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：

年级	15	专业（方向）	移动互联网
学号	15352155	姓名	赖贤城
电话	13727024851	Email	754578682@qq.com
开始日期	2017/12/19	完成日期	2017/12/19

一、 实验题目

数据存储（一）

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。

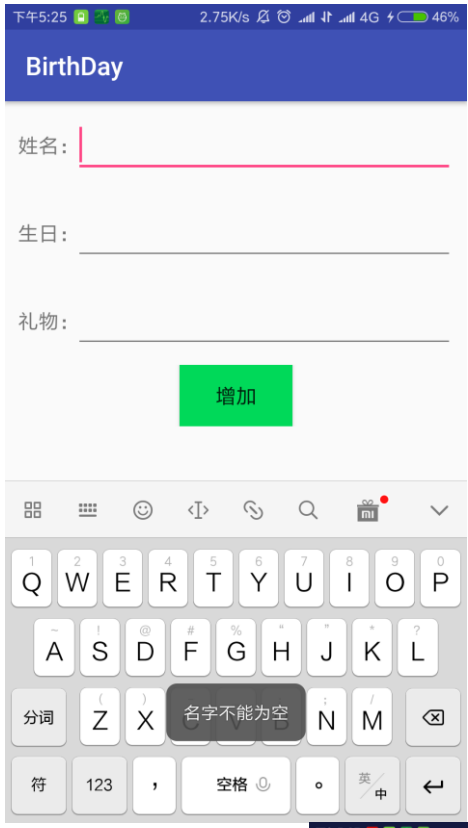
长按条目：

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



（2） 实验步骤以及关键代码

1. 创建数据库，建表，以及实现对外接口

SQLiteOpenHelper 是用于操纵 SQLite 的类，通过重写它来构建自己的一个数据库

```
public class BirthDB extends SQLiteOpenHelper {

    private static final String TABLE_NAME = "Contacts";
    Context context;

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

根据分析，这次任务的每条记录需要包含“名字”、“生日”、“礼物”三项，以名字作为主键，据此可以创建一个表存储这些记录：

```
@Override
public void onCreate(SQLiteDatabase db) {
    String CREATE_TABLE = "create table " + TABLE_NAME
            + "(name text , "
            + "birth text , "
            + "gift text);";
    db.execSQL(CREATE_TABLE);
}
```

接着实现这种操作数据库的函数以供后面的使用，这样可以让 activity 中的逻辑简单一点。根据分析，

- 增加记录界面需要：
  a) 判断某个名字是否存在

```
public boolean isExits(String name) {
    SQLiteDatabase db = getWritableDatabase();
    Cursor cursor = db.rawQuery( sql: "select * from " + TABLE_NAME + " where name='"
            + name + "'",  selectionArgs: null);
    boolean isExits = false;
    if (cursor.moveToNext()) {
        isExits = true;
    }
    db.close();
    return isExits;
}
```

  b) 插入新的记录

```
public void insert(String name, String birth, String gift) {
    SQLiteDatabase db = getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put("name", name);
    values.put("birth", birth);
    values.put("gift", gift);
    db.insert(TABLE_NAME,  nullColumnHack: null, values);
    db.close();
}
```

- 主界面需要：
  a) 删除某条记录

```
public void delete(String name) {
    SQLiteDatabase db = getWritableDatabase();
    String whereClause = "name = ?"; // 主键列名 = ?
    String[] whereArgs = {name}; // 主键的值
    db.delete(TABLE_NAME, whereClause, whereArgs);
    db.close();
}
```

  b) 更新某条记录

```java
public void update(String name, String birth, String gift) {
    SQLiteDatabase db = getWritableDatabase();
    String whereClause = "name = ?"; // 主键列名 = ?
    String[] whereArgs = {name}; // 主键的值
    ContentValues values = new ContentValues();
    values.put("birth", birth);
    values.put("gift", gift);
    db.update(TABLE_NAME, values, whereClause, whereArgs);
    db.close();
}
```

c) 查询全部的记录

```java
public ArrayList<BirthdayBean> queryAll() {
    SQLiteDatabase db = getWritableDatabase();
    Cursor cursor = db.rawQuery( sql: "select * from " + TABLE_NAME,  selectionArgs: null);
    ArrayList<BirthdayBean> birthdayBeans = new ArrayList<>();
    while (cursor.moveToNext()) {
        String name = cursor.getString(cursor.getColumnIndex( columnName: "name"));
        String birthday = cursor.getString(cursor.getColumnIndex( columnName: "birth"));
        String gift = cursor.getString(cursor.getColumnIndex( columnName: "gift"));
        BirthdayBean bean = new BirthdayBean(name, birthday, gift,  tel: "");
        birthdayBeans.add(bean);
    }
    db.close();
    return birthdayBeans;
}
```

◆ 除此之外，主界面还需要读取通讯录，获取电话号码，同样将其写在数据库操作当中

```java
public String queryTel(String name) {
    String tel = "无";
    ContentResolver resolver = context.getContentResolver();
    Cursor cursor = context.getContentResolver().query(ContactsContract.Contacts.CONTENT_URI,
            projection: null,  selection: null,  selectionArgs: null,  sortOrder: null);
    while (cursor != null && cursor.moveToNext()) {
        //获得联系人ID
        String id = cursor.getString(cursor.getColumnIndex(android.provider.ContactsContract.Contacts._ID));
        //获得联系人姓名
        String tel_name = cursor.getString(cursor.getColumnIndex(android.provider.ContactsContract.Contacts.DISPLAY_NAME));
        //获得联系人手机号码
        Cursor phone = resolver.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,  projection: null,
                selection: ContactsContract.CommonDataKinds.Phone.CONTACT_ID + "=" + id,  selectionArgs: null,  sortOrder: null);

        StringBuilder sb = new StringBuilder("");
        if (tel_name.equals(name)) {
            while (phone.moveToNext()) { //取得电话号码(可能存在多个号码)
                int phoneFieldColumnIndex = phone.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER);
                String phoneNumber = phone.getString(phoneFieldColumnIndex);
                sb.append(phoneNumber + " ");
            }
            tel = sb.toString();
            phone.close();
            break;
        }
    }
    cursor.close();
    return tel;
}
```

2. 增加生日信息界面

这个界面比较简单，因为上面的步骤中数据库已经实现了将数据插入数据库表的函数以及判断用户是否已经存在的函数，因此这里只需要读取用户的输入，然后

1) 判断是否为空（使用 TextUtils）
2) 判断数据库是否已经有该用户
3) 插入用户输入的新记录到数据库

```
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_add);
    Name = findViewById(R.id.tv_name);
    Birthday = findViewById(R.id.tv_birthday);
    Gift = findViewById(R.id.tv_gift);
    btAdd = findViewById(R.id.bt_new);
    btAdd.setOnClickListener(new View.OnClickListener()
    {
        public void onClick(View v)
        {
            if (TextUtils.isEmpty(Name.getText().toString()))
            {
                Toast.makeText( context: AddActivity.this, text: "名字不能为空", Toast.LENGTH_SHORT).show();
                return;
            }
            boolean isExits = db.isExits(Name.getText().toString());

            if (isExits)
            {
                Toast.makeText( context: AddActivity.this, text: "名字不能重复", Toast.LENGTH_SHORT).show();
                return;
            }
            db.insert(Name.getText().toString(), Birthday.getText().toString(), Gift.getText().toString());
            finish();
        }
    });
}
```

① ② ③

3. 实现主页面

a. *生日信息列表，使用 RecyclerView*

由于本次重点不是 RecyclerView，因此这里简要解释而已。

实现 adapter，在 onBindViewHolder 中填入数据，与之前的方法一样，这里点击和长按调用的是传入的接口，接口将在使用 RecyclerView 的时候实现

```
@Override
public void onBindViewHolder(final BirthVH holder, int position) {
    //直接在这里设置数据
    String name = BirthList.get(position).name;
    String birthday = BirthList.get(position).birthDay;
    String gift = BirthList.get(position).gift;
    holder.tvName.setText(name);
    holder.tvBirthday.setText(birthday);
    holder.tvGift.setText(gift);
    holder.itemView.setOnClickListener((v) -> {
            itemClickListener.onClick(holder.getAdapterPosition());
    });
    holder.itemView.setOnLongClickListener((v) -> {
            itemClickListener.onLongClick(holder.getAdapterPosition());
            return true;//消费长按事件
    });
}
```

上述代码中的接口定义

```
public interface recyItemClickListener {
    void onClick(int position);
    void onLongClick(int position);
}
```

在主界面中使用 adapter，实现点击和长按接口，下图的两个 show 函数将在下面详细解释

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    btNew = findViewById(R.id.bt_new);
    btNew.setOnClickListener((v) → {
            Intent i = new Intent( packageContext: MainActivity.this, AddActivity.class);
            startActivity(i);
    });
    recyclerView = findViewById(R.id.rc_items);
    recyItemClickListener itemClickListener = new recyItemClickListener() {
        @Override
        public void onClick(int position) { showChangeDialog(position); }          点击事件
        @Override
        public void onLongClick(int position) { showDeleteDialog(position); }      长按事件
    };
    birthAdapter = new BirthAdapter( context: this, birthdayBeans , itemClickListener);
    recyclerView.setLayoutManager(new LinearLayoutManager( context: this));
    recyclerView.setAdapter(birthAdapter);
}
```

上面将 birthdayBeans 与 adapter 相关联之后，下一步是将数据填入 birthdayBeans 之中，这样 RecyclerView 才有数据可以显示，这里就用到了数据库实现的查询全部信息的函数了

```
private void refreshView(){
    ArrayList<BirthdayBean> tmpList = db.queryAll();
    birthdayBeans.clear();
    birthdayBeans.addAll(tmpList);
    birthAdapter.notifyDataSetChanged();
}
```

b. 点击事件调用的 showChangeDialog 函数

A. 使用 LayoutInflater 将布局 inflate 成 view，通过 setView 函数将这个 view 设置到对话框

```
private void showChangeDialog(final int position){
    final AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
    LayoutInflater factor = LayoutInflater.from(MainActivity.this);
    View view_in = factor.inflate(R.layout.dialog_layout,  root: null);
    builder.setView(view_in);
```

B. 将数据填入对话框

- 基本的信息是通过获取 birthdayBeans 对应位置的数据填入就行了。
- 麻烦的是电话这个数据的填入，这里使用了接口的技术，在接口的实现中调用查询电话的函数作为待会儿的回调函数。经过试验发现，这个查询操作比较耗时，因此这里新开了一个子线程来执行查询操作，然后通过 post 函数更新界面。 这个接口的调用在 requestAndGetTel 这个函数中，下面将解释这个函数。

```
// *********填入数据***********
final BirthdayBean bean = birthdayBeans.get(position);
final TextView tv_name = view_in.findViewById(R.id.tv_name);
tv_name.setText(bean.name);
final TextView tv_tel = view_in.findViewById(R.id.tv_tel);
telInterface = (TelInterface) () → {
        Thread t = new Thread((Runnable) () → {
                final String tel = db.queryTel(bean.name);
                tv_tel.post(() → { tv_tel.setText(tel); });
        });
        t.start();
};
requestAndGetTel();
final EditText tv_birthday = view_in.findViewById(R.id.tv_birthday);
tv_birthday.setText(bean.birthDay);
final EditText tv_gift = view_in.findViewById(R.id.tv_gift);
tv_gift.setText(bean.gift);
```

➢ 动态请求权限与执行回调函数，当授权成功的时候就执行上面定义的回调函数，也就是进行查询电话的操作

```
private void requestAndGetTel(){
    if (ContextCompat.checkSelfPermission( context: this, android.Manifest.permission.READ_CONTACTS)
            != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions( activity: this, new String[]{android.Manifest.permission.READ_CONTACTS}, requestCode: 1);
    } else {//已授权
        telInterface.setTel();
    }
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    if (requestCode == 1) {
        if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            //权限申请成功
            telInterface.setTel();
        } else {
            Toast.makeText( context: MainActivity.this, text: "获取联系人的权限申请失败", Toast.LENGTH_SHORT).show();
            finish();
        }
    }
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
}
```

执行回调函数

➢ 当用户选择保存修改的时候，调用数据库的 update 函数修改数据库中的记录，然后调用 refreshView 函数重新从数据库获取数据刷新 RecyclerView

```
builder.setPositiveButton("保存修改", (dialog, which) -> {
        String birthday = tv_birthday.getText().toString();
        String gift = tv_gift.getText().toString();
        db.update(bean.name, birthday, gift);   //数据库更新数据
        refreshView();      //刷新界面
});
builder.setNegativeButton("放弃修改", (dialog, which) -> {
        dialog.cancel();
});
builder.show();
```

c. *长按事件的 showDeleteDialog 函数*

删除的操作比较简单，当用户选择删除的时候，调用数据库的 delete 函数删除数据库中对应的记录，然后调用 refreshView 函数重新从数据库获取数据刷新 RecyclerView 就可以了

```
private void showDeleteDialog(final int position){
    final AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
    builder.setTitle("是否删除？");

    builder.setPositiveButton("是", (dialog, which) -> {
            db.delete(birthdayBeans.get(position).name);    //数据库删除数据
            refreshView();      //刷新界面
    });
    builder.setNegativeButton("否", (dialog, which) -> { dialog.cancel(); });
    builder.show();
}
```

（3） 实验遇到困难以及解决思路

问题：假设 birthdayBeans 是传入的 RecyclerView 适配器的数组，更新数据的时候使用下面这种方式却发现 RecyclerView 的 item 没有一点变化

```
birthdayBeans = db.queryAll();
birthAdapter.notifyDataSetChanged();
```

调试分析：经过调试发现上面的语句执行之后 birthdayBeans 的内容确实变了，但是 adapter 里面的 list 并没有变！ 一下子明白过来了，adapter 里面的 list 指向的是**旧的** birthdayBeans 指向的内容，而 `birthdayBeans = db.queryAll()` 之后虽然 birthdayBeans 是更新了，但是其实变得是指向的地址，虽然它指向了新地址，但是

adapter 里面的 list 指向的旧地址的内容并没有变，这就导致了 RecyclerView 的内容并没有更新

解决：查询数据的时候不要直接 birthdayBeans 指向新的内容，而是将新的内容复制一份到旧的地址，就像下面这样

```
ArrayList<BirthdayBean> tmpList = db.queryAll();
birthdayBeans.clear();
birthdayBeans.addAll(tmpList);
birthAdapter.notifyDataSetChanged();
```

四、 课后实验结果

无

五、 实验思考及感想

1. 安卓内置的这个 SQLite 很友好，对于 sql 语法不熟的人提供了函数接口以供调用，甚至不需要用户名和口令就能直接操作，对于一些小型的需要结构化的数据，真的很方便
2. 通过分析解决了“实验遇到困难以及解决思路”，对于 java 中的“指针”理解的更深了，以后要用好 final 这个关键词，再也不能在不该改地址的地方乱改地址了

作业要求：

1. 命名要求: 学号_姓名_实验编号，例如 15330000_林 XX_lab1。
2. 实验报告提交格式为 pdf。
3. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。