置函数都没有删除操作，转念一想文件流是一种文件 IO 的通道，不可能会有删除方法。查看文档发现可以使用 Context.deleteFile( )方法来操作。

四、 课后实验结果

无

五、 实验思考及感想

经过本次实验，对于 Android 系统中存储的管理有了新的认识。Android 数据存储分为 Internal Storage 和 External Storage 两种，前者是将数据存储在 AppData 中 Application 的对应目录下，只能由应用自己访问，对于整个系统而言，是某个应用的私有成员；而后者是将数据存放到外存载体（如 SD 卡、闪存）中，这是一个公有的目录，任何应用都可以读取外置存储中的数据。现在很多手机都只有一块存储空间（闪存）也不再支持 SD 卡，Android 系统文件和用户文件存放在同一根目录下，文件浏览器也可以浏览 AppData 中的文件，但是 Internal Storage 和 External Storage 的区别仍然存在，文件浏览器可以查看到 AppData 中存在应用 Cache 文件，但是它不能对文件进行 CRUD 操作，即 Cache 文件仍然是应用不公开的私有数据。

作业要求：

1. 命名要求: 学号_姓名_实验编号，例如 15330000_林 XX_lab1。
2. 实验报告提交格式为 pdf。
3. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。