安卓实验报告

课程：手机平台应用开发

任课教师：刘宁

K	V
年级	15级
专业（方向）	计应
学号	15331421
姓名	郑佳锋
电话	137193254712
Email	1773643139@qq.com
开始日期	12.7
完成日期	12.9

实验题目

数据存储（二）

实现内容

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

实验结果

实验截图

应用主界面：

点击添加条目：

新增条目：

点击添加并返回主界面：

点击条目弹出对话框：


上午9:10
Birthday Reminder
增加条目
ヽ(*￣∀￣)ノ~ ★恭喜发财★ ~
姓名:郑佳锋
生日:326
礼物:
电话:137 1932 5472
放弃修改
保存修改

进行编辑并保存：


上午9:10
Birthday Reminder
增加条目
ヽ(*￣∀￣)ノ~ ★恭喜发财★ ~
姓名:郑佳锋
生日:211
礼物:
电话:137 1932 5472
放弃修改
保存修改

回到主界面：

增加另一条记录：

删除第一条记录，退出应用后重新打开：

实验步骤及关键代码

1.主界面的设计：

首先是一个按钮，然后是一个ListView。ListView使用simpleAdapter，每一条目的布局是三个TextView，分别显示姓名、生日、礼物。设置点击事件弹出对话框，长按事件删除。关键代码如下：

```
list = (ListView) findViewById(R.id.list);
String[] name = new String[]{"姓名"};
String[] birth = new String[]{"生日"};
String[] gift = new String[]{"礼物"};
for (int i = 0; i < name.length; i++) {
            Map<String, String> temp = new LinkedHashMap<String, String>();
            temp.put("name", name[i]);
            temp.put("birth", birth[i]);
            temp.put("gift", gift[i]);
            data.add(temp);
        }
simpleAdapter = new SimpleAdapter(MainActivity.this, data, R.layout.item,
                new String[]{"name", "birth","gift"}, new int[]{R.id.name, R.id.birth,
R.id.gift});
list.setAdapter(simpleAdapter);
```

2.新增界面的设计：

三个EditText加上三个TextView，其中，生日的TextView只能输入数字。底部是一个“增加”按钮。

每次点击按钮后，对“姓名”输入框进行检查，若为空，弹出提示；否则，再进行其他操作。

```
String name_str = name.getText().toString();
String birth_str = birth.getText().toString();
String gift_str = gift.getText().toString();
if(name_str.equals("")){
           Toast.makeText(AddItem.this, "名字为空，请完善", Toast.LENGTH_SHORT).show();
     }
    else
    ……
```

3.主界面和新增界面的数据传递：

新增界面编辑完成后，点击增加按钮需要返回主界面，并将数据显示在主界面。最开始打算使用EventBus实现。但是想想EventBus使用太麻烦了，这里完全可以使用startActivityForResult来写。在主界面重写onActivityResult方法对新增界面传过来的Intent进行处理即可。

新增界面：

```
Intent mIntent = new Intent();
mIntent.putExtra("name", name_str);
mIntent.putExtra("birth", birth_str);
mIntent.putExtra("gift", gift_str);
setResult(0, mIntent);
AddItem.this.finish();
```

主界面：

```
@Override
    protected void onActivityResult(int requestCode, int resultCode, Intent intent) {
        if (intent != null && intent.hasExtra("name")) {
            String name_str = intent.getStringExtra("name");
            String birth_str = intent.getStringExtra("birth");
            String gift_str = intent.getStringExtra("gift");
            Map<String, String> temp = new LinkedHashMap<String, String>();
            temp.put("name", name_str);
            temp.put("birth", birth_str);
            temp.put("gift", gift_str);
            ……
        }
}
```

4.对话框的设计：

点击条目后，弹出自定义对话框。使用inflate加载布局文件。布局文件中首先是一个TextView，显示”恭喜发财“。

下面是四行，分别显示姓名、生日、礼物、电话以及它们的输入框。因为姓名、电话不可改，所以使用TextView。加载到布局后，需要将条目中的数据传到对话框来。最后设计放弃修改和保存修改按钮。

```
LayoutInflater factor = LayoutInflater.from(MainActivity.this);
final View layout = factor.inflate(R.layout.dialoglayout, null);
final AlertDialog.Builder alertDialog = new AlertDialog.Builder(MainActivity.this);
alertDialog.setView(layout);
((TextView) layout.findViewById(R.id.name_input)).setText(data.get(position).get("name"));
((EditText) layout.findViewById(R.id.birth_input)).setText(data.get(position).get("birth"));
((EditText) layout.findViewById(R.id.gift_input)).setText(data.get(position).get("gift"));
```

5.通讯录的读取：

首先在AndroidManifest.xml文件中申请读取权限。

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

注意安装完应用完需要到设置中授权读取通讯录的权限，否则可能会闪退。

读取电话号码时，首先使用query函数获得通信列表。再通过游标进行遍历。使用getString获得每一条记录的姓名，若与查询姓名一致并且拥有电话号码，就获取它的Contacts._ID。 最后，再通过这个ID找到对应的号码。记得每个游标使用完后都要close。

```
Cursor cursor = getContentResolver().query(ContactsContract.Contacts.CONTENT_URI,
                          null, null, null, null);
int flag = 0;
String Number = "";
while(cursor.moveToNext()){
          String name = cursor.getString(cursor.getColumnIndex(
                               ContactsContract.Contacts.DISPLAY_NAME));
          int isHas=Integer.parseInt(cursor.getString(
          cursor.getColumnIndex(ContactsContract.Contacts. HAS_PHONE_NUMBER )));
          if(name.equals(data.get(position).get("name")) && isHas == 1){
               String contactId = cursor.getString(cursor.getColumnIndex(
                                ContactsContract.Contacts._ID));
          Cursor phone =  getContentResolver().query
          (ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
           null, ContactsContract.CommonDataKinds.Phone.CONTACT_ID + "=" + contactId,
                                null, null);
          if(phone.moveToFirst()) {
                  Number = phone.getString(phone.
                  getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER));
                      }
           flag = 1;
           phone.close();
           break;
               }
          }
          cursor.close();
```

6.最后就是数据库的设计了：

创建MyDbHelper类，继承自SQLiteOpenHelper。里面写七个函数：构造函数，onCreate， onUpgrade，insert，delete，isExist，update。其中isExist函数返回值是boolean类型，判断是否存在某个姓名的记录。其他的函数就比较常规，主要是调用数据库的一些操作。数据库的操作，都要先getWritableDatabase()一下。

isExist函数：

```
public boolean isExist(String name){
        SQLiteDatabase db = getWritableDatabase();
        boolean result = false;
        Cursor cursor = db.query(TABLE_NAME, null, null, null, null,null,null,null);
        for (cursor.moveToFirst();!(cursor.isAfterLast());cursor.moveToNext()) {
            if(name.equals(cursor.getString(1)))
                result = true;
        }
        cursor.close();
        db.close();
        return result;
}
```

数据库的使用，首先是在应用启动的时候，遍历整个数据库，将数据读取出来，并显示在主页面。

```
myDbHelper = new MyDbHelper(this, "myDataBase", null, 1);
SQLiteDatabase db = myDbHelper.getReadableDatabase();
Cursor cursor = db.query(TABLE_NAME, null, null, null, null,null,null,null);
for (cursor.moveToFirst();!(cursor.isAfterLast());cursor.moveToNext()) {
            Map<String, String> temp = new LinkedHashMap<String, String>();
            temp.put("name", cursor.getString(1));
            temp.put("birth", cursor.getString(2));
            temp.put("gift", cursor.getString(3));
            data.add(temp);
}
cursor.close();
db.close()
```

然后，在新增页面中，需要检查输入的名字是否已经存在数据库中。

```
MyDbHelper myDbHelper =  myDbHelper = new MyDbHelper(getApplicationContext(), "myDataBase",
null, 1);;
       if(myDbHelper.isExist(name_str)){
              Toast.makeText(AddItem.this, "名字重复啦，请检查",   Toast.LENGTH_SHORT).show();
              }
```

收到新增页面的数据后，主页面必须将它显示出来，并保存在数据库中。

```
data.add(temp);
simpleAdapter.notifyDataSetChanged();
myDbHelper.insert(name_str, birth_str, gift_str);
```

弹出对话框并进行修改后，需要更新数据库。

```
String new_birth = ((EditText) layout.findViewById(R.id.birth_input)).getText().toString();
String new_gift = ((EditText) layout.findViewById(R.id.gift_input)).getText().toString();
data.get(position).put("birth", new_birth);
data.get(position).put("gift", new_gift);
simpleAdapter.notifyDataSetChanged();
myDbHelper.update(data.get(position).get("name"),new_birth, new_gift);
```

长按条目删除后，需要将记录也从数据库删除。

```
dialogbuilder.setPositiveButton("是",
         new DialogInterface.OnClickListener() {
                  @Override
                  public void onClick(DialogInterface dialogInterface, int i) {
                              myDbHelper.delete(data.get(arg2).get("name"));
                              data.remove(arg2);
                              simpleAdapter.notifyDataSetChanged();
                  }
 });
```

实验遇到困难以及解决思路

- 设计界面时，TextView和EditText对齐后，感觉会有些错位，需要微调看起来才比较美观。
- 设计对话框时，最开始不知道怎样将数据传到对话框以及得到对话框中的数据。上网查了很多教程，多半是要写一个继承自对话框的类的，非常的麻烦，因此走了许多弯路。后来，思索了整个加载对话框的过程，并不断尝试，才知道可以使用inflate得到的布局来找到或者设置对应的控件。

  ```
  ((TextView) layout.findViewById(R.id.name_input)).setText(data.get(position).get("name"));
  String new_birth = ((EditText) layout.findViewById(R.id.birth_input)).getText().toString();
  ```

  用起来真的非常的舒服。
- 数据库设计时，最开始不知道怎样在不同的Activity之间共享一个数据库。因为在主界面声明了一个数据库后，其他界面是不可见的。后来通过查资料，才发现可以将数据库设计成一个封装的类，创建，插入，删除等等都在类中实现，对同一个数据库进行操作。这样，不管是哪一个页面，用到数据库的时候，只需要声明它的一个对象，然后调用方法就行了。

实验思考及感想

这周实验的主要内容是SQLitebase以及ContentProvideer的使用。SQLitebase是一个轻量级的、非常好用的数据库。它并非是Android所独有，在之前的UWP开发中也使用过这个数据库。

本次实验难度相比前几次要大一些，花了大概一天的时间。但是觉得收获颇多，也享受开发完成后的那种成就感。