中山大学数据科学与计算机学院软件工程（移动信息工程）本科生实验报告

移动应用开发实验报告

姓名	学号	班级	电话	邮箱	日期
蔡烨	15352010	1501班	15989010669	caiy29@mail2.sysu.edu.cn	2017/12/18

1. 实验题目

数据存储（二）

2. 实验目的

- 学习SQL数据库的使用
- 学习ContentProvider的使用
- 复习Android界面编程

3. 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。

4. 实验过程

1. 编写四个xml文件，分别是主页面、主页面列表的item、添加页面和对话框样式。主要的注意点在于，最好不要用weight去定义三个列的宽度，因为这样写内容过长时不会换行，造成不美观。
2. 编写数据库的类myDB.class

   这里必须写一个构造函数super父类的，否则会报错。

   然后就是onCreate里面建表，再override一下onUpgrade就可以了。

```
public class myDB extends SQLiteOpenHelper {
    private static final String DB_name = "myDB.db";
    private static final String TABLE_name = "birthday";
    private static final int version = 1;

    public myDB(Context context){
        super(context,DB_name,null,version);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        String CREATE_TABLE = "create table if not exists " + TABLE_name + "(name text primary key, date text, gift text)";
        try{
            db.execSQL(CREATE_TABLE);
        }catch(Exception e){
            System.out.println(e.getMessage());
        }
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) { }
}
```

3. 编写添加页面的activity：addItem.class

   这个类主要在于对“增加”按钮的监听。在监听事件里，首先判断获取到的name输入框里的内容是否为空，是的话toast，否则再判断表中是否存在相同的name，是的话再toast，如果没有，再往表中插入，插入的数据来自三个输入框的内容。插入之后先发送一个resultcode回去给mainActivity，再finish掉当前的，这样就会回到主页并且更新页面（在onActivityResult里处理）。

```
add_btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(add_name.length()==0){
            Toast.makeText(addItem.this,"名字不能为空", Toast.LENGTH_SHORT).show();
        } else {
            String name = add_name.getText().toString();
            String date = add_date.getText().toString();
            String gift = add_gift.getText().toString();

            myDB db = new myDB(getBaseContext());
            SQLiteDatabase sqlite = db.getWritableDatabase();
            Cursor cursor = sqlite.query(table_name,null,"name=?",new String[]{name},null,null,null);
            if(cursor.moveToNext()){//存在名字相同的条目
                Toast.makeText(addItem.this,"该名字已被添加过",Toast.LENGTH_SHORT).show();
            }else{ //插入条目
                String sql = "insert into " + table_name + "(name,date,gift) values('" + name + "','" + date + "','" + gift + "')";
                try{
                    sqlite.execSQL(sql);
                    sqlite.close();
                }catch (Exception e){
                    System.out.println(e.getMessage());
                }
                setResult(1,new Intent());
                finish();
```

4. 编写主页面的activity：MainActivity.class
   1. 更新列表updateUI

      从数据库的表里取出所以的数据，放进一个list里面。然后写一个适配器，将list里的内容一个个装载进去。需要注意的是，每次调用这个函数，这个list都必须重新分配内存，不然就会出现添加完一条数据返回时，前面的所有数据都被再次添加的情况，或者是修改完该数据被再次添加。

```
//更新界面列表
private void updateUI(){
    try{
        myDB db = new myDB(getBaseContext());
        SQLiteDatabase sqlite = db.getWritableDatabase();
        //查询数据
        Cursor cursor = sqlite.rawQuery("select * from "+table_name,null);
        data = new ArrayList<Map<String, String>>();
        if(cursor != null){
            while(cursor.moveToNext()){
                Map<String,String> map = new HashMap<String,String>();
                map.put("name",cursor.getString(0));
                map.put("date",cursor.getString(1));
                map.put("gift",cursor.getString(2));
                data.add(map);
            }
            //列表适配器
            adapter = new SimpleAdapter(MainActivity.this,data,R.layout.item, new String[]{"name","date","gift"},
                    new int[]{R.id.item_name,R.id.item_date,R.id.item_gift});
            lv.setAdapter(adapter);
        }
    }catch (Exception e){
        System.out.println(e.getMessage());
```

2. 为增加按钮添加监听事件，用intent实现页面跳转。为list列表增加item的长按监听事件，弹出对话框，具体实现方法跟之前实验写的对话框是一模一样的，不再赘述。需要注意的是用户确认删除后，不仅要从高数据库的表里删除，还需要更新当前存储数据内容的list，将其删除，才能同步页面和数据库（不然就得再调用一次updateUI）。

3. 为list列表的item的点击添加监听事件，弹出对话框。这一次的对话框需要加载的是前面写好的对话框的layout，而且在获取控件的时候，不能再直接findViewById，而必须在这个函数前声明是哪个view，否则会默认为mainActivity对应的xml，而出现空对象的问题。

```
lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, final int position, long id) {
        LayoutInflater factory = LayoutInflater.from(MainActivity.this);
        View newview = factory.inflate(R.layout.dialog,null);
        AlertDialog.Builder dialog = new AlertDialog.Builder(MainActivity.this);

        final TextView edit_name = (TextView) newview.findViewById(R.id.edit_name);
        final EditText edit_date = (EditText) newview.findViewById(R.id.edit_date);
        final EditText edit_gift = (EditText) newview.findViewById(R.id.edit_gift);
        TextView phone = (TextView) newview.findViewById(R.id.phone);

        System.out.println(edit_name);
        edit_name.setText(data.get(position).get("name"));
        edit_date.setText(data.get(position).get("date"));
        edit_gift.setText(data.get(position).get("gift"));
```

4. 因为页面的数据是可以改变的，因此在用户按下对话框确认按钮的时候，保存进表的数据是要重新从控件上获取。

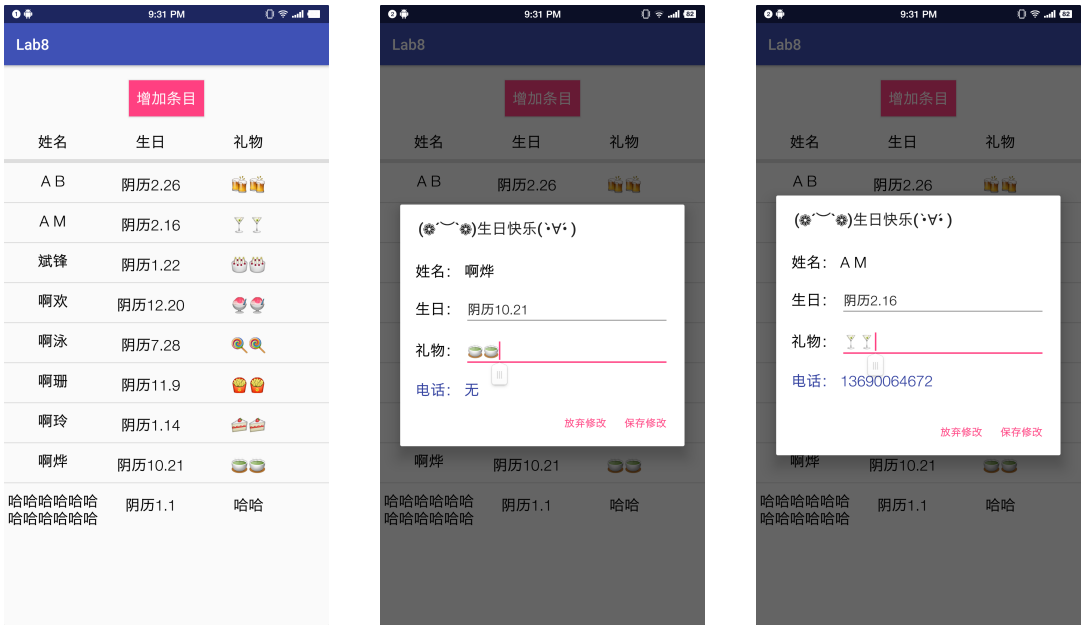
5. 电话号码的获取：注册之后，首先要判断有没有权限，没有就申请。然后用getContentResolver().query()方法读取联系人，返回的是一个cursor。采用遍历的方法，用getString(cursor.getColumnIndex())获取联系人的名字，对比看是不是我们要查找的名字。如果一样，再判断该信息中是否有电话号码，有的话一个个拿出来，存到一个string里，最后将这个string写进电话号码那个TextView。

```java
//获取电话号码
if(ContextCompat.checkSelfPermission(MainActivity.this, Manifest.permission.READ_CONTACTS)!= PackageManager.PERMISSION_GRANTED){
    ActivityCompat.requestPermissions(MainActivity.this, new String[]{Manifest.permission.READ_CONTACTS},0);
}
//用getContentResolver方法读取联系人
String number = "";
Cursor cur = getContentResolver().query(ContactsContract.Contacts.CONTENT_URI, null,null,null,null);
while (cur.moveToNext()){
    if(cur.getString(cur.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME)).equals(edit_name.getText().toString())){
        //判断该联系人的信息中，是否有电话号码
        if(Integer.parseInt(cur.getString(cur.getColumnIndex(ContactsContract.Contacts.HAS_PHONE_NUMBER)))>0){
            //取出该联系人信息中的电话号码
            Cursor phones = getContentResolver().query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,null,
                    ContactsContract.CommonDataKinds.Phone.CONTACT_ID + "=" + cur.getString(cur.getColumnIndex(ContactsContract.Contacts._ID)), null,null);
            while(phones.moveToNext()){
                number += phones.getString(phones.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER)) + "\n";
            }
            phones.close();
        }
    }
}
cur.close();
//如果手机通讯录中没有对应的联系人则将号码设为无
if(number.equals(""))
    number = "无";
phone.setText(number);
```

获取权限

5. 实验结果

从左到右：主页面、修改框（无电话）、修改框（有电话）



从左到右：空字段添加、重复添加、正常添加

从左到右：添加后、删除

6. 实验思考及感想

因为这个星期在做web大作业，数据库方面刚好是我负责写的，加上之前安卓的期中项目也已经解除过，所以关于数据库的操作语句用起来比较熟悉。这次的新的知识点其实是获取电话号码那个了吧，逻辑上其实不复杂，只是那些语句如果不借助外力是写不出来的，不过因为有TA的文档和搜索引擎，所以还是不难解决的。其他实验中遇到的问题，都基本上写进实验过程了。