

Lab7 实验报告

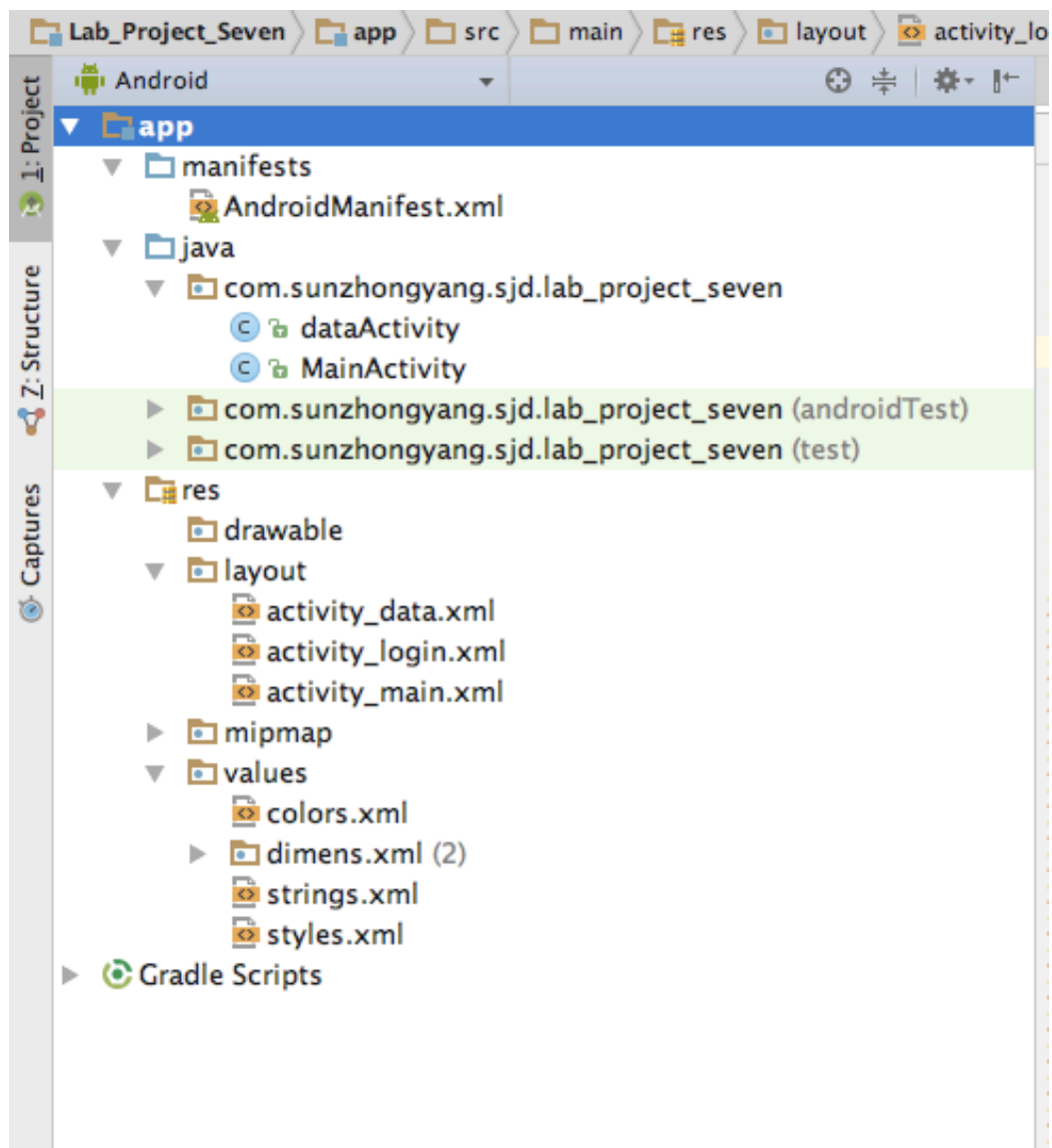
13331233 孙中阳

参考资料

《第一行代码 Android》 郭霖 著
《Lab7实验文档》

实验步骤

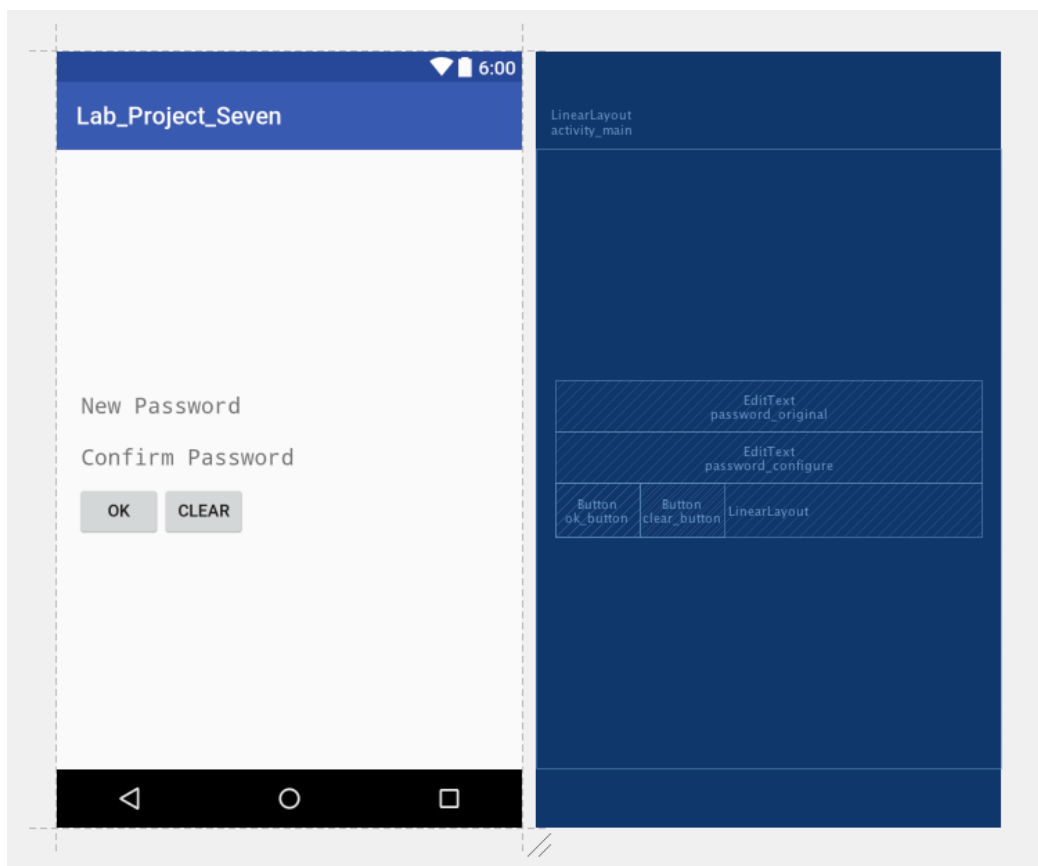
本次试验较上一次实验难度有所提升，主要需要熟悉 `SharedPreferences` 的使用及和Android 中的文件操作，活动生命周期控制等
图为项目结构



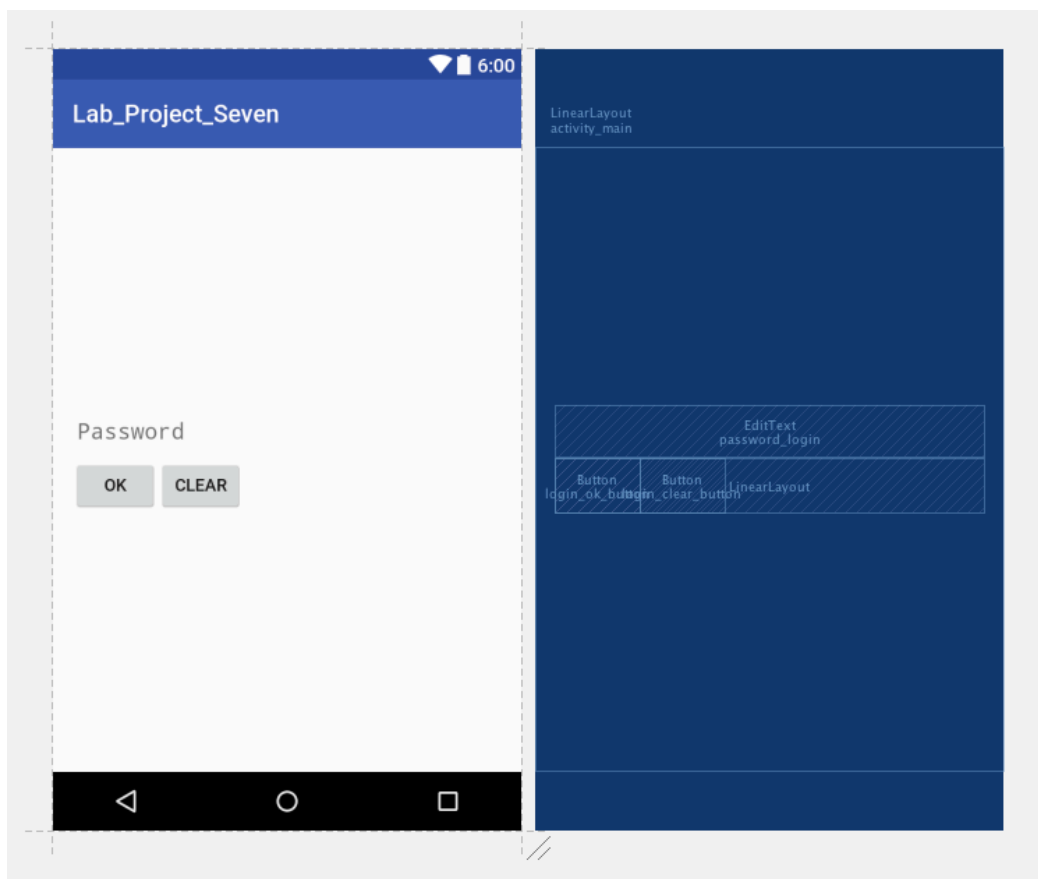
从实验文档看，需要实现的目标主要有

- 1、使用 `SharedPreferences` 保存密码
- 2、使用有关文件操作方法保存数据到文件

图为 `MainActivity` 初次登陆布局

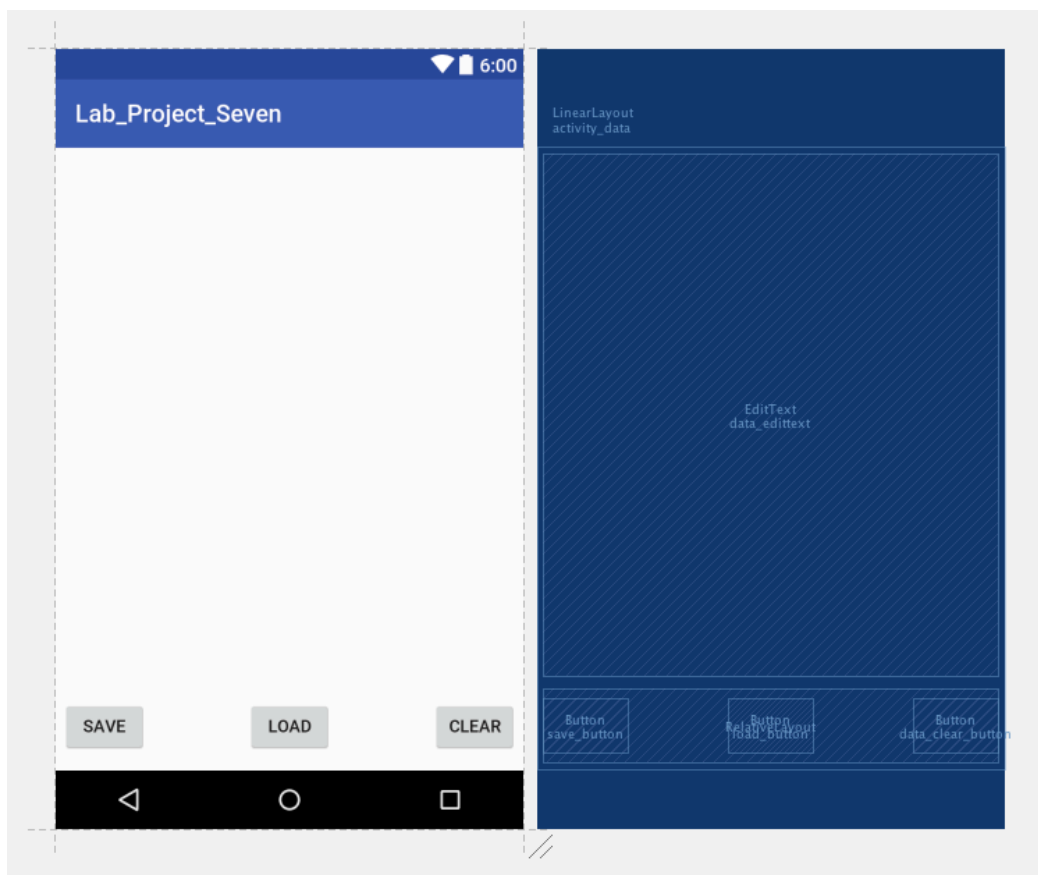


图为 MainActivity 保存密码后布局



可以看到，这里没有初次登录时用于确认密码的输入框

图为 `dataActivity` 布局



文本输入框有一定的外边距

接下来考虑实现

本次开发主要涉及两个方面：

1.通过 `SharedPreferences` 暂存密码，并实现身份确认功能

首先需要在重写的 `onCreate()` 中获取一个 `SharedPreferences` 对象

```
//获取SharedPreferences对象,从中可获得密码有关信息
final SharedPreferences password_preferences=getSharedPrefere
nces("password",Context.MODE_PRIVATE);
final SharedPreferences.Editor editor=password_preferences.
edit();
```

通过这个对象，可执行存入键值对及检验键是否存在

在载入布局前，首先通过观察是否有存储密码的键，并以此为依据判断密码是否存在。如果密码存在，则载入输入密码即登陆的布局，否则载入需要校验并记录密码的布局

```
if(password_preferences.contains("pas"))
```

其中 `pas` 为密码的键

判断输入框中的输入是否合法并弹出提示等部分之前的实验已经有所介绍，这里着重关注将密码存入及取出的部分

存入部分，需要用到之前获取的 `editor` 对象

```
//保存密码
editor.putString("pas", configureInput);
editor.commit();
```

这里以 `String` 格式存储密码

```
if(password_login_edittext.getText().toString().equals(password_preferences.getString("pas", ""))
{
    //打开新页面
    Intent intent = new Intent(MainActivity.this, dataActivity.class);
    startActivity(intent);
}
```

取出部分包含在一个判断中，使用 `getString()` 函数取出密码对应的 `String` 对象

2.通过 `Android` 提供的文件操作实现文本的存取

这一部分主要实现于 `dataActivity` 中，先为文件取一个好听的名字比如 `my_content`，然后通过 JAVA 的 `FileOutputStream` 及 `FileInputStream` 实现文件的读写工作，读写文件均需要使用异常捕获

读文件部分

```

//从文件中读取内容
try (FileInputStream fileInputStream = openFileInput(FILE_NAME))
{
    //从文件中读取出一个bytes数组
    byte[] contents = new byte[fileInputStream.available()]
;
    fileInputStream.read(contents);

    //将数组转化为字符串
    String result = new String(contents);
    data_edittext.setText(result);

    //这里可以说一说
    //设置光标位置为文本结尾处
    Editable text = data_edittext.getText();
    data_edittext.setSelection(text.length());

    //提示载入成功
    Toast.makeText(dataActivity.this, "Load successfully",
Toast.LENGTH_SHORT).show();
}
catch (IOException ex)
{
    //如果失败则提示
    Toast.makeText(dataActivity.this, "Fail to load file",
Toast.LENGTH_SHORT).show();
    Log.e("TAG", "Fail to read file.");
}
}

```

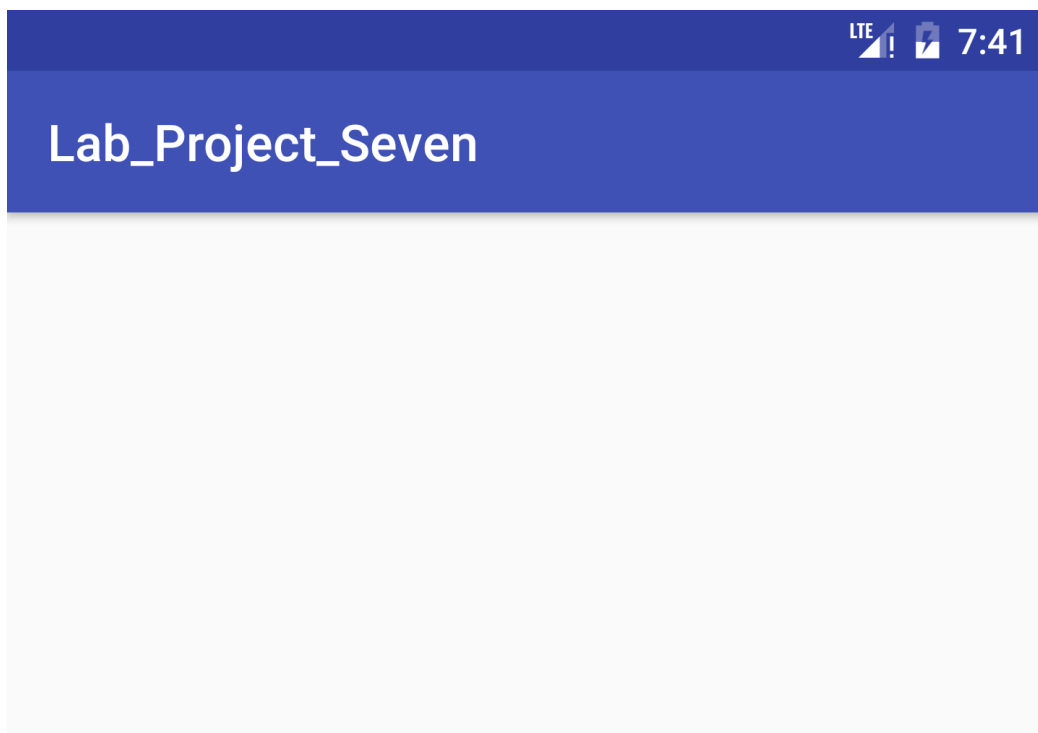
写文件部分

```
//向文件写入内容
try (FileOutputStream fileOutputStream = openFileOutput(FILE_NAME, MODE_PRIVATE))
{
    //获取文本输入框中的内容
    String str = data_edittext.getText().toString();
    //获取Bytes数组并写入文件
    fileOutputStream.write(str.getBytes());

    //提示写入成功
    Toast.makeText(dataActivity.this, "save successfully",
        Toast.LENGTH_SHORT).show();
}
catch (IOException ex)
{
    Log.e("TAG", "Fail to save file.");
}
```

实验成果截图

需要提供密码的页面



New Password

Confirm Password

OK

CLEAR

提示输入不能为空

LTE 7:41

Lab_Project_Seven

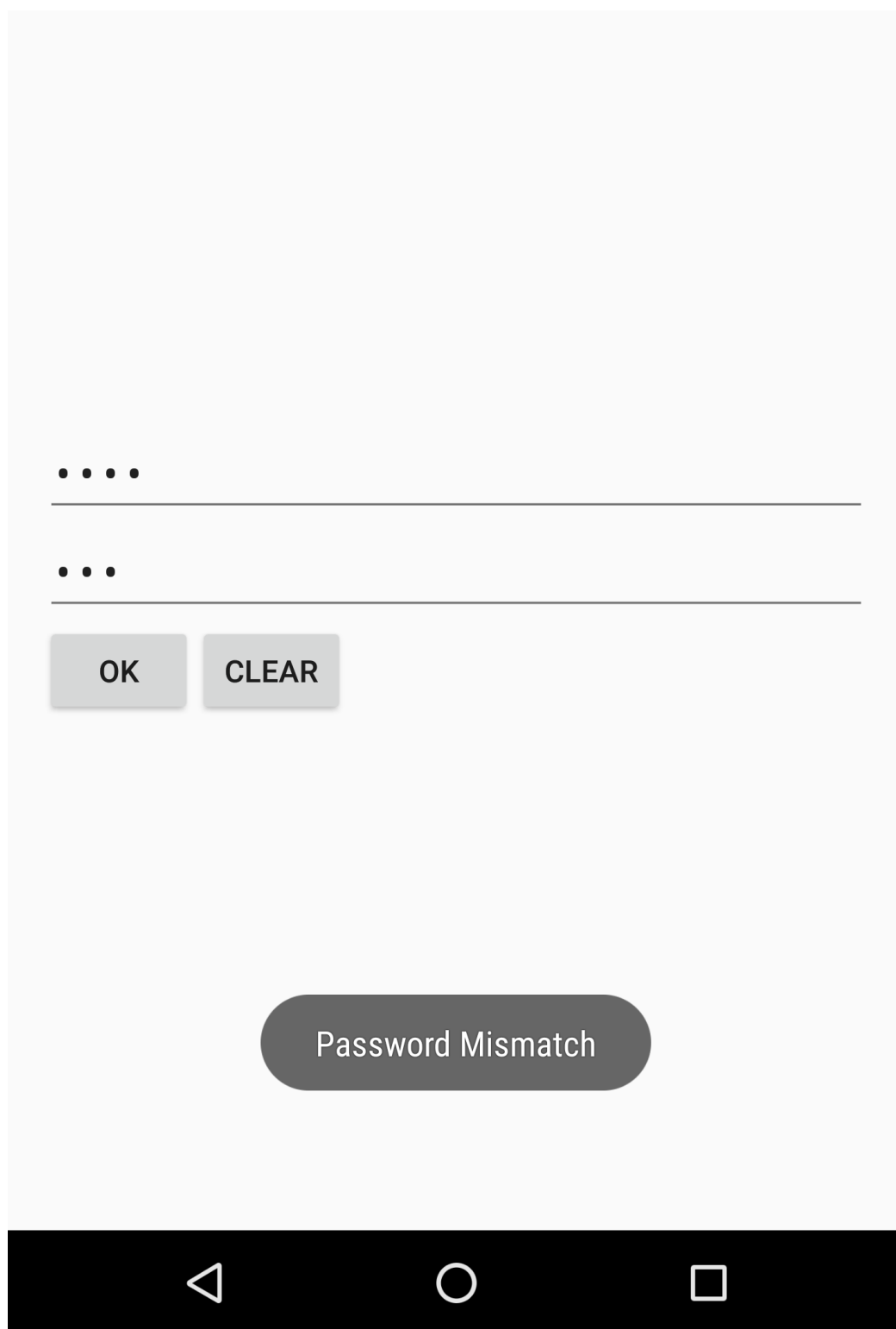
• • • •

Confirm Password

OK CLEAR

Password cannot be empty

如果输入内容不同则提示



点击 **CLEAR** 后页面被清空



Lab_Project_Seven

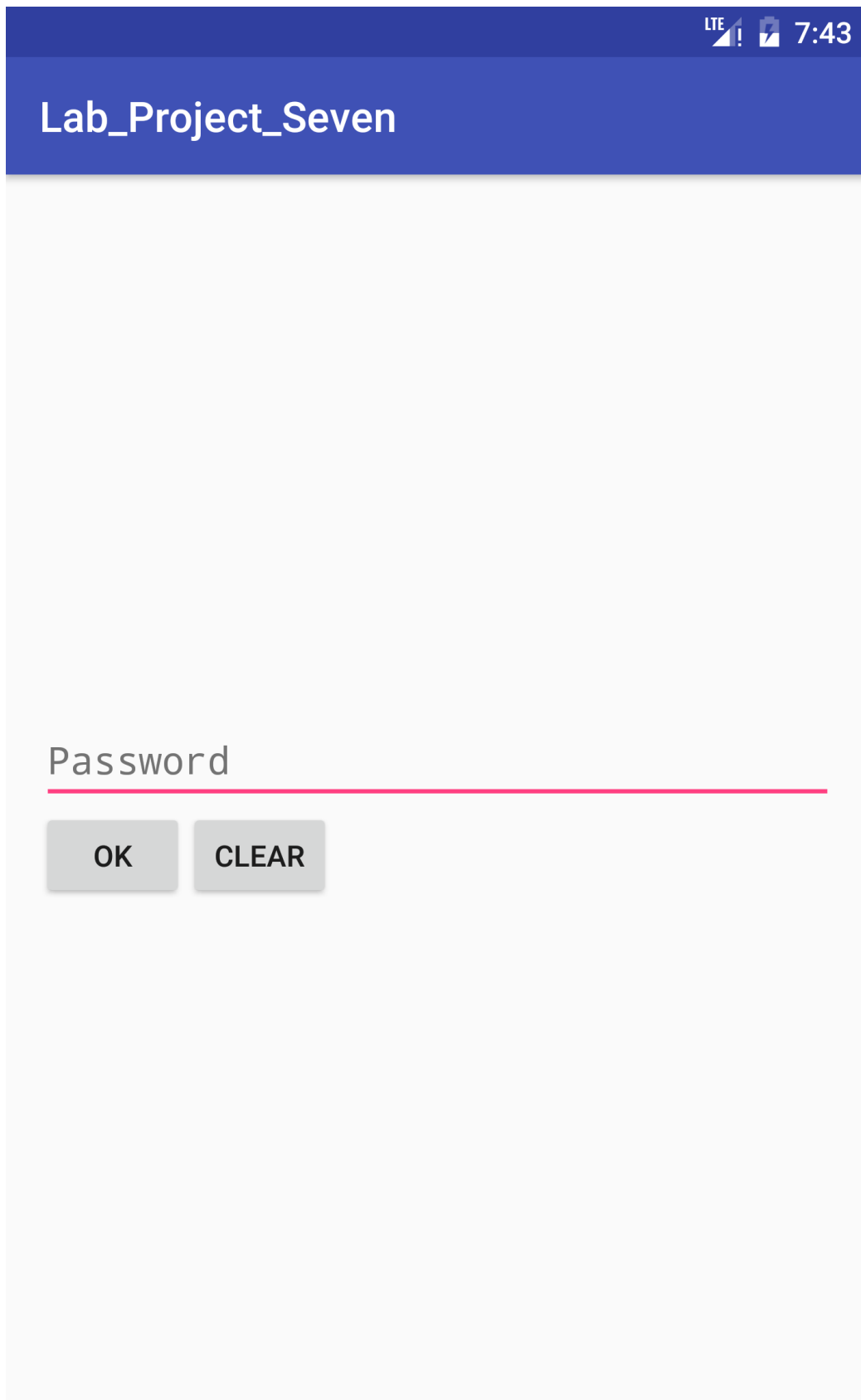
New Password

Confirm Password

OK

CLEAR

成功保存密码后再次打开应用，进入输入密码以登陆的页面

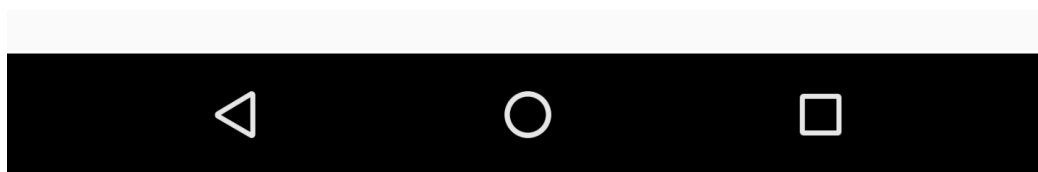


LTE 7:43

Lab_Project_Seven

Password

OK CLEAR



提示密码错误

LTE 7:43

Lab_Project_Seven

.....

OKCLEAR

Invalid Password

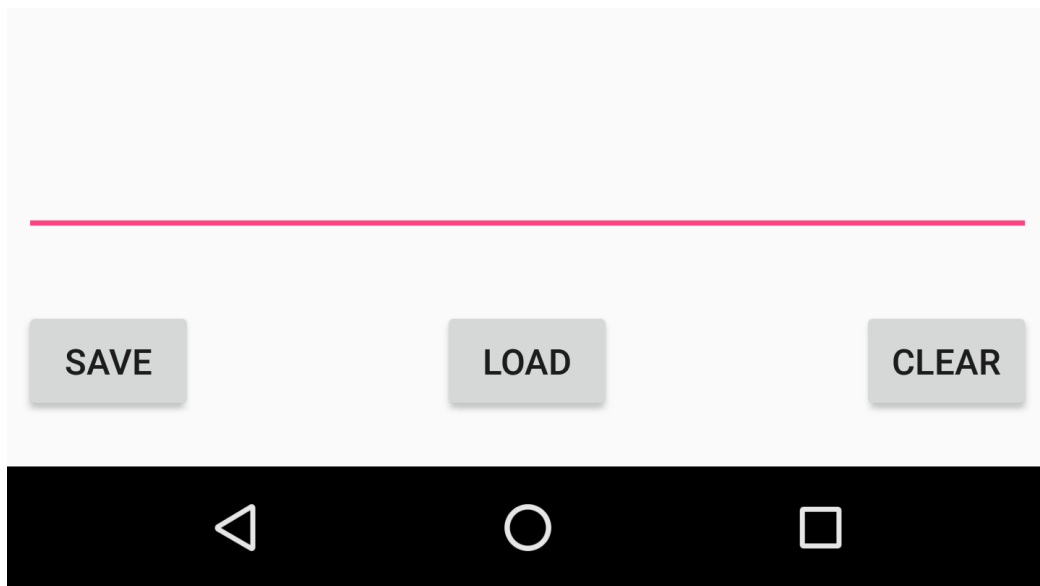


登陆成功后进入存取文本页面

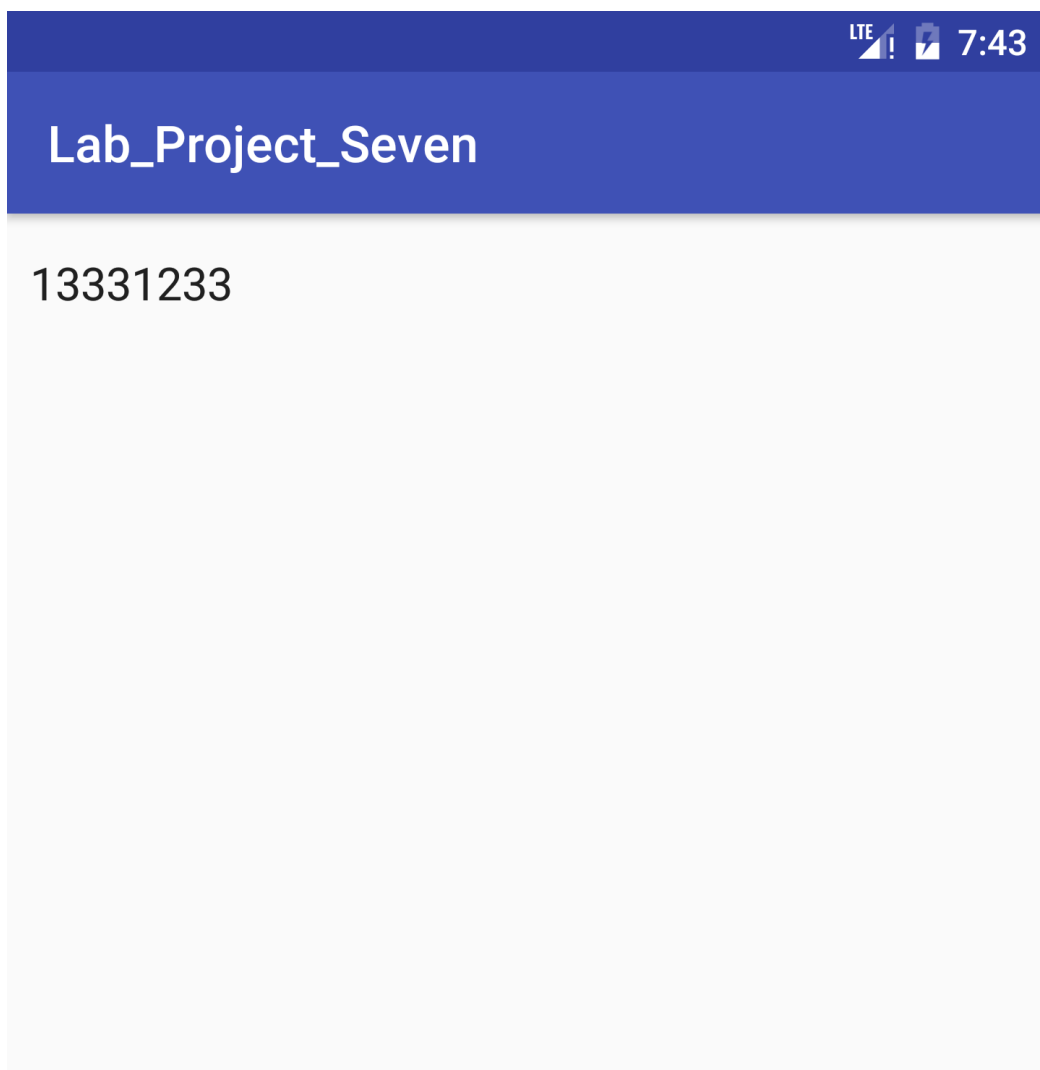
LTE 7:42

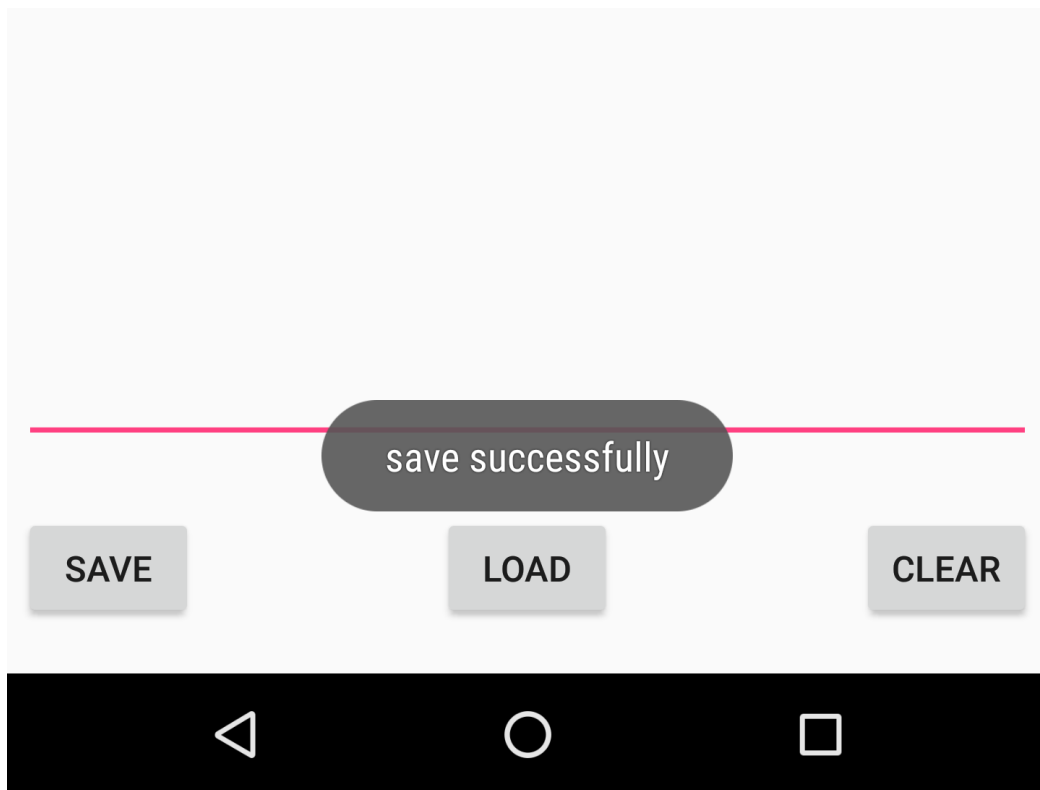
Lab_Project_Seven



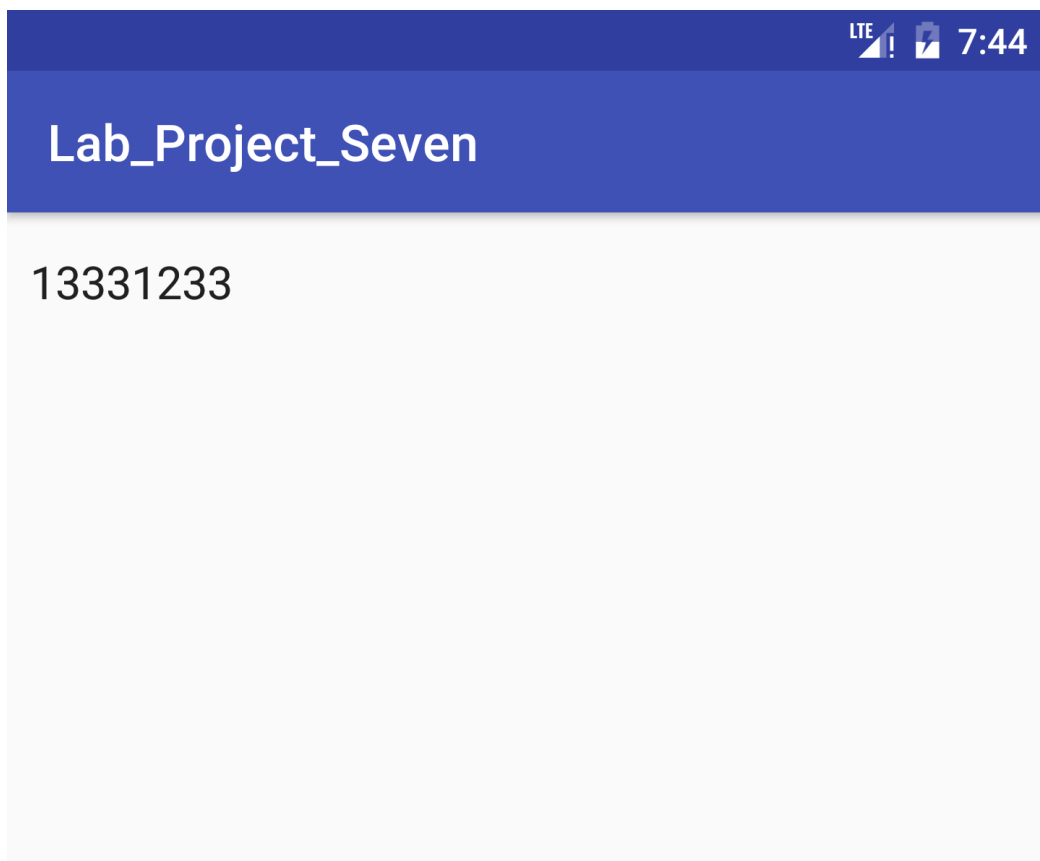


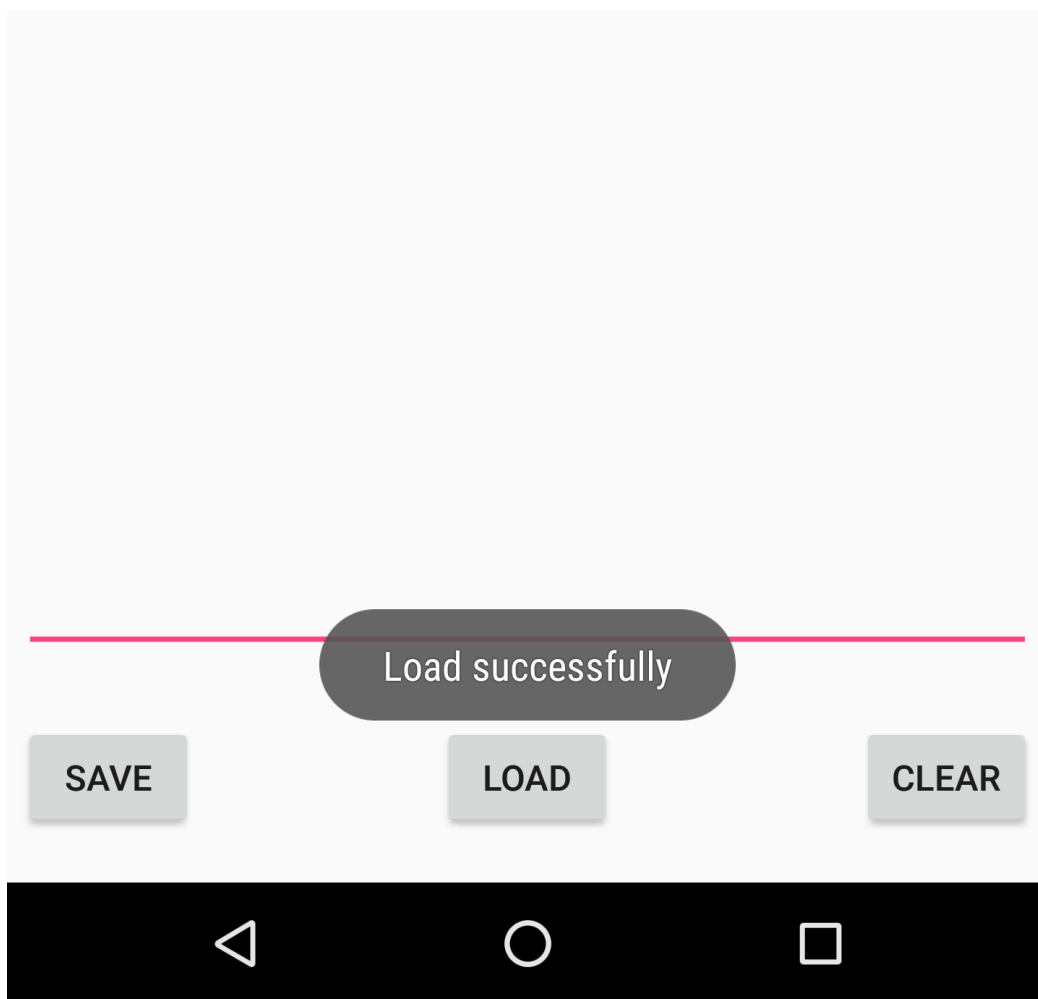
成功保存





成功载入





实验过程遇到的问题

遇到一个小问题，很快就解决了

在检测登陆状况时，我最初的想法是设置一个全局变量，全局变量保存是否已有密码的信息，这部分信息通过 `SharedPreferences` 的 `contains()` 函数检查是否存在有关的键来实现。这个全局变量在 `onCreate()` 函数外，而 `SharedPreferences` 的 `contains()`，`Editor` 的 `putString()`，`commit()` 函数只有在 `onCreate()` 函数内部才能够执行。

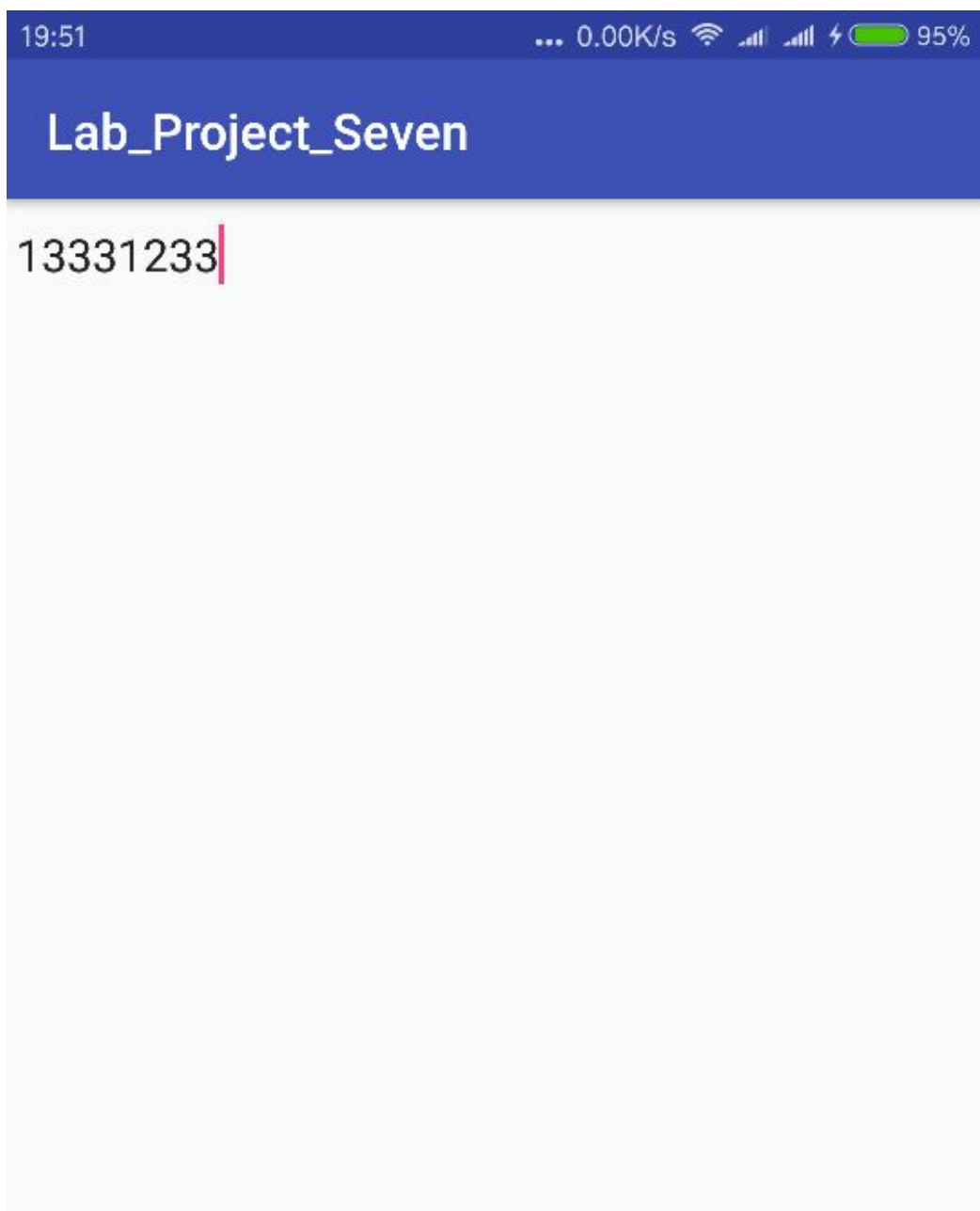
所以最终更改了方案，把上述部分放在 `onCreate()` 内部实现，问题解决

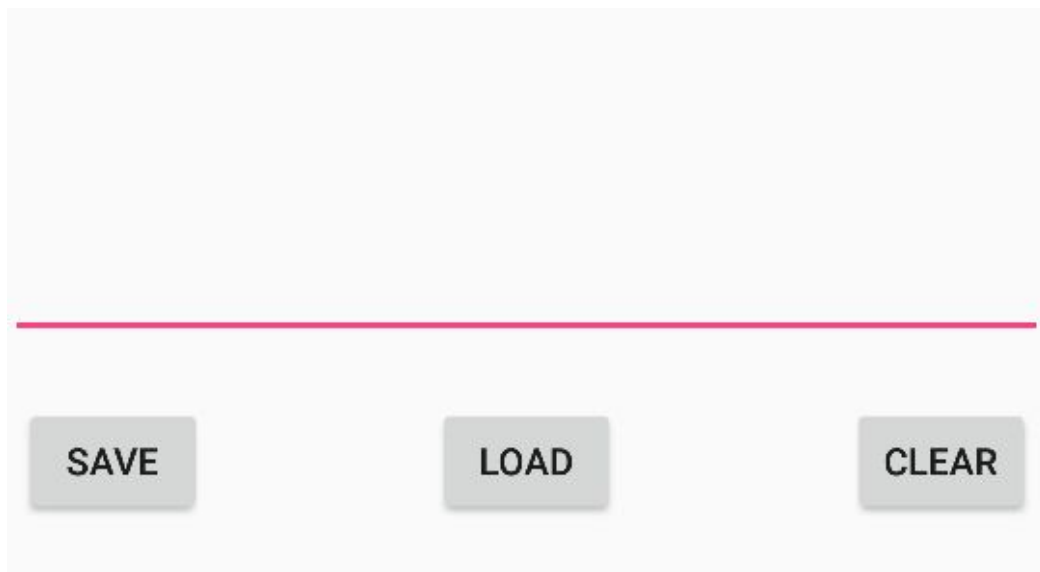
思考与总结

一般情况下程序将文本 `LOAD` 之后光标会停留在开始的位置不变，这之后的编辑造成了小小的困扰。于是我增加了对光标的设置，使得载入后光标自动定位在文本的末尾

```
//设置光标位置为文本结尾处  
Editable text = data_edittext.getText();  
data_edittext.setSelection(text.length());
```

如图所示





做了一点微小的工作，使得程序更自然。

另外，本次试验深入了解了Android开发的有关知识，尤其是和 `SharedPreferences` 以及文件存取有关的内容。这次实验的内容适中，大部分内容可以通过查阅PPT或作业说明得到解决，只有少部分需要百度。

原理部分更多的参考了上课的课件。Android较UWP应用范围更广，网上的资料也更多，问题也解决的比较顺利。总的来说比较有收获。