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： 郑贵峰

年级	15 级	专业（方向）	软件工程（移动信息工程） 互联网方向
学号	15352211	姓名	林苗
电话	13763360840	Email	554562948@qq.com
开始日期	2017.12.19	完成日期	2017.12.20

一、 实验题目

实验八 数据存储 （二）

二、 实现内容

实现一个生日备忘录，要求实现：

- 使用 SQLite 数据库保存生日的相关信息，并使得每一次运行程序都可以显示出已经存储在数据库里的内容；
- 使用 ContentProvider 来获取手机通讯录中的电话号码。

功能要求：

A. 主界面包含增加生日条目按钮和生日信息列表；

B. 点击“增加条目”按钮，跳转到下一个 Activity 界面，界面中包含三个信息输入框（姓名、生日、礼物）和一个“增加”按钮，姓名字段不能为空且不能重复；

C. 在跳转到的界面中，输入生日的相关信息后，点击“增加”按钮返回到主界面，此时，主界面中应更新列表，增加相应的生日信息；

D. 主界面列表点击事件：

- 点击条目：
  弹出对话框，对话框中显示该条目的信息，并允许修改；
  对话框下方显示该寿星电话号码（如果手机通讯录中有的话，如果没有就显示“无”）
  点击“保存修改”按钮，更新主界面生日信息列表。
- 长按条目：
  弹出对话框显示是否删除条目；
  点击“是”按钮，删除该条目，并更新主界面生日列表。

三、 课堂实验结果

（1） 实验截图

主页面初始界面

添加条目初始界面

当添加的姓名为空时，会有提示

当添加的姓名在数据库中已存在，会有提示

首页展示添加到数据库的各项信息

单击某一项，显示弹框显示对应信息

首页展示更新后的礼物信息

当通讯录中没有该联系人时，电话显示“无”

（2） 实验步骤及关键代码

1） 首先要写好布局。这次需要实现的布局有：主页面 activity_main.xml、listview 每一项的布局 listitem.xml、增加页面 activity_add.xml、对话框页面 dialoglayout.xml。布局方面比较简单，具体代码就不贴出来了。

2) 新建一个数据库类 MyDB.java，继承 SQLiteOpenHelper。重写构造函数。

```
public class MyDB extends SQLiteOpenHelper{

    private static final String DB_NAME = "Contacts.db"; //数据库名
    public static final String TABLE_NAME = "Contacts"; //表名
    private static final int DB_VERSION = 1; //数据库版本

    public MyDB(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, factory, version);
    }
    //构造函数，获取数据库
    public MyDB(Context context) {
        this(context, DB_NAME, null, DB_VERSION);
    }
```

在 onCreate 函数中创建数据库，将 name 设为主码，避免重复。

```
@Override
public void onCreate(SQLiteDatabase db) {
    //创建数据库
    String CREATE_TABLE = "CREATE TABLE if not exists "
            + TABLE_NAME
            + "(name TEXT PRIMARY KEY,"
            + "birth TEXT,"
            + "gift TEXT)";
    db.execSQL(CREATE_TABLE);

}
```

重写 onUpgrade 函数，虽然这次没有用到，但必须重写才能够实例化。

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

}
```

编写增加 insert 操作的函数，传入姓名、生日、礼物这三个参数，将数据插入到指定表中。

```
//增加
public void insert(String name, String birth, String gift) {
    SQLiteDatabase db = getWritableDatabase();//获取到当前可读写的数据库
    ContentValues values = new ContentValues();
    values.put("name", name);
    values.put("birth", birth);
    values.put("gift", gift);
    db.insert(TABLE_NAME, null, values);//插入数据
    db.close();
}
```

更新 update 函数，读取姓名、生日、礼物，对指定姓名那一行数据作更新。其中 whereClause 是 sql 语句中的 where 语句，whereArgs 是替换掉 whereClause 中占位符？的参数列表。

```
//更新
public void update(String name, String birth, String gift){
    SQLiteDatabase db = getWritableDatabase();//获取当前可读写的数据库
    String whereClause = "name = ?";//where语句
    String[] whereArgs = {name};//where语句的参数
    ContentValues values = new ContentValues();
    values.put("name",name);
    values.put("birth",birth);
    values.put("gift",gift);
    db.update(TABLE_NAME, values, whereClause, whereArgs);//更新
    db.close();
}
```

删除 delete 函数，根据名字来删除对应的元组。

```
//删除
public void delete(String name){
    SQLiteDatabase db = getWritableDatabase();//获取当前可读写的数据库
    String whereClause = "name = ?";//where语句
    String[] whereArgs = {name};//where语句的参数
    db.delete(TABLE_NAME, whereClause, whereArgs);//删除
    db.close();
}
```

查询指定名字条目，可用于在添加的 activity 中检测名字是否已经在数据库中存在。使用 rawQuery 直接执行查询语句。

```
//查找name条目
public Cursor ifexit(String name){
    SQLiteDatabase db = getWritableDatabase();//获取当前可读写的数据库
    String query = "select * from Contacts where name = \"" +name+"\"";//select语句
    Log.e("debug",query);
    return db.rawQuery(query,null);//执行select语句，返回结果集
}
```

查询整个表的所有条目。

```
//查询所有条目
public Cursor selectall(){
    SQLiteDatabase db = getWritableDatabase();//获取当前可读写的数据库
    String query = "select * from Contacts ";//select语句
    return db.rawQuery(query,null);//返回整个结果集
}
```

数据库操作定义部分全部完成。

3) 注册 Add 活动，编写 Add.java。

```
public class Add extends AppCompatActivity {


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add);
```

将要用到的控件与布局中的 ID 绑定起来。

```
//绑定布局控件
final EditText NAME = (EditText)findViewById(R.id.editname);
final EditText BIRTH = (EditText)findViewById(R.id.editbirth);
final EditText GIFT = (EditText)findViewById(R.id.editgift);
final Button ADDBUTTON = (Button)findViewById(R.id.add);
```

实现添加按钮的点击事件。获取输入框的内容。

```
//添加按钮点击事件
ADDBUTTON.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        String name = NAME.getText().toString(); //获取姓名
        String birth = BIRTH.getText().toString(); //获取生日
        String gift = GIFT.getText().toString(); //获取礼物
```

首先检测名字是否为空，若为空，则弹出 Toast 提示。

```
//名字为空
if(name.equals("")) {
    Toast.makeText(Add.this, "名字不能为空", Toast.LENGTH_SHORT).show();
}
```

如果不为空，那么连接数据库，传入名字参数，调用上面定义的 ifexit 查询是否有该名字的条目，如果有，则说明名字在数据库中已存在，弹出 Toast 提示；如果没有，那么就调用 insert 将信息加入到表中。

```
else{
    MyDB db = new MyDB(getBaseContext()); //取得数据库
    Cursor cursor = db.ifexit(name); //检查名字是否重复
    //如果cursor有条目，则说明名字在数据中已存在
    if (cursor.moveToFirst() == true) {
        Toast.makeText(Add.this, "名字不能重复", Toast.LENGTH_SHORT).show();
    }
    //名字在数据库中不存在，可以增添
    else {
        db.insert(name,birth,gift); //添加到数据库
        setResult(99, new Intent()); //返回一个99结果值
        finish(); //结束当前任务
    }
}
```

增加活动事件实现完毕。

4) 编写 MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    public List<Map<String, String>> datas = new ArrayList<Map<String, String>>();
    private ListView LV; //listview列表
    public SimpleAdapter adapter; //listview适配器
```

编写更新 ListView 列表函数。

```
//更新list列表函数
public void dataUpdate() {
    try {
        MyDB db = new MyDB(getBaseContext()); //连接数据库
        Cursor cursor = db.selectall(); //获取表中的所有数据
        datas  = new ArrayList<Map<String, String>>(); //新建datas队列
```

如果取得的结果不为空，那么就用下标获取到姓名、生日、礼物，使用 map 映射，将其加入到 datas 队列中，更新适配器，将适配器与 ListView 绑定起来。

```
        //如果结果为空，不做任何操作
        if (cursor == null) {
        }
        //如果结果不为空
        else {
            //取出所有结果
            while (cursor.moveToNext()) {
                String name1 = cursor.getString(0); //获取姓名
                String birth1 = cursor.getString(1); //获取生日
                String gift1 = cursor.getString(2); //获取礼物
                Map<String, String> map = new HashMap<String, String>();
                map.put("name", name1);
                map.put("birth", birth1);
                map.put("gift", gift1);
                datas .add(map);
            }
            //更新适配
            adapter = new SimpleAdapter(MainActivity.this, datas, R.layout.listitem,
                    new String[]{"name", "birth", "gift"},
                    new int[]{R.id.name, R.id.birth, R.id.gift});
            LV.setAdapter(adapter); //绑定适配器
        }
    } catch (SQLException e) {

    }
```

重写 onActivityResult 函数，根据发送码和接收码识别从 Add 返回到主界面时进行了添加操作，这个时候要调用 dataUpdate 函数，更新 ListView。

```
//识别activity的发送码和接收码
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    //如果是main->add, add点击添加->main
    if (requestCode == 9 && resultCode == 99) {
        //更新listview
        dataUpdate();
    }
}
```

编写 onCreate 函数。绑定 ListView，每次进入到 onCreate 函数时，更新 ListView。

```
LV = (ListView) findViewById(R.id.list);
dataUpdate(); //根据数据库的内容更新listview
```

实现添加条目按钮的点击事件，注意发送码。

```
final Button addbutton = (Button)findViewById(R.id.add); //添加条目按钮
addbutton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(MainActivity.this, Add.class);
        startActivityForResult(intent, 9);

    }
});
```

实现 ListView 的单击事件。要求点击时弹出对话框，可以修改信息。首先使用 LayoutInflator 类，将布局显示在对话框中。

```
LayoutInflater factory = LayoutInflater.from(MainActivity.this);
View newView = factory.inflate(R.layout.dialoglayout, null); //绑定ui界面
AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
```

绑定布局的控件 ID，设置各项显示的内容。

```
final TextView nameTV = (TextView) newView.findViewById(R.id.editname);
final  EditText birthET = (EditText) newView.findViewById(R.id.editbirth);
final EditText giftET = (EditText) newView.findViewById(R.id.editgift);
final TextView phoneTV = (TextView) newView.findViewById(R.id.editphone);

nameTV.setText(datas.get(i).get("name"));
birthET.setText(datas.get(i).get("birth"));
giftET.setText(datas.get(i).get("gift"));
phoneTV.setText("无");
```

接下来要查找联系人列表，看联系人列表中是否有该联系人的电话，如果有就显示在电话那一栏。首先要在 AndroidManifest.xml 中声明权限。

```
<uses-permission android:name="android.permission.READ_CONTACTS"/>
```

返回刚才的 ListView 点击事件。用 cursor 获得联系人列表。

```
//获取联系人列表
Cursor cursor = getContentResolver().query(ContactsContract.Contacts.CONTENT_URI,
        null, null, null, null);
```

读取每个联系人，首先用 getColumnIndex 得到名字在表 ContactsContact.Contact 中的下标，然后用 getString 访问对应下标得到当前条目联系人的名字。

```
while (cursor.moveToNext() ) {
    //得到显示出来的名字在ContactsContact.Contact中的下标
    int nameIndex = cursor.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME);
    //读取到名字
    String name = cursor.getString(nameIndex);
```

如果联系人名字与对话框中的名字一样，那么就读取其电话号码。同样，必须先用 getColumnIndex 获得号码在表 ContactsContact.Contact 中的下标，然后用下标访问得到电话号码，读取第一个电话号码，显示在电话那一栏。

```
if (name.equals(nameTV.getText().toString())) {
    //获取ID
    String contactId = cursor.getString(cursor.getColumnIndex(ContactsContract.Contacts._ID));
    //根据联系人ID查询对应的电话号码
    Cursor phoneNumbers = getContentResolver().query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
            null, ContactsContract.CommonDataKinds.Phone.CONTACT_ID + " = "
            + contactId, null, null);
    String strPhoneNumber = "";
    //获取第一个电话号码
    if (phoneNumbers.moveToNext())
    {
        strPhoneNumber = phoneNumbers.getString(phoneNumbers.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUM
    }

    phoneNumbers.close();
    phoneTV.setText(strPhoneNumber);//显示
}
```

设置自定义对话框的内容，如果点击保存修改，那么调用 update 函数进行更新，且要调用 dataUpdate 刷新 ListView 内容。点击放弃修改则不进行任何操作。

```
// 自定义对话框
builder.setView(newView);
builder.setTitle("改啥(・ω・)✧");
builder.setPositiveButton("保存修改", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        MyDB db = new MyDB(getBaseContext());
        //更新数据库
        db.update(nameTV.getText().toString(),birthET.getText().toString(),giftET.getText().toString());
        //更新listview
        dataUpdate();
    }
});
builder.setNegativeButton("放弃修改", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {

    }
});
builder.show();
```

实现 ListView 的长按事件。获取下标，然后调用 delete 语句删除数据库表中对应的条目，可以直接删除 datas 队列相应的项，然后用适配器更新改动，可以省去对全 ListView 表的更新。

```
// listview的长按事件
LV.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {
    @Override
    public boolean onItemLongClick(AdapterView<?> parent, View view, final int position, long id) {
        AlertDialog.Builder message = new AlertDialog.Builder(MainActivity.this);

        message.setMessage("是否删除");
        message.setPositiveButton("是", (dialogInterface, i) -> {
                //删除
                MyDB db = new MyDB(getBaseContext());
                db.delete(datas.get(position).get("name"));
                // 删除listview中的对应数据
                datas.remove(position);
                adapter.notifyDataSetChanged();
        });
        message.setNegativeButton("否", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialogInterface, int i) {
            }
        });
        message.create().show();
        return true;
    }
});
```

至此，所有功能全部实现。

（3） 实验遇到的困难及解决思路

1) 在写 MyDB 类时，提示“There is no default constructor available in ‘android.database.sqlite.SQLiteOpenHelper’”

解决思路：翻译过来应该是说没有构造函数，因此在网上看了别人的例子，加了构造函数之后就可以了。

```
public MyDB(Context context, String name, SQLiteDatabase.CursorFactory factory,
                    int version) {
    super(context, name, factory, version);
}
```

2) 查找名字是否存在时，闪退。

解决思路：用 Log.e 将查询语句打印出来，发现我的查询语句是这样的：

```
12-19 22:12:26.060 2043-2043/com.example.linm.lab8 E/debug: select * from Contacts where name = 略略略
```

于是就猜想可能是名字没有用双引号括起来，修改了查询语句，将名字前后用双引号括起来，就没问题了。

3) listview 超出屏幕无法滑动显示

解决方法：看了之前购物车的布局，发现整体是用 Relativelayout，这次是用约束布局，将约束布局改成线性布局 LinearLayout 就好了。

四、 课后实验结果

在增加内容的时候，如果发现全部输错了想重新输入，需要一个个删掉，比较麻烦，因此简单设置了一个清空按钮，点击即可清空全部内容。界面如下：

在 Add.java 中增加如下代码：

```
final Button CLEARBUTTON = (Button)findViewById(R.id.clear);
```

```
CLEARBUTTON.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        NAME.setText("");
        BIRTH.setText("");
        GIFT.setText("");
    }
});
```

五、 实验思考及感想

本次实验用到了数据库的知识。由于这学期也修学了数据库这门课程，以及在 web 中也有涉及到数据库的知识和大量练习，因此对简单的增、删、改语句的编写操作还是比较熟练的。虽然具体操作还是有细微差别，但是看了网上的一些例子之后还是能够触类旁通的，因此数据库这方面的实现感觉不会很困难。

这次学到了一个新的知识点，就是返回一个活动时，比如在添加界面点击添加按钮之后，会先 finish 掉这个任务，再返回 MainActivity。一开始我发现添加的条目不能够直接在 ListView 中显示出来，尽管我在 onCreate 函数中有调用更新 ListView 列表的函数，但发现没有起到作用，只有在退出程序，重新打开时，才会发现条目被添加进去了。看了往届师姐使用的方法，是在从 MainActivity 跳转到 Add 时，发送一个发送码，在 finish 掉 Add 时，调用 setResult 返回一个接收码，在 MainActivity 中，重写 onActivityResult 函数，在指定发送码和接收码时调用更新 ListView 的函数。百度之后了解到这种方法常用于识别 Activity 之间的数据传递，其中 requestCode 常用于识别是由哪个 Activity 返回，resultCode 是子 Activity 返回的，这样就能对不同情况进行不同操作，可以灵活用于其它很多种情况。

六、 参考资料

使用 Content Provider 得到联系人信息
http://blog.csdn.net/wangkuifeng0118/article/details/7023642

SQLiteDatabase 中 query、insert、update、delete 方法参数说明
https://www.cnblogs.com/maxinliang/archive/2013/01/22/2871474.html

onActivityResult 的用法
http://blog.csdn.net/double2hao/article/details/50281103