**基于Android的读书笔记软件**

1. **系统的需求分析**

**1.1系统目标**

目本程序旨在开发一款简单实用、易操作的一款读书笔记软件。

**1.2功能需求**

该记事本软件应该具有以下功能：

（1）登录

（2）书籍分类管理

（3）书籍管理

（4）我的读书笔记

（5）汇总每个用户的笔记数量

（6）统计某一个用户的每一本书的笔记数量

1. **系统的概要设计**

该软件是一款是基于Android开发的手机应用，简单实用，易于上手。

* 1. 运行环境

1.Android手机或平板电脑：主频在1Ghz及以上，内存为1GB以上，系统版本为Android4.4.2及以上。

2.在android studio平台下安装了安卓android SDK 插件的电脑：Window All

* 1. 数据库表设计

用户表

|  |  |  |
| --- | --- | --- |
| 列名 | 数据类型 | 是否允许空 |
| Code | text | 不允许 |
| username | text | 不允许 |
| userpass | text | 不允许 |

分类表

|  |  |  |
| --- | --- | --- |
| 列名 | 数据类型 | 是否允许空 |
| bookid | text | 不允许 |
| sortname | text | 不允许 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

书籍表

|  |  |  |
| --- | --- | --- |
| 列名 | 数据类型 | 是否允许空 |
| bookcode | text | 不允许 |
| bookname | text | 不允许 |
| bookuser | text | 不允许 |
| bookchuban | text | 不允许 |
| Isbn | text | 不允许 |
| sort | text | 不允许 |

笔记表

|  |  |  |
| --- | --- | --- |
| 列名 | 数据类型 | 是否允许空 |
| code | text | 不允许 |
| bookcode | text | 不允许 |
| data | text | 不允许 |
| content |  |  |

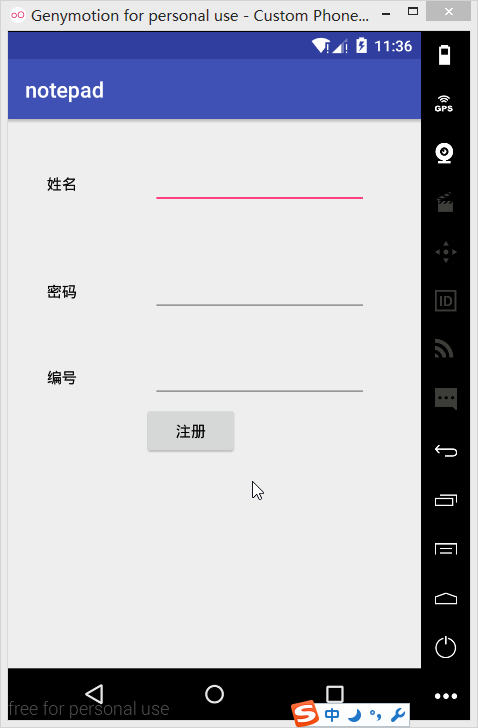
* 1. 数据库表语句

|  |
| --- |
| //用户表  "create table User ("  + "code text primary key, "  + "username text, "  + "userpass text)";  //分类表  "create table Booksort("  +"bookid text primary key,"  +"sortname text)";  //书籍表  "create table Book("  +"bookcode text primary key,"  +"bookname text,"  +"bookuser text,"  +"bookchuban text,"  +"ISBN text,"  +"sort text)";  //笔记表  "create table Note("  +"code text ,"  +"bookcode text,"  +"data text,"  +"content text,"  +"primary key(code,bookcode,data),"  +" FOREIGN KEY(code) REFERENCES User(code),"  + "FOREIGN KEY(bookcode) REFERENCES Book(bookcode))"; |

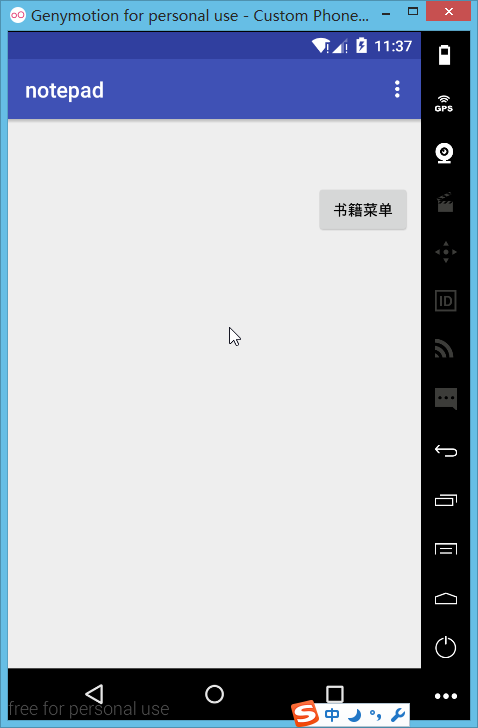
1. **设计与实现部分**

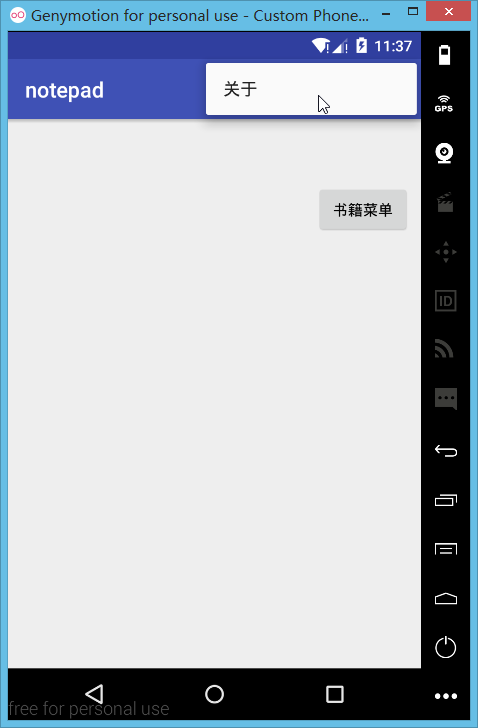
运行画面截图











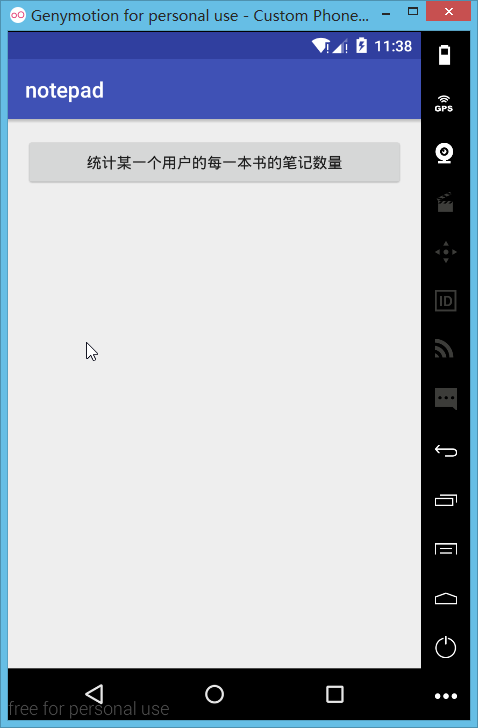














关键代码：

package com.example.diy.myapplication1;

/\*\*

\* Created by diy on 2017/4/14.

\*/

public class BookofManage {

private String bookcode;

private String name;

private String Zname;

private String publish;

private String ISBN;

private String sort;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getBookcode() {

return bookcode;

}

public void setBookcode(String bookcode) {

this.bookcode = bookcode;

}

public String getZname() {

return Zname;

}

public void setZname(String zname) {

Zname = zname;

}

public String getPublish() {

return publish;

}

public void setPublish(String publish) {

this.publish = publish;

}

public String getISBN() {

return ISBN;

}

public void setISBN(String ISBN) {

this.ISBN = ISBN;

}

public String getSort() {

return sort;

}

public void setSort(String sort) {

this.sort = sort;

}

}

package com.example.diy.myapplication1;

import android.content.DialogInterface;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.support.v7.app.AlertDialog;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ListView;

import android.widget.SimpleAdapter;

import android.widget.Toast;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class BookManage extends AppCompatActivity {

private Button add;

private Button delete;

private Button update;

private Button find;

private OpenHelper sqlHelper;

private ListView listView;

List<BookofManage> bookofManageList;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_book\_manage);

sqlHelper = new OpenHelper(this, "sqlite\_dbname", null, 2);

add=(Button)findViewById(R.id.btn\_insert);

add.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(BookManage.this);

builder.setTitle("录入");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.bookmanage,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText bookcode = (EditText) view.findViewById(R.id.et\_bookcode);

final EditText name = (EditText) view.findViewById(R.id.et\_name);

final EditText zname = (EditText) view.findViewById(R.id.et\_Zname);

final EditText chuban = (EditText) view.findViewById(R.id.et\_publish);

final EditText ISBN=(EditText) view.findViewById(R.id.et\_ISBN);

final EditText sort=(EditText) view.findViewById(R.id.et\_sort);

builder.setView(view);//设置login\_layout为对话提示框

builder.setCancelable(false);//设置为不可取消

//设置正面按钮，并做事件处理

builder.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

BookofManage b=new BookofManage();

String boocode = bookcode.getText().toString().trim();

String name1 = name.getText().toString().trim();

String zname1=zname.getText().toString().trim();

String chuban1=chuban.getText().toString().trim();

String is=ISBN.getText().toString().trim();

String sort1=sort.getText().toString().trim();

b.setBookcode(boocode);

b.setName(name1);

b.setZname(zname1);

b.setPublish(chuban1);

b.setISBN(is);

b.setSort(sort1);

int result=SqliteDB.getInstance(getApplicationContext()).saveBoomanage(b);

if (result==1){

Toast.makeText(BookManage.this, "录入成功", Toast.LENGTH\_SHORT).show();

}else if (result==-1)

{

Toast.makeText(BookManage.this, "编号已存在", Toast.LENGTH\_SHORT).show();

}

else

{

Toast.makeText(BookManage.this, "！", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder.show();//显示Dialog对话框

}

});

delete=(Button)findViewById(R.id.btn\_delete);

delete.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder1 = new AlertDialog.Builder(BookManage.this);

builder1.setTitle("删除");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.manage1,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText et\_username = (EditText) view.findViewById(R.id.et\_username);

builder1.setView(view);//设置login\_layout为对话提示框

builder1.setCancelable(false);//设置为不可取消

builder1.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

BookofManage bm=new BookofManage();

String name = et\_username.getText().toString().trim();

bm.setBookcode(name);

int result=SqliteDB.getInstance(getApplicationContext()).deleteBook(bm);

if (result==1){

Toast.makeText(BookManage.this, "删除成功", Toast.LENGTH\_SHORT).show();

}else if(result==-1)

{

Toast.makeText(BookManage.this, "编号不存在", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder1.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder1.show();//显示Dialog对话框

}

});

update=(Button)findViewById(R.id.btn\_change);

update.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(BookManage.this);

builder.setTitle("录入");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.bookmanage,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText bookcode = (EditText) view.findViewById(R.id.et\_bookcode);

final EditText name = (EditText) view.findViewById(R.id.et\_name);

final EditText zname = (EditText) view.findViewById(R.id.et\_Zname);

final EditText chuban = (EditText) view.findViewById(R.id.et\_publish);

final EditText ISBN=(EditText) view.findViewById(R.id.et\_ISBN);

final EditText sort=(EditText) view.findViewById(R.id.et\_sort);

builder.setView(view);//设置login\_layout为对话提示框

builder.setCancelable(false);//设置为不可取消

//设置正面按钮，并做事件处理

builder.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

BookofManage b=new BookofManage();

String boocode = bookcode.getText().toString().trim();

String name1 = name.getText().toString().trim();

String zname1=zname.getText().toString().trim();

String chuban1=chuban.getText().toString().trim();

String is=ISBN.getText().toString().trim();

String sort1=sort.getText().toString().trim();

b.setBookcode(boocode);

b.setName(name1);

b.setZname(zname1);

b.setPublish(chuban1);

b.setISBN(is);

b.setSort(sort1);

int result=SqliteDB.getInstance(getApplicationContext()).updateBook(b);

if (result==1){

Toast.makeText(BookManage.this, "修改成功", Toast.LENGTH\_SHORT).show();

}else if (result==-1)

{

Toast.makeText(BookManage.this, "编号不存在", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder.show();//显示Dialog对话框

}

});

find=(Button)findViewById(R.id.btn\_find);

listView=(ListView)findViewById(R.id.listview2) ;

find.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

try{

SQLiteDatabase db = sqlHelper.getWritableDatabase();

//游标查询每条数据

Cursor cursor = db.query("Book", null, null, null, null, null, null);

//定义list存储数据

List<Map<String, Object>> list = new ArrayList<Map<String, Object>>();

//适配器SimpleAdapter数据绑定

//错误:构造函数SimpleAdapter未定义 需把this修改为MainActivity.this

SimpleAdapter adapter = new SimpleAdapter(BookManage.this, list, R.layout.stu\_item1,

new String[]{"bookcode", "bookname","bookuser","bookchuban","ISBN","sort"},

new int[]{R.id.boo, R.id.boo1,R.id.boo2,R.id.boo3,

R.id.boo4,R.id.boo5});

//读取数据 游标移动到下一行

while(cursor.moveToNext()) {

Map<String, Object> map = new HashMap<String, Object>();

map.put( "bookcode", cursor.getString(cursor.getColumnIndex("bookcode")) );

map.put( "bookname", cursor.getString(cursor.getColumnIndex("bookname")) );

map.put( "bookuser", cursor.getString(cursor.getColumnIndex("bookuser")) );

map.put( "bookchuban", cursor.getString(cursor.getColumnIndex("bookchuban")) );

map.put( "ISBN", cursor.getString(cursor.getColumnIndex("ISBN")) );

map.put( "sort", cursor.getString(cursor.getColumnIndex("sort")) );

list.add(map);

}

listView.setAdapter(adapter);

}

catch (Exception e){

Log.i("exception", e.toString());

}

}

});

}

}

package com.example.diy.myapplication1;

/\*\*

\* Created by Administrator on 2015/11/10.

\*/

public class User {

private String id;

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getUserpass() {

return userpwd;

}

public void setUserpass(String userpwd) {

this.userpwd = userpwd;

}

private String username;

private String userpwd;

}

package com.example.diy.myapplication1;

import android.content.ContentValues;

import android.content.Context;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.util.Log;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

/\*\*

\* Created by Administrator on 2015/11/10.

\*/

public class SqliteDB {

/\*\*

\* 数据库名

\*/

public static final String DB\_NAME = "sqlite\_dbname";

/\*\*

\* 数据库版本

\*/

public static final int VERSION = 1;

private static SqliteDB sqliteDB;

private SQLiteDatabase db;

private SqliteDB(Context context) {

OpenHelper dbHelper = new OpenHelper(context, DB\_NAME, null, VERSION);

db = dbHelper.getWritableDatabase();

}

/\*\*

\* 获取SqliteDB实例

\*

\* @param context

\*/

public synchronized static SqliteDB getInstance(Context context) {

if (sqliteDB == null) {

sqliteDB = new SqliteDB(context);

}

return sqliteDB;

}

/\*\*

\* 将User实例存储到数据库。

\*/

public int saveUser(User user) {

if (user != null) {

/\* ContentValues values = new ContentValues();

values.put("username", user.getUsername());

values.put("userpwd", user.getUserpwd());

db.insert("User", null, values);\*/

Cursor cursor = db.rawQuery("select \* from User where username=?", new String[]{user.getUsername().toString()});

if (cursor.getCount() > 0) {

return -1;

} else {

try {

db.execSQL("insert into User(code,username,userpass) values(?,?,?)", new String[]{user.getId().toString(),user.getUsername().toString(), user.getUserpass().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

} else {

return 0;

}

}

//储存

public int saveBoomanage(BookofManage bookofManage)

{

if(bookofManage!=null)

{

Cursor cursor=db.rawQuery("select \* from Book where bookcode=?",new String[]{bookofManage.getBookcode().toString()});

if(cursor.getCount()>0)

{

return -1;

}

else

{

try

{

db.execSQL("insert into Book(bookcode,bookname,bookuser,bookchuban,ISBN,sort)values(?,?,?,?,?,?)",new String[]{bookofManage.getBookcode().toString(),

bookofManage.getName().toString(),bookofManage.getZname().toString(),bookofManage.getPublish().toString(),

bookofManage.getISBN().toString(),bookofManage.getSort().toString()});

}catch(Exception e)

{

Log.d("错误", e.getMessage().toString());

}

return 1;

}

}

else

{

return 0;

}

}

//储存

public int saveBooksort(BookofSort booksort) {

if (booksort != null) {

Cursor cursor = db.rawQuery("select \* from Booksort where bookid=?", new String[]{booksort.getBookid().toString()});

if (cursor.getCount() > 0) {

return -1;

} else {

try {

db.execSQL("insert into Booksort(bookid,sortname) values(?,?) ", new String[]{booksort.getBookid().toString(), booksort.getSortname().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

} else {

return 0;

}

}

//储存

public int savenote(note n)

{

if(n!=null)

{

Cursor cursor = db.rawQuery("select \* from User where code=?", new String[]{n.getCode().toString()});

Cursor cursor1=db.rawQuery("select \* from Book where bookcode=?", new String[]{n.getBookcode()});

if (cursor.getCount()<1||(cursor1.getCount())<1) {

return -1;

} else {

try {

db.execSQL("insert into Note(code,bookcode,data,content) values(?,?,?,?) ", new String[]{n.getCode().toString(),

n.getBookcode().toString(),n.getData().toString(),n.getContent().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

} else {

return 0;

}

}

//删除

public int updateBooksort(BookofSort bookofSort) {

Cursor cursor = db.rawQuery("select \* from Booksort where bookid=?", new String[]{bookofSort.getBookid().toString()});

if (cursor.getCount() < 0 || cursor.getCount() == 0) {

return -1;

} else {

try {

db.delete("Booksort", "bookid=?", new String[]{bookofSort.getBookid().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

}

public int deleteBook(BookofManage bookofManage)

{

Cursor cursor = db.rawQuery("select \* from Book where bookcode=?", new String[]{bookofManage.getBookcode().toString()});

if (cursor.getCount() < 0 || cursor.getCount() == 0) {

return -1;

} else {

try {

db.delete("Book", "bookcode=?", new String[]{bookofManage.getBookcode().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

}

public int deletenote(note n)

{

Cursor cursor=db.rawQuery("select \* from Note where code=?",new String[]{n.getCode().toString()});

Cursor cursor1=db.rawQuery("select \* from Note where bookcode=?", new String[]{n.getBookcode().toString()});

Cursor cursor2=db.rawQuery("select \* from Note where data=?",new String[]{n.getData().toString()});

if (cursor.getCount() < 0 || cursor1.getCount()<1||cursor2.getCount()<1) {

return -1;

} else {

try {

db.delete("Note", "code=? and bookcode=? and data=?", new String[]{n.getCode().toString(),n.getBookcode().toString(),n.getData().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

}

//修改

public int changeBooksort(BookofSort bookofSort) {

ContentValues values = new ContentValues();

Cursor cursor1 = db.rawQuery("select \* from Booksort where bookid=?", new String[]{bookofSort.getBookid().toString()});

if (cursor1.getCount() < 0 || cursor1.getCount() == 0) {

return -1;

}

else{

try {

db.update("Booksort", values, "sortname= ?", new String[]{bookofSort.getSortname().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

}

public int updatenote(note n) {

ContentValues values = new ContentValues();

Cursor cursor = db.rawQuery("select \* from Note where code=?", new String[]{n.getCode().toString()});

Cursor cursor1 = db.rawQuery("select \* from Note where bookcode=?", new String[]{n.getBookcode().toString()});

Cursor cursor2 = db.rawQuery("select \* from Note where data=?", new String[]{n.getData().toString()});

if (cursor1.getCount() < 1 || cursor.getCount() < 1 || cursor2.getCount() < 1) {

return -1;

} else {

try {

db.update("Note", values, "content= ?", new String[]{n.getContent().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

}

//更新

public int updateBook(BookofManage bookofManage)

{

ContentValues values = new ContentValues();

Cursor cursor1 = db.rawQuery("select \* from Book where bookcode=?", new String[]{bookofManage.getBookcode().toString()});

if (cursor1.getCount() < 0 || cursor1.getCount() == 0) {

return -1;

}

else{

try {

db.update("Book", values, "bookname= ?", new String[]{bookofManage.getName().toString()});

db.update("Book", values, "bookuser= ?", new String[]{bookofManage.getZname().toString()});

db.update("Book", values, "bookchuban= ?", new String[]{bookofManage.getPublish().toString()});

db.update("Book", values, "ISBN= ?", new String[]{bookofManage.getISBN().toString()});

db.update("Book", values, "sort= ?", new String[]{bookofManage.getSort().toString()});

} catch (Exception e) {

Log.d("错误", e.getMessage().toString());

}

return 1;

}

}

// public List<BookofSort>loadSort()

// {

// List<BookofSort> list = new ArrayList<BookofSort>();

// Cursor cursor = db

// .query("Booksort", null, null, null, null, null, null);

// if (cursor.moveToFirst()) {

// do {

// BookofSort sort=new BookofSort();

//

// sort.setBookid(cursor.getString(cursor

// .getColumnIndex("bookid")));

// sort.setSortname(cursor.getString(cursor.getColumnIndex("sortname")));

// list.add(sort);

// } while (cursor.moveToNext());

// }

// return list;

//

//

// }

/\*\*

\* 从数据库读取User信息。

\*/

public List<User> loadUser() {

List<User> list = new ArrayList<User>();

Cursor cursor = db

.query("User", null, null, null, null, null, null);

if (cursor.moveToFirst()) {

do {

User user = new User();

user.setId(cursor.getString(cursor.getColumnIndex("code")));

user.setUsername(cursor.getString(cursor

.getColumnIndex("username")));

user.setUserpass(cursor.getString(cursor

.getColumnIndex("userpass")));

list.add(user);

} while (cursor.moveToNext());

}

return list;

}

public int Quer(String pwd,String name)

{

HashMap<String,String> hashmap=new HashMap<String,String>();

Cursor cursor =db.rawQuery("select \* from User where username=?", new String[]{name});

// hashmap.put("name",db.rawQuery("select \* from User where name=?",new String[]{name}).toString());

if (cursor.getCount()>0)

{

Cursor pwdcursor =db.rawQuery("select \* from User where userpass=? and username=?",new String[]{pwd,name});

if (pwdcursor.getCount()>0)

{

return 1;

}

else {

return -1;

}

}

else {

return 0;

}

}

}

package com.example.diy.myapplication1;

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import java.util.ArrayList;

import java.util.List;

public class RegisterActivity extends AppCompatActivity {

private Button zc;

private EditText username;

private EditText userpass;

private EditText id;

private TextView state;

private List<User> userList;

private List<User> dataList = new ArrayList<>();

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_register);

zc=(Button)findViewById(R.id.tj);

username=(EditText)findViewById(R.id.Username1);

userpass=(EditText)findViewById(R.id.Userpass1);

id=(EditText)findViewById(R.id.password1);

state=(TextView)findViewById(R.id.state);

zc.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String name=username.getText().toString().trim();

String pass=userpass.getText().toString().trim();

String id1=id.getText().toString().trim();

User user=new User();

user.setUsername(name);

user.setUserpass(pass);

user.setId(id1);

int result=SqliteDB.getInstance(getApplicationContext()).saveUser(user);

if (result==1){

//state.setText("注册成功！");

startActivity(new Intent(RegisterActivity.this,LoginActivity.class));

}else if (result==-1)

{

state.setText("用户名已经存在！");

}

else

{

state.setText("！");

}

}

});

}

}

package com.example.diy.myapplication1;

import android.content.Context;

import android.database.sqlite.SQLiteDatabase;

import android.database.sqlite.SQLiteDatabase.CursorFactory;

import android.database.sqlite.SQLiteOpenHelper;

public class OpenHelper extends SQLiteOpenHelper {

//建表语句

public static final String CREATE\_USER = "create table User ("

+ "code text primary key, "

+ "username text, "

+ "userpass text)";

//分类表

public static final String CREATE\_BOOKSORT=

"create table Booksort("

+"bookid text primary key,"

+"sortname text)";

//书籍表

public static final String CREATE\_BOOK=

"create table Book("

+"bookcode text primary key,"

+"bookname text,"

+"bookuser text,"

+"bookchuban text,"

+"ISBN text,"

+"sort text)";

// //笔记表

public static final String CREATE\_NOTE=

"create table Note("

+"code text ,"

+"bookcode text,"

+"data text,"

+"content text,"

+"primary key(code,bookcode,data),"

+" FOREIGN KEY(code) REFERENCES User(code),"

+ "FOREIGN KEY(bookcode) REFERENCES Book(bookcode))";

public OpenHelper(Context context, String name, CursorFactory factory,

int version) {

super(context, name, factory, version);

// TODO Auto-generated constructor stub

}

@Override

public void onCreate(SQLiteDatabase db) {

// TODO Auto-generated method stub

db.execSQL(CREATE\_USER);//创建用户表

db.execSQL(CREATE\_BOOK);//创建书籍表

db.execSQL(CREATE\_BOOKSORT);//创建书籍分类表

db.execSQL(CREATE\_NOTE);//创建笔记表

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

// TODO Auto-generated method stub

}

}

package com.example.diy.myapplication1;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

public class Number extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_number);

}

}

package com.example.diy.myapplication1;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.ListView;

import android.widget.SimpleAdapter;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class Notenumer extends AppCompatActivity {

private Button find;

private OpenHelper sqlHelper;

private ListView listView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_notenumer);

sqlHelper = new OpenHelper(this, "sqlite\_dbname", null, 2);

find=(Button)findViewById(R.id.notenumber\_find);

listView=(ListView)findViewById(R.id.listview4);

find.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

try {

SQLiteDatabase db = sqlHelper.getWritableDatabase();

//游标查询每条数据

Cursor cursor=db.rawQuery("select code,count(\*) from Note ",null);

//定义list存储数据

List<Map<String, Object>> list = new ArrayList<Map<String, Object>>();

//适配器SimpleAdapter数据绑定

//错误:构造函数SimpleAdapter未定义 需把this修改为MainActivity.this

SimpleAdapter adapter = new SimpleAdapter(Notenumer.this, list, R.layout.stu\_item4,

new String[]{"code,count(\*)"},

new int[]{R.id.a1,R.id.b1});

//读取数据 游标移动到下一行

while(cursor.moveToNext()) {

Map<String, Object> map = new HashMap<String, Object>();

map.put( "code", cursor.getString(cursor.getColumnIndex("code")) );

// map.put( "username", cursor.getString(cursor.getColumnIndex("username")) );

map.put( "count(\*)", cursor.getString(cursor.getColumnIndex("count(\*)")) );

list.add(map);

}

listView.setAdapter(adapter);

}

catch (Exception e){

Log.i("exception", e.toString());

}

}

});

}

}

package com.example.diy.myapplication1;

import android.content.DialogInterface;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.support.v7.app.AlertDialog;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ListView;

import android.widget.SimpleAdapter;

import android.widget.Toast;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class Mynote extends AppCompatActivity {

private Button add;

private Button delete;

private Button update;

private Button find;

private OpenHelper sqlHelper;

private ListView listView;

List<note> noteList;

private EditText code,bookcode,data,content;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_mynote);

sqlHelper = new OpenHelper(this, "sqlite\_dbname", null, 2);

add=(Button)findViewById(R.id.note\_find);

add.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(Mynote.this);

builder.setTitle("录入");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.note,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText usercode = (EditText) view.findViewById(R.id.note\_code1);

final EditText bookcode= (EditText) view.findViewById(R.id.note\_bookcode1);

final EditText data = (EditText) view.findViewById(R.id.note\_data1);

final EditText content= (EditText) view.findViewById(R.id.note\_content1);

builder.setView(view);//设置login\_layout为对话提示框

builder.setCancelable(false);//设置为不可取消

//设置正面按钮，并做事件处理

builder.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

note n=new note();

String ucode = usercode.getText().toString().trim();

String bcode = bookcode.getText().toString().trim();

String da = data.getText().toString().trim();

String cont = content.getText().toString().trim();

n.setCode(ucode);

n.setBookcode(bcode);

n.setData(da);

n.setContent(cont);

int result=SqliteDB.getInstance(getApplicationContext()).savenote(n);

if (result==1){

Toast.makeText(Mynote.this, "录入成功", Toast.LENGTH\_SHORT).show();

}else if (result==-1)

{

Toast.makeText(Mynote.this, "您的编号没有完全写正确", Toast.LENGTH\_SHORT).show();

}

else

{

Toast.makeText(Mynote.this, "！", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder.show();//显示Dialog对话框

}

});

delete=(Button)findViewById(R.id.note\_delete);

delete.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder1 = new AlertDialog.Builder(Mynote.this);

builder1.setTitle("删除");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.note1,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText usercode = (EditText) view.findViewById(R.id.note\_code1);

final EditText bookcode= (EditText) view.findViewById(R.id.note\_bookcode1);

final EditText data = (EditText) view.findViewById(R.id.note\_data1);

builder1.setView(view);//设置login\_layout为对话提示框

builder1.setCancelable(false);//设置为不可取消

builder1.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

note n=new note();

String ucode = usercode.getText().toString().trim();

String bcode=bookcode.getText().toString().trim();

String da=data.getText().toString().trim();

n.setCode(ucode);

n.setBookcode(bcode);

n.setData(da);

int result=SqliteDB.getInstance(getApplicationContext()).deletenote(n);

if (result==1){

Toast.makeText(Mynote.this, "删除成功", Toast.LENGTH\_SHORT).show();

}else if(result==-1)

{

Toast.makeText(Mynote.this, "各种编号不存在", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder1.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder1.show();//显示Dialog对话框

}

});

update=(Button)findViewById(R.id.note\_update);

update.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(Mynote.this);

builder.setTitle("修改");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.note,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText usercode = (EditText) view.findViewById(R.id.note\_code1);

final EditText bookcode= (EditText) view.findViewById(R.id.note\_bookcode1);

final EditText data = (EditText) view.findViewById(R.id.note\_data1);

final EditText content= (EditText) view.findViewById(R.id.note\_content1);

builder.setView(view);//设置login\_layout为对话提示框

builder.setCancelable(false);//设置为不可取消

//设置正面按钮，并做事件处理

builder.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

note n=new note();

String ucode = usercode.getText().toString().trim();

String bcode = bookcode.getText().toString().trim();

String da = data.getText().toString().trim();

String cont = content.getText().toString().trim();

n.setCode(ucode);

n.setBookcode(bcode);

n.setData(da);

n.setContent(cont);

int result=SqliteDB.getInstance(getApplicationContext()).updatenote(n);

if (result==1){

Toast.makeText(Mynote.this, "修改成功", Toast.LENGTH\_SHORT).show();

}else if (result==-1)

{

Toast.makeText(Mynote.this, "编号不存在", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder.show();//显示Dialog对话框

}

});

find=(Button)findViewById(R.id.note\_insert);

listView=(ListView)findViewById(R.id.listview3) ;

find.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

try{

SQLiteDatabase db = sqlHelper.getWritableDatabase();

//游标查询每条数据

Cursor cursor = db.query("Note", null, null, null, null, null, null);

//数据库标的名称

//selection 查询条件

//selectionArgs:查询结果

//groupBy : 分组列

//having : 分组条件

//orderBy :拍序列

//limit: 分页查询限制

//定义list存储数据

List<Map<String, Object>> list = new ArrayList<Map<String, Object>>();

//适配器SimpleAdapter数据绑定

//错误:构造函数SimpleAdapter未定义 需把this修改为MainActivity.this

SimpleAdapter adapter = new SimpleAdapter(Mynote.this, list, R.layout.stu\_item3,

new String[]{"code", "bookcode","data","content"},

new int[]{R.id.a, R.id.b,R.id.c,R.id.d

});

//读取数据 游标移动到下一行

while(cursor.moveToNext()) {

Map<String, Object> map = new HashMap<String, Object>();

map.put( "code", cursor.getString(cursor.getColumnIndex("code")) );

map.put( "bookcode", cursor.getString(cursor.getColumnIndex("bookcode")) );

map.put( "data", cursor.getString(cursor.getColumnIndex("data")) );

map.put( "content", cursor.getString(cursor.getColumnIndex("content")) );

list.add(map);

}

listView.setAdapter(adapter);

}

catch (Exception e){

Log.i("exception", e.toString());

}

}

});

}

}

package com.example.diy.myapplication1;

import android.app.AlertDialog;

import android.content.Intent;

import android.os.Bundle;

import android.support.v7.app.AppCompatActivity;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.LinearLayout;

import android.widget.RelativeLayout;

import com.jeremyfeinstein.slidingmenu.lib.SlidingMenu;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {

//声明控件（主页面）

private Button left;

//声明侧滑兰

private SlidingMenu menu;

//声明子布局控件

private RelativeLayout left\_update;

private RelativeLayout left\_clear;

private RelativeLayout left\_lixian;

private RelativeLayout left\_about;

private RelativeLayout left\_day;

private RelativeLayout GY;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

//绑定id

left = (Button) findViewById(R.id.btn);

left.setOnClickListener(this);

slidingMenu();

}

private void slidingMenu() {

//初始化侧滑栏

menu=new SlidingMenu(this);

//设置类型

menu.setMode(SlidingMenu.LEFT);

//设置触屏模式

menu.setTouchModeAbove(SlidingMenu.TOUCHMODE\_FULLSCREEN);

//设置阴影宽度

menu.setShadowWidthRes(R.dimen.shadow\_width);

//滑动主页面剩余宽度

menu.setBehindOffsetRes(R.dimen.slidingmenu\_offset);

//设置渐变效果

menu.setFadeDegree(0.4f);

//加载到当前activity

menu.attachToActivity(this,SlidingMenu.SLIDING\_CONTENT);

//加载左侧布局

View view= LinearLayout.inflate(this,R.layout.activity\_leftment\_activity,null);

menu.setMenu(view);

//绑定id

left\_update = (RelativeLayout) findViewById(R.id.manage);

left\_clear = (RelativeLayout) findViewById(R.id.bookmanage);

left\_lixian = (RelativeLayout) findViewById(R.id.readnote);

left\_about = (RelativeLayout) findViewById(R.id.number);

left\_day = (RelativeLayout) findViewById(R.id.TJ);

GY=(RelativeLayout)findViewById(R.id.GY);

//设置监听

left\_update.setOnClickListener(this);

left\_clear.setOnClickListener(this);

left\_lixian.setOnClickListener(this);

left\_about.setOnClickListener(this);

left\_day.setOnClickListener(this);

GY.setOnClickListener(this);

}

@Override

public void onClick(View v) {

switch (v.getId()){

case R.id.btn:

menu.toggle();//点击按钮切换到侧滑页

break;

case R.id.manage:

startActivity(new Intent(MainActivity.this,Booksort.class));

break;

case R.id.bookmanage:

startActivity(new Intent(MainActivity.this,BookManage.class));

break;

case R.id.readnote:

startActivity(new Intent(MainActivity.this,Mynote.class));

break;

case R.id.number:

startActivity(new Intent(MainActivity.this,Number.class));

break;

case R.id.TJ:

startActivity(new Intent(MainActivity.this,Notenumer.class));

break;

case R.id.GY:

// startActivity(new Intent(MainActivity.this,GY.class));

}

}

//右上角关于

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch(item.getItemId()){

case R.id.menu\_about:

new AlertDialog.Builder(this)

.setTitle("关于")

.setMessage("软件版本：v1.0,开发者：底云飞")

.setPositiveButton("确定", null)

.show();

break;

default:

}

return true;

}

}

package com.example.diy.myapplication1;

import android.app.AlertDialog;

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import java.util.ArrayList;

import java.util.List;

public class LoginActivity extends AppCompatActivity {

private Button dl;

private Button zc;

private EditText username;

private EditText userpass;

private TextView state1;

private List<User> userList;

private List<User> dataList = new ArrayList<>();

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

dl= (Button) findViewById(R.id.dl);

zc=(Button)findViewById(R.id.zc1);

username= (EditText) findViewById(R.id.name1);

userpass= (EditText) findViewById(R.id.age1);

state1= (TextView) findViewById(R.id.state1);

dl.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String name=username.getText().toString().trim();

String pass=userpass.getText().toString().trim();

int result=SqliteDB.getInstance(getApplicationContext()).Quer(pass,name);

if (result==1)

{

//state1.setText("登录成功！");

startActivity(new Intent(LoginActivity.this,MainActivity.class));

}

else if (result==0){

state1.setText("用户名不存在！");

}

else if(result==-1)

{

state1.setText("密码错误！");

}

}

});

zc.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

startActivity(new Intent(LoginActivity.this,RegisterActivity.class));

}

});

}

//右上角关于

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch(item.getItemId()){

case R.id.menu\_about:

new AlertDialog.Builder(this)

.setTitle("关于")

.setMessage("软件版本：v1.0,开发者：底云飞")

.setPositiveButton("确定", null)

.show();

break;

default:

}

return true;

}

}

package com.example.diy.myapplication1;

import android.content.DialogInterface;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.support.v7.app.AlertDialog;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ListView;

import android.widget.SimpleAdapter;

import android.widget.Toast;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class Booksort extends AppCompatActivity {

private Button add;

private Button delete;

private Button update;

private Button find;

private OpenHelper sqlHelper;

private ListView listView;

List<BookofSort> bookofSortList;

private EditText et\_username,et\_password;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_booksort);

sqlHelper = new OpenHelper(this, "sqlite\_dbname", null, 2);

add=(Button)findViewById(R.id.btn\_find);

//添加

add.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(Booksort.this);

builder.setTitle("录入");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.booksort,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText et\_username = (EditText) view.findViewById(R.id.et\_username);

final EditText et\_password = (EditText) view.findViewById(R.id.et\_password);

builder.setView(view);//设置login\_layout为对话提示框

builder.setCancelable(false);//设置为不可取消

//设置正面按钮，并做事件处理

builder.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

BookofSort bookofSort=new BookofSort();

String name = et\_username.getText().toString().trim();

String pass = et\_password.getText().toString().trim();

bookofSort.setBookid(name);

bookofSort.setSortname(pass);

int result=SqliteDB.getInstance(getApplicationContext()).saveBooksort(bookofSort);

if (result==1){

Toast.makeText(Booksort.this, "录入成功", Toast.LENGTH\_SHORT).show();

}else if (result==-1)

{

Toast.makeText(Booksort.this, "编号已存在", Toast.LENGTH\_SHORT).show();

}

else

{

Toast.makeText(Booksort.this, "！", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder.show();//显示Dialog对话框

}

});

//查询

listView=(ListView)findViewById(R.id.listview1) ;

find=(Button)findViewById(R.id.btn\_insert);

find.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

try {

SQLiteDatabase db = sqlHelper.getWritableDatabase();

//游标查询每条数据

Cursor cursor = db.query("Booksort", null, null, null, null, null, null);

//定义list存储数据

List<Map<String, Object>> list = new ArrayList<Map<String, Object>>();

//适配器SimpleAdapter数据绑定

//错误:构造函数SimpleAdapter未定义 需把this修改为MainActivity.this

SimpleAdapter adapter = new SimpleAdapter(Booksort.this, list, R.layout.stu\_item,

new String[]{"bookid", "sortname"},

new int[]{R.id.book, R.id.book1});

//读取数据 游标移动到下一行

while(cursor.moveToNext()) {

Map<String, Object> map = new HashMap<String, Object>();

map.put( "bookid", cursor.getString(cursor.getColumnIndex("bookid")) );

map.put( "sortname", cursor.getString(cursor.getColumnIndex("sortname")) );

list.add(map);

}

listView.setAdapter(adapter);

}

catch (Exception e){

Log.i("exception", e.toString());

}

}

});

//删除

delete=(Button)findViewById(R.id.btn\_delete);

delete.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder1 = new AlertDialog.Builder(Booksort.this);

builder1.setTitle("删除");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.boosort1,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText et\_username = (EditText) view.findViewById(R.id.et\_username);

builder1.setView(view);//设置login\_layout为对话提示框

builder1.setCancelable(false);//设置为不可取消

builder1.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

BookofSort bookofSort=new BookofSort();

String name = et\_username.getText().toString().trim();

bookofSort.setBookid(name);

int result=SqliteDB.getInstance(getApplicationContext()).updateBooksort(bookofSort);

if (result==1){

Toast.makeText(Booksort.this, "删除成功", Toast.LENGTH\_SHORT).show();

}else if(result==-1)

{

Toast.makeText(Booksort.this, "编号不存在", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder1.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder1.show();//显示Dialog对话框

}

});

//修改

update=(Button)findViewById(R.id.btn\_change);

update.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(Booksort.this);

builder.setTitle("修改");

//通过布局填充器获login\_layout

View view = getLayoutInflater().inflate(R.layout.booksort,null);

//获取两个文本编辑框（密码这里不做登陆实现，仅演示）

final EditText et\_username = (EditText) view.findViewById(R.id.et\_username);

final EditText et\_password = (EditText) view.findViewById(R.id.et\_password);

builder.setView(view);//设置login\_layout为对话提示框

builder.setCancelable(false);//设置为不可取消

//设置正面按钮，并做事件处理

builder.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

BookofSort bookofSort=new BookofSort();

String name = et\_username.getText().toString().trim();

String pass = et\_password.getText().toString().trim();

bookofSort.setBookid(name);

bookofSort.setSortname(pass);

int result=SqliteDB.getInstance(getApplicationContext()).updateBooksort(bookofSort);

if (result==1){

Toast.makeText(Booksort.this, "修改成功", Toast.LENGTH\_SHORT).show();

}else if (result==-1)

{

Toast.makeText(Booksort.this, "编号不存在", Toast.LENGTH\_SHORT).show();

}

}

});

//设置反面按钮，并做事件处理

builder.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int i) {

}

});

builder.show();//显示Dialog对话框

}

});

}

}

1. **心得体会**

通过这次对Android应用程序读书笔记的开发，在开发过程中，强化了一些页面的跳转，页面之间的传值，以及关于数据库的增删改查，还有关于页面的布局内容的回顾，不过应用程序有时会崩溃的强制退出，需要重新卸载安装才可以使用，在同学以及网络的帮助下，解决了一些问题，不过还有有地方需要写成笔记内容记下来，方便以后回顾查看。