手机平台应用开发实验报告

学号: 14331098　　**班级:** 周五下午班

姓名: 黄建武　　**实验名称:** Lab7

一、参考资料

1. Android官方开发者文档
2. Android - 文件读写操作 总结

二、实验环境

- 系统: Windows 10 64位
- 工具: Android Studio 2.1.3
- JDK版本: 1.8
- minSdk: 19
- targetSdk: 24

三、实验目的

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。

四、实验步骤

1. 阅读实验要求文档以及对应的课件PPT。
2. 运行Demo程序，了解实验要求细节。
3. 首先完成创建密码界面，页面整体使用一个垂直布局LinearLayout，下面的OK和CLEAR按钮是一个水平布局的LinearLayout。代码的整体结构如下:

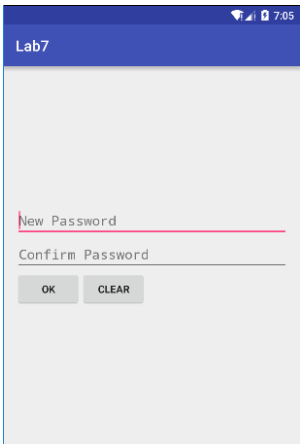
```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="ml.huangjw.lab7.MainActivity">

    <EditText...>

    <EditText...>

    <LinearLayout...>
</LinearLayout>
```

4. 实现创建密码页面的事件逻辑，判断密码合法性，保存密码，清除文本，显示Toast等。使用SharedPreferences存储密码以及使用一个布尔变量记录是否创建了密码，当进入应用时从SharedPreferences中读取该布尔变量，如果已经创建了密码，则隐藏第一个文本框。 5. 实现输入密码界面的布局，页面整体使用一个垂直布局的LinearLayout，下方的三个按钮使用一个RelativeLayout进行定位，这里将文本框的layout_weight设为1，RelativeLayout的layout_weight设为0，使文本框占据所以剩余空间。代码的整体结构如下:

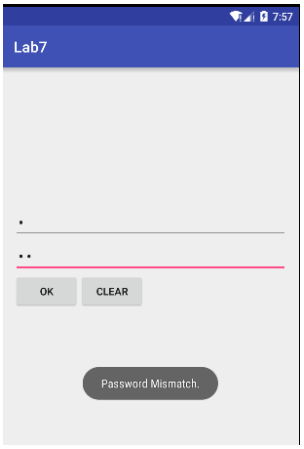
```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="ml.huangjw.lab7.MainActivity">

    <EditText...>

    <EditText...>

    <LinearLayout...>
</LinearLayout>
```

6. 实现输入密码页面的事件逻辑，清除，导入文件内容，写入文件等操作，使用openFileOutput()和openFileInput()读写文件，操作结束后关闭文件。 7. 进行调试和完善。

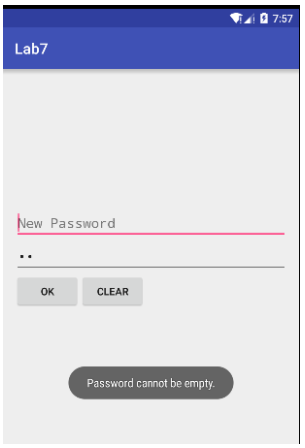
五、实验结果



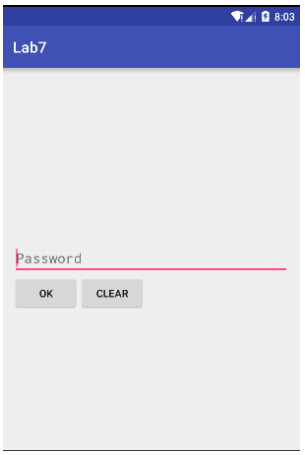
首次进入创建密码界面



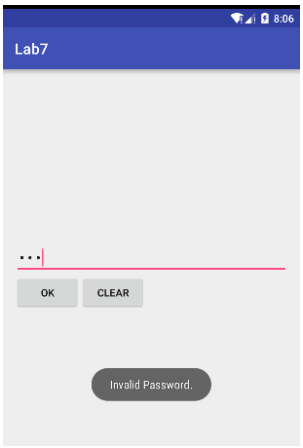
密码不匹配



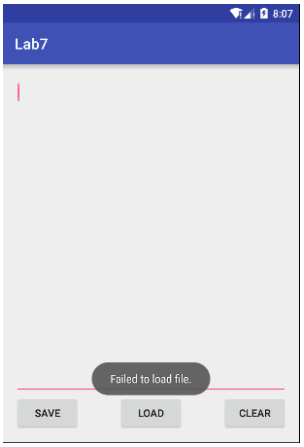
密码为空



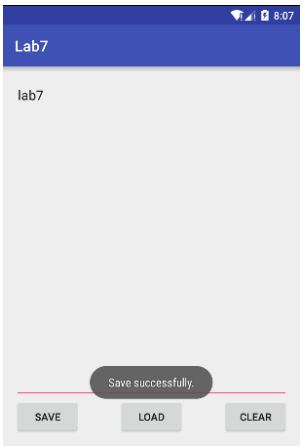
第二次进入创建密码页面



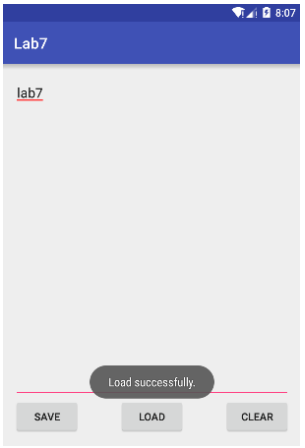
密码不正确(密码为$test$)



文件加载失败



成功保存文件



成功导入文件

六、实验问题

1. java.io.FileNotFoundException: Lab7.txt: open failed: EROFS (Read-only file system)
   将数据写入文件中的时候提示文件为只读，查阅资料后才发现我使用的是

```
FileOutputStream fos = new FileOutputStream(FILENAME);
```

   获取一个FileOutStream然后对文件进行写操作,而使用该种方法获取的对象是只读属性的，要进行写操作，需要改用下面的方法

```
FileOutputStream fos = openFileOutput(FILENAME, MODE_PRIVATE);
```

2. EditText遇到问题
   实现第二个页面的文本框的布局时，一开始设置了各个控件的layout_weight后，在文本框中输入文字还是在文本框中间，没有从左上角开始，一开始想使用textAlignment，想想发现可以使用android:gravity="top|start"属性，设置完成后文字成功从左上角开始，不过发现了demo中不好的一点，当文本框中内容超出屏幕时文字并不会自动换行，查阅资料后得知EditText默认是singleLine，想要自动换行需要设置属性android:inputType="textMultiLine"

3. 载入文件时出现乱码
   载入文件时一开始是将读入文件获取到的Byte数组直接调用toString函数转为字符串然后赋值给文本框，但是文本框显示的是乱码，应该是不能直接从Byte数组转化为String，查看String的构造函数可知有一个构造函数是结束一个Byte数组，修改后具体代码如下：

```
try (FileInputStream fis = openFileInput(FILENAME)) {
    byte[] contents = new byte[fis.available()];
    fis.read(contents);
    fis.close();
    et.setText(new String(contents));
    showToast(R.string.success1);
} catch (IOException ex) {
    showToast(R.string.error4);
    ex.printStackTrace();
}
```

七、实验总结

本次实验难度较低，通过该实验，了解和掌握了Android中简单的数据存储和文件读写，少量和不涉及隐私的数据可以使用SharedPreferences进行存储，操作方便，而一些重要的信息如用户信息等可以写入文件，读写文件时使用openFileOutput()和openFileInput()函数对文件进行操作。

在Android中，Internal Storage是把数据存储在设备内部存储器上，存储在$/data/data/ < package\ name >$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，文件管理器中也看不到。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。Internal Storage空间十分有限，不适合用来存储大文件，可用来存储应用主要的数据。而External storage在文件浏览器里是可以看见，位于/mnt目录下，需要先在Androidmainfest.xml文件中注册新建删除和读写的权限，数据存在External storage时，数据成为共有的，所有人都可见的和可用的，External storage并不只表示SD卡，很多没有插SD卡的设备，系统会虚拟出一部分存储空间用来做公共存储。External storage多用来存储一些公开的，体积较大的文件，如多媒体文件。

本次实验也暴露出一些问题，对界面编程的知识掌握不牢，有些许遗忘，在实现第二个页面的文本框占据剩余所有空间以及文字左上对齐时花了较长时间，需要吸取教训。