# 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：

| | | | |
|---|---|---|---|
| 年级 | 15 级 | 专业（方向） | 软件工程（移动信息工程） |
| 学号 | 15352223 | 姓名 | 刘朝开 |
| 电话 | 15626489095 | Email | 1209354383@qq.com |
| 开始日期 | 2017.9.30 | 完成日期 | 2017.10.1 |

## 一、 实验题目

数据存储（一）

## 二、 实现内容

1、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、 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，更多可以参考链接一和链接二。

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；

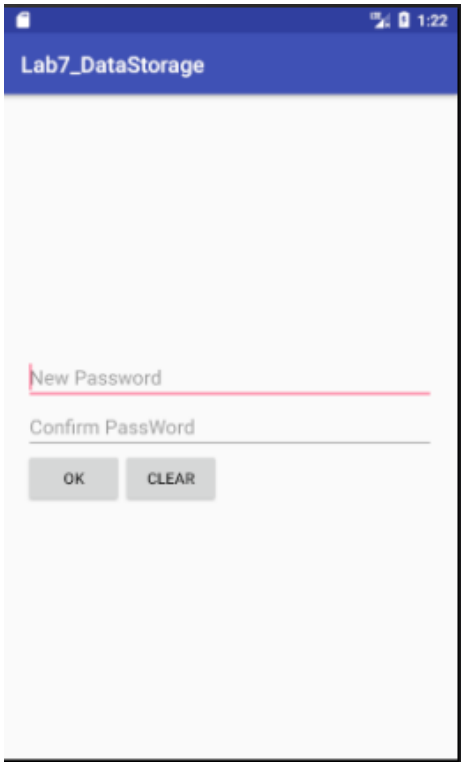
d、 点击 LOAD 按钮，能够按照文件名从内存中读取文件内容，并将文件内容写入到编辑框中。如果成功导入，则弹出成功的 Toast 提示，如果导入失败（例如：文件不存在），则弹出读取失败的 Toast 提示。

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。

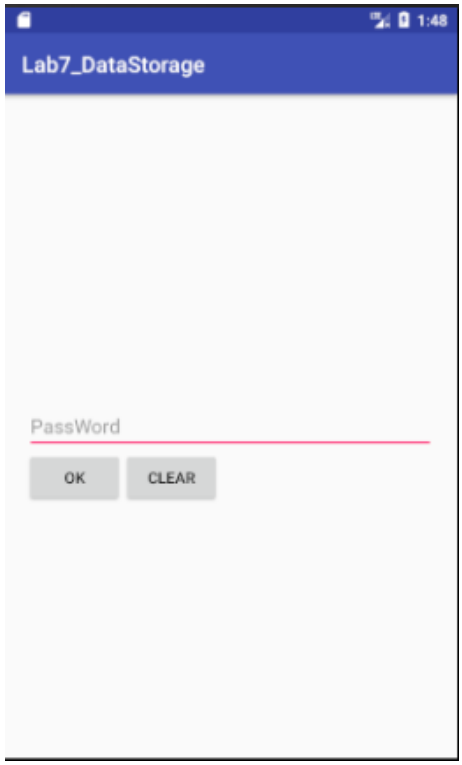
## 三、 课堂实验结果

（1） 实验截图




初始密码界面

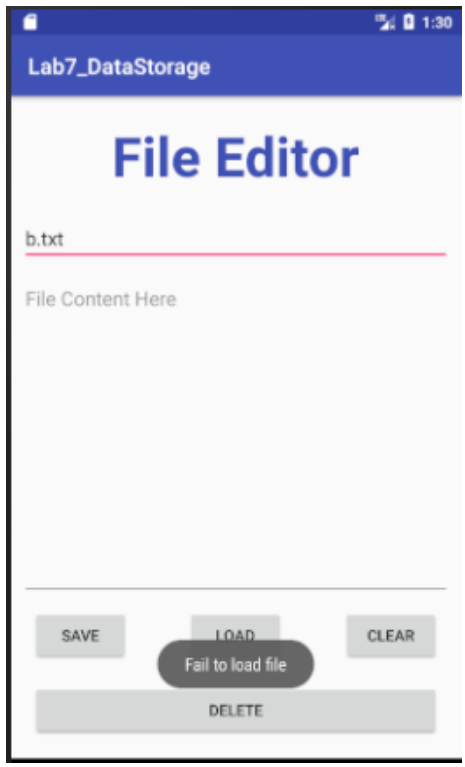

密码为空提示




密码匹配后重新进入界面


密码错误提示


保存成功提示




写入失败提示


写入成功提示


删除成功提示

（2） 实验步骤以及关键代码

步骤 1：新建 project,配置 activity_main 的布局文件；新建 EditActivity，配置 activity_edit 的布局文件。

**在 MainActivity 中：**

步骤 2：实例化布局控件：

```
void initView()//初始化实例
{
    OK = (Button) findViewById(R.id.ok);
    Clear = (Button) findViewById(R.id.clear);
    editTexts[0] = (EditText) findViewById(R.id.Npassword);
    editTexts[1] = (EditText) findViewById(R.id.Cpassword);
    editTexts[2] = (EditText) findViewById(R.id.password);
}
```

步骤 3：通过变量 isFirstin 记录 app 是否第一次设置密码来设置界面；

```
void setUI()//设置界面
{
    if(isFirstin) {
        editTexts[0].setVisibility(View.VISIBLE);
        editTexts[1].setVisibility(View.VISIBLE);
        editTexts[2].setVisibility(View.INVISIBLE);
    }
    else {
        editTexts[0].setVisibility(View.INVISIBLE);
        editTexts[1].setVisibility(View.INVISIBLE);
        editTexts[2].setVisibility(View.VISIBLE);
    }
}
```

步骤 4：设置点击事件：

```
case R.id.ok:
    if(isFirstin) {
        Npassword = editTexts[0].getText().toString();
        Cpassword = editTexts[1].getText().toString();
        if(Npassword.equals("") || Cpassword.equals("")) {
            Toast.makeText( context: MainActivity.this, text: "Password cannot be empty",Toast.LENGTH_SHORT).show();
        }
        else if(!Npassword.equals(Cpassword)) {
            Toast.makeText( context: MainActivity.this, text: "Password Mismatch",Toast.LENGTH_SHORT).show();
        }
        else {
            isFirstin = false;
            setUI();
            SharedPreferences.Editor editor = getSharedPreferences( name: "data",MODE_PRIVATE).edit();
            editor.putBoolean( s: "isFirstin", b: false);
            editor.putString( s: "Npassword",Npassword);
            editor.apply();//提交，将isFirstIn设为false
        }
    }
    else
    {
        password = editTexts[2].getText().toString();
        if(password.equals(Npassword)){
            Intent intent = new Intent( packageContext: MainActivity.this,EditActivity.class);
            startActivity(intent);
            finish();
        }
        else {
            Toast.makeText( context: MainActivity.this, text: "Password Mismatch",Toast.LENGTH_SHORT).show();
        }
    }
    break;
case R.id.clear:
    if(isFirstin){
        editTexts[0].setText("");
        editTexts[1].setText("");
    }
    else {
        editTexts[2].setText("");
    }
    break;
default:break;
```

步骤 5：在 onCreat 中获得 isFirstin 和之前设置的密码：

```
SharedPreferences preferences = getSharedPreferences( name: "data",MODE_PRIVATE);
isFirstin = preferences.getBoolean( s: "isFirstin", b: true);
setUI();
Npassword = preferences.getString( s: "Npassword", s1: "");
```

**在 EditActivity 中：**

步骤 6：实例化布局控件：

```
private void initView()//初始化实例
{
    buttons[0] = (Button) findViewById(R.id.save);
    buttons[1] = (Button) findViewById(R.id.load);
    buttons[2] = (Button) findViewById(R.id.clear);
    buttons[3] = (Button) findViewById(R.id.delete);
    title_editText = (EditText) findViewById(R.id.edit_title);
    content_editText = (EditText) findViewById(R.id.edit_content);
}
```

步骤 7：定义保存文件函数：传入参数包括文件名和文件内容。

```
private void save(String fileName, String fileContent)//保存文件，传入文件名和文件内容
{
    FileOutputStream out = null;
    BufferedWriter writer = null;
    try {
        out = openFileOutput(fileName, Context.MODE_PRIVATE);
        writer = new BufferedWriter(new OutputStreamWriter(out));
        writer.write(fileContent);
        Toast.makeText( context: EditActivity.this, text: "Save successfully",Toast.LENGTH_SHORT).show();
    } catch (IOException e) {
        e.printStackTrace();
        Toast.makeText( context: EditActivity.this, text: "Fail to save file",Toast.LENGTH_SHORT).show();

    }finally {
        try {
            if(writer!=null) {
                writer.close();
            }
        }catch (IOException e){
            e.printStackTrace();
        }
    }
}
```

步骤 8：定义导入函数，传入参数为文件名，返回值为文件内容

```
private String load(String fileName)//加载文件，传入文件名，返回文件内容
{
    FileInputStream in = null;
    BufferedReader reader = null;
    StringBuilder content = new StringBuilder();
    try {
        in = openFileInput(fileName);
        reader = new BufferedReader(new InputStreamReader(in));
        String line = "";
        while ((line = reader.readLine())!=null){
            content.append(line);
        }
        Toast.makeText( context: EditActivity.this, text: "Load successfully",Toast.LENGTH_SHORT).show();
    }catch (IOException e){
        e.printStackTrace();
        Toast.makeText( context: EditActivity.this, text: "Fail to load file",Toast.LENGTH_SHORT).show();
    }finally {
        if(reader != null){
            try{
                reader.close();
            }catch (IOException e){
                e.printStackTrace();
            }
        }
    }
    return content.toString();
}
```

步骤 9：实现按钮点击事件

```
switch (view.getId())
{
    case R.id.save:
        fileName = title_editText.getText().toString();
        if (fileName.contains("/")) {
            Toast.makeText( context: this,  text: "Invilid file name!", Toast.LENGTH_SHORT).show();
        }
        else {
            fileContent = content_editText.getText().toString();
            save(fileName, fileContent);
        }
        break;
    case R.id.load:
        fileName = title_editText.getText().toString();
        fileContent=load(fileName);
        content_editText.setText(fileContent);
        break;
    case R.id.clear:
        content_editText.setText("");
        break;
    case R.id.delete:
        fileName = title_editText.getText().toString();
        boolean flag = deleteFile(fileName);
        if(flag) Toast.makeText( context: EditActivity.this, text: "Delete successfully",Toast.LENGTH_SHORT).show();
        else Toast.makeText( context: this,  text: "There is no such file", Toast.LENGTH_SHORT).show();
        break;
    default:break;
```

（3） 实验遇到困难以及解决思路

困难：实验中没有对文件名中的分隔符进行判断，文件保存出错。

解决方法：判断输入的文件名是否含有分隔符，若含有，则弹出错误信息，不允许保存。

代码如下:

```
if (fileName.contains("/")) {
    Toast.makeText( context: this,  text: "Invilid file name!", Toast.LENGTH_SHORT).show();
}
```

## 四、 课后实验结果

## 五、 实验思考及感想

实验思考：

1 如何使 EditText 占据指定大小的空间？

使用 LinearLayout 和 layout_weight 属性。

2 当 Activity 不可见时，如何将其从 activity stack 中除去（按返回键直接返回 Home）？

1)在 AndroidManifest.xml 中设置 noHistory 属性。

2)在打开下一个 activity 时调用 finsh()函数。

3 如何根据需要隐藏/显示特定的控件？

如下代码：设置控件属性可见和不可见。

```
editTexts[0].setVisibility(View.INVISIBLE);
editTexts[1].setVisibility(View.INVISIBLE);
editTexts[2].setVisibility(View.VISIBLE);
```

4 Internal Storage 和 External Storage 的区别以及他们分别适用的场景。

Internal Storage ：把数据存储在设备内部存储器上，存储在/data/data/<package name>/files 目录下。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。卸载应用程序后，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。它是属于应用程序的，文件管理器看不见，只有这个应用本身可以看到和使用。

External Storage：一个应用把数据存在 external storage 上时,那么数据成为共有的,所有人都可见的和可用的。

感想：通过这次实验，简单回顾了一下布局的设计，学习和使用了 android 的内部存储的使用，包括写入文件，读取文件和删除文件。这次实验难度较低，实验过程也并不复杂，所以做起来也比较轻松。自己主要上网查阅了一些资料，做了一些简单的总结。

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。