常州大学
CHANGZHOU UNIVERSITY

实 验 报 告

第 四 次实验

课程名称： 移动终端软件开发

实验名称： 数据存储与访问

学　　院： 信息数理学院　**班　　级：** 计算机 162

学　　号： 16430121　**姓　　名：** 林锦雄

完成日期： 2019.6.6

指导教师： 高晋树　**成　　绩：**

一、实验目的

1. 掌握 SharedPreferences 的使用方法；
2. 掌握各种文件存储的区别与适用情况；
3. 了解 SQLite 数据库的特点和体系结构；
4. 掌握 SQLite 数据库的建立和操作方法；
5. 理解 ContentProvider 的用途和原理；
6. 掌握 ContentProvider 的创建与使用方法。

二、实验过程

1、简单存储

简单存储指的是 Android 系统提供的 SharedPreferences 数据保存方式，将数据以最简单的方式进行永久性保存。

应用程序在使用过程中会被用户或系统关闭，如果能够在程序关闭前保存用户输入的数据，就可以在程序再次启动程序时恢复这些数据，进而提升用户体验。

使用 SharedPreferences 方式在程序关闭时保存用户在图 1.1 的界面上输入的数据，并在程序重新启动时自动恢复这些数据。

在 onStart()函数中调用 loadSharePreferences()函数，读取保存的信息，在 onStop()函数中调用 saveSharePreferences()函数，在关闭时保存界面上的信息。

代码如下：

```
1. public static final String PREFERENCE_NAME = "SaveSetting";
2. Public static int MODE=Context.MODE_WORLD_READABLE+Context.MODE_WORLD_WRITEABLE;
3. @Override
4. protected void onStart() {
5.     super.onStart();
6.     loadSharePreferences();
7. }
8. private void loadSharePreferences() {
9.     SharedPreferences share = getSharedPreferences(PREFERENCE_NAME, MODE);
10.      String name = share.getString("name", “韩松”");
11.      et.setText(name);
12.  }
13.  @Override
14.  protected void onStop() {
15.      super.onStop();
16.      saveSharePreferences();
17.  }
18.  private void saveSharePreferences() {
19.      SharedPreferences share = getSharedPreferences(PREFERENCE_NAME, MODE);
20.      SharedPreferences.Editor editor = share.edit();
21.      editor.putString("name", e1.getText().toString());
22.      editor.commit();
23.  }
```

2、建立存储设置界面

为图 1.2 的“设置”菜单项添加一个“存储设置”子菜单项，效果如图 4.1 所示。

图 4.1

创建新的 Blank Activity，Activity Name 和 Layout Name 修改为有实际含义的名称，界面如图 4.2 所示，其中“数据库存储”是默认选择项。

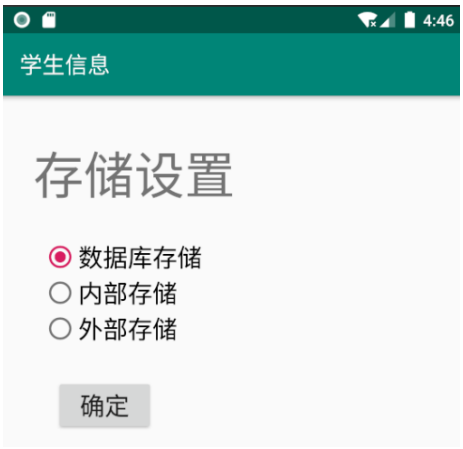

图 4.2

当点击“存储设置”子菜单项时，则启动存储设置界面。

代码如下：

```
int SUB1;
if(id == R.id.item2){
    Intent intent = new Intent(MainActivity.this,SaveActivity.class);
    startActivityForResult(intent,SUB1);
}
```

当点击 “确定”按钮则采用选择的存储方式保存数据，同时返回到图 1.1 的界面。

代码如下：

```
Button choose = (Button) findViewById(R.id.button1);
choose.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        if(radio0.isChecked()) Saveway = "sql";
        else if(radio1.isChecked()) Saveway = "inside";
```

```
7.          else if(radio2.isChecked()) Saveway = "outside";
8.          Uri data = Uri.parse(Saveway);
9.          Intent result = new Intent(null,data);
10.           setResult(RESULT_OK,result);
11.           finish();
12.      }
13.  });
```

3、文件存储

当在图 4.2 的存储设置界面选择“内部存储”时，点击图 1.1 界面上的“添加”按钮则将界面上输入的数据以文件形式保存在内部存储器上。

当在图 4.2 的存储设置界面选择“外部存储”时，点击图 1.1 界面上的“添加”按钮则将界面上输入的数据以文件形式保存在外部存储器上。

4、数据库存储

当在图 4.2 的存储设置界面选择“数据库存储”时，将图 1.1 和图 2.2 的界面上的相关数据保存在 SQLite 数据库中。

在 MainActivity 中的 onActivityResult()函数中获取从 SaveActivity 中获取的代表存储方式的变量 Saveway：

```
1. if(requestCode==SUB1){
2.     if(resultCode==Activity.RESULT_OK){
3.         Saveway = data.getData().toString();
4.     }
5. }
```

在添加按钮的点击事件上选择存储方式，其中 StudentsqlSave()，inSideSave()，outSideSave()分别为数据库存储，内部存储，外部存储的函数。

```
1. if(Saveway.equals("sql")){
2.     dbadapter.open();
3.     StudentsqlSave();
4. } else if(Saveway.equals("inside")){
5.     inSideSave();
6. } else if(Saveway.equals("outside")){
7.     outSideSave();//需要注册
8. } else{
9.     Toast.makeText(getApplicationContext(),"未设置存储方式
", Toast.LENGTH_LONG).show();
10. }
```

4.1 建立数据库

建立 SQLite 数据库，数据库名称自定义，并建立基本信息表（BaseInfo）和专业信息表（MajorInfo），在完成建立数据库的工作后，编程实现基本的数据库操作功能，包括数据的添加、删除、更新和查询。

4.2 基本信息表的操作

（1）添加按钮

在图 1.1 的界面上添加“删除”按钮

点击“添加”按钮，则把相关数据加入基本信息表中，添加成功应当给出相

应提示信息，添加不成功要给出不成功原因的提示信息。添加数据的同时要调用实验三中的服务计算 BMI 指数。

```
1. //在存储时为 People 的体质项获取服务中计算 BMI 的函数：
2. p.health=myservice.gettizhi(Double.valueOf(et3.getText().toString()),Double.value
Of(et2.getText().toString()));;
3. //在打开基本信息表的快捷菜单查看按钮的点击事件中增加：
4. data.putString("name",p.toString());
5. //在子 Activity 中接受该信息：
6. String msg = intent.getStringExtra("name");
```

点击“删除”按钮，则把基本信息表中全部数据删除，删除操作前要给出相应提示信息，确定是否删除。

```
1. delete.setOnClickListener(new OnClickListener() {
2.     @Override
3.     public void onClick(View v) {
4.         new AlertDialog.Builder(MainActivity.this).setMessage("是否确定删除？
").setPositiveButton("确定",new  DialogInterface.OnClickListener()  {
5.             @Override
6.             public void onClick(DialogInterface dialog, int which) {
7.             dbadapter.deleteALLDataStudent();
8.             }
9.         }).setNegativeButton("取消",  new DialogInterface.OnClickListener()  {
10.             @Override
11.             public void onClick(DialogInterface dialog, int which) {
12.                 dialog.dismiss();
13.             }
14.         }).show();
15.     }
16. }
```

（2）添加快捷菜单项

在图 1.4 已有的快捷菜单基础上为界面上的 ListView 控件对应的快捷菜单添加“修改”和“删除”两个菜单项。效果如图 4.3 所示。

图 4.3

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v, ContextMenuInfo menuInf
o) {
    menu.add(0, CONTEXT_MENU_1, 0, "查看");
    menu.add(0, CONTEXT_MENU_2, 1, "删除");
    menu.add(0, CONTEXT_MENU_3, 2, "修改");
}
```

点击“查看”菜单项则会启动信息显示界面，此时该界面上应该显示出所选择查看项的所有字段信息。

点击“修改”菜单项则会根据界面上中输入的内容修改所选择项的相关数据库记录，同时更新下面的 ListView 控件列表，修改成功，则要给出相应提示，如果修改不成功，也要给出不成功原因的提示信息。

```
if(item.getItemId()==CONTEXT_MENU_2){
    People people = new People();
    people.name = e1.getText().toString();
    long count = dbadapter.updataOneDataStudent(id, people);
    if (count == -1 ){
        Toast.makeText(getApplicationContext(), "修改失败
", Toast.LENGTH_LONG).show();
    } else {
        Toast.makeText(getApplicationContext(), "修改成功
", Toast.LENGTH_LONG).show();
    }
     list1.set(am.position, e1.getText().toString());
     adapter.notifyDataSetChanged();
}
```

点击“删除”菜单项则会删除所选择项的数据库记录，同时更新下面的 ListView 控件列表。

```
if(item.getItemId()==CONTEXT_MENU_3){
    dbadapter.deleteOneDataStduent(am.position);
    Toast.makeText(getApplicationContext(), "删除成功
", Toast.LENGTH_LONG).show();
    list1.remove(am.position);
    adapter.notifyDataSetChanged();
}
```

4.3 专业信息表的操作

当点击 “专业设置”菜单项时，则会启动专业设置界面，此时专业设置界面上应当将专业信息表中的所有专业列出来。效果如图 4.3 所示。

图 4.4

点击“添加”按钮则把 EditText 控件中输入的专业名称加入专业信息表，同时更新下面的 ListView 控件列表，如果 EditText 控件为空，则要给出相应提示。

```
b1.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
        if(equals(e1.getText().toString())==false){
            People people = new People();
            people.major = e1.getText().toString();
            dbadapter.insertMajor(people);
            list1.add(e1.getText().toString());
            adapter.notifyDataSetChanged();
         } else{
             Toast.makeText(getApplicationContext(),"输入为空
", Toast.LENGTH_LONG).show();
         }
     }
 });
```

长按专业设置界面上的 ListView 控件将启动对应的快捷菜单。

点击“修改”菜单项则会根据 EditText 控件中输入的内容修改所选择项的相关数据库记录，同时更新下面的 ListView 控件列表，如果 EditText 控件为空，则要给出相应提示。

在 onContextItemSelected()加入如下代码：

```
case CONTEXT_MENU_01:
    if(e1.getText().toString()!=null){
        People people = new People();
        people.major = e1.getText().toString();
        dbadapter.updataOneDataMajor(a.position, people);
        list1.set(a.position, e1.getText().toString());
        adapter.notifyDataSetChanged();
    } else {
        Toast.makeText(getApplicationContext(),"无替换内容
", Toast.LENGTH_LONG).show();
     }
```

点击“删除”菜单项则删除所选择项的数据库记录，同时更新下面的 ListView 控件列表。

```
case CONTEXT_MENU_02:
    dbadapter.deleteOneDataMajor(a.position);
    list1.remove(a.position);
    adapter.notifyDataSetChanged();
    break;
```

5、数据共享

在图 4.3 界面上添加一个“通讯录”选项菜单，效果如图 4.5 所示。

图 4.5

点击“通讯录”菜单项则利用 ContentProvider 组件获取 Android 系统内置通讯录里的一条联系人信息，将该条记录的联系人姓名填入图 4.3 的姓名输入框，然后将该联系人添加到基本信息表中，同时更新界面上的 ListView 控件列表。

三、实验总结

通过这次实验，初步掌握了 Android 应用开发中各种内部储存、外部储存、数据库储存方法，掌握 SharedPreferences 的使用方法；掌握各种文件存储的区别与适用情况；了解 SQLite 数据库的特点和体系结构；掌握 SQLite 数据库的建立和操作方法；理解 ContentProvider 的用途和原理；掌握 ContentProvider 的创建与使用方法。

数据存储与访问，无论是什么开发，只要设计到数据，都是十分重要的一项内容，熟练掌握各种数据的存储和访问方法，才能写出更好的程序，开发更好的软件。

四、思考练习

如何解决 SQLite 中文乱码的问题？

答：

```
SQLiteDatabase db = dbHelper.getWritableDatabase();
Cursor cursor = db.rawQuery("select * from info where id=?", new String[]{"8332"}
);
if (cursor != null && cursor.moveToFirst()) {
    info.setText(cursor.getString(1) + "\n" + new String(cursor.getBlob(2),"GBK")
 + "\n" + new String(cursor.getBlob(3),"GBK"));//转码处理
    cursor.close();
}
```

当然，这不能根治问题，每次查询时都要转换编码，很麻烦，可以直接把数据库转成 UTF-8 编码的，这样在 Android 系统下跑起来畅通无阻，可以使用 sqlite developer 工具来替代 SQLite Administrator。