# 聊天界面

目录

[聊天界面 1](#_Toc482188195)

[一、 项目总概况 1](#_Toc482188196)

[二、 项目具体实现和效果预览 3](#_Toc482188197)

[1、 登录界面 3](#_Toc482188198)

[2、 注册界面 11](#_Toc482188199)

[3、 主界面 15](#_Toc482188200)

[4、 聊天界面 34](#_Toc482188201)

[三、 数据库表 44](#_Toc482188202)

[四、 心得体会 46](#_Toc482188203)

## 项目总概况

#### 项目主要内容

登录界面，实现了从数据库获取信息进行身份认证并登录，顶端自定义了actionBar为了让字体居中。然后输入框和按钮样式用了shape.xml文件，还有一个注册的跳转连接。注册页面，对用户名，密码，确认密码都做了相应的限制，用totast提示错误，然后对数据库做操作，还有返回按钮，用intent跳转。主界面，用listview实现好友列表的显示，实现了右弹出框，右弹出框实现了帐号的注销，用intent跳回到登录界面，还加了退出的按钮。右上角可以添加用户。聊天界面，用listview显示聊天记录，底部是一个发送框。

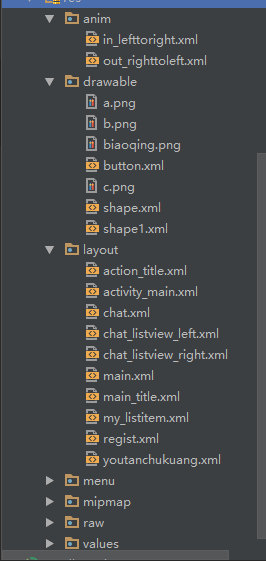
#### 项目各文件说明

············聊天activity

············主界面，好友界面activity

············登录界面activity

············注册activity



················动画弹出效果

···············动画缩回效果

·······················图片文件

·······················button的shape样式文件

·······················shape样式文件

·······················登陆界面actionBar

·······················登录界面布局文件

·······················聊天界面布局文件

·······················聊天listView左边文件

·······················聊天listview右边文件

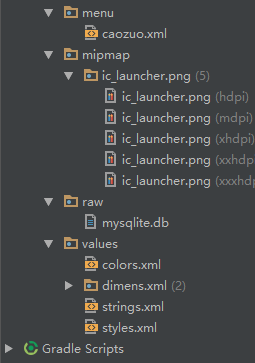
·······················主界面布局文件

·······················主界面actionBar

·······················主界面显示布局文件

·······················注册界面

·······················右弹出框popuwindow文件



··········popupMenu弹出菜单布局文件

·········安卓图标文件夹

········数据库资源文件

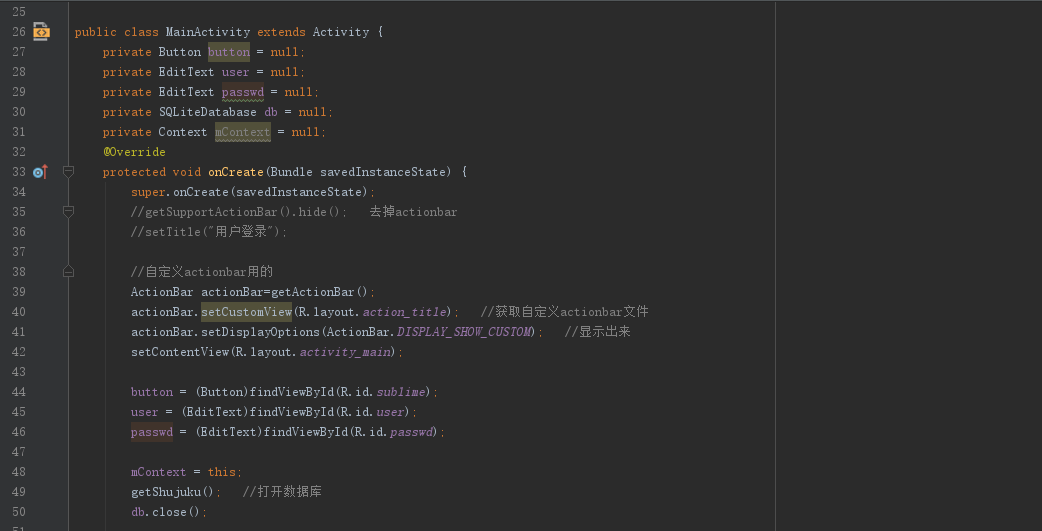
········颜色文件

········字符串文件

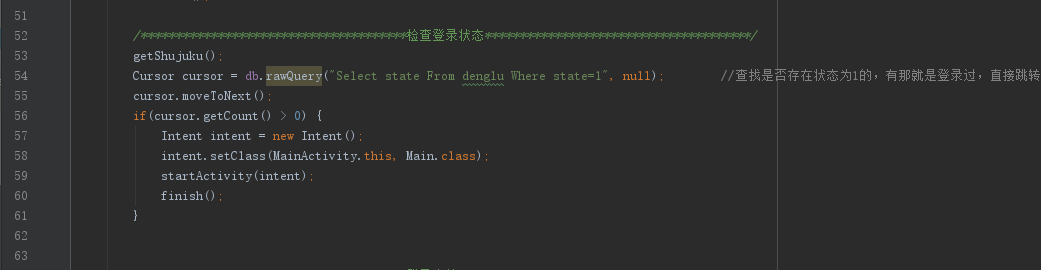
········主题风格文件

## 项目具体实现和效果预览

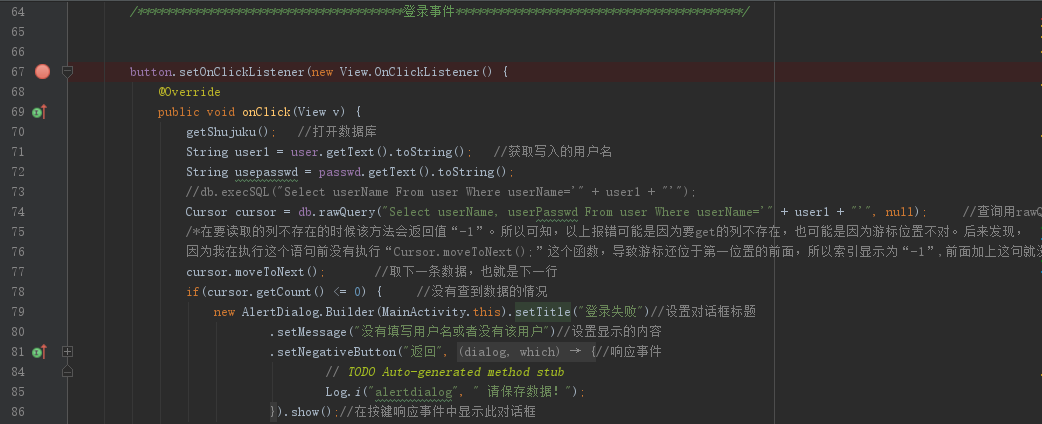
### 登录界面



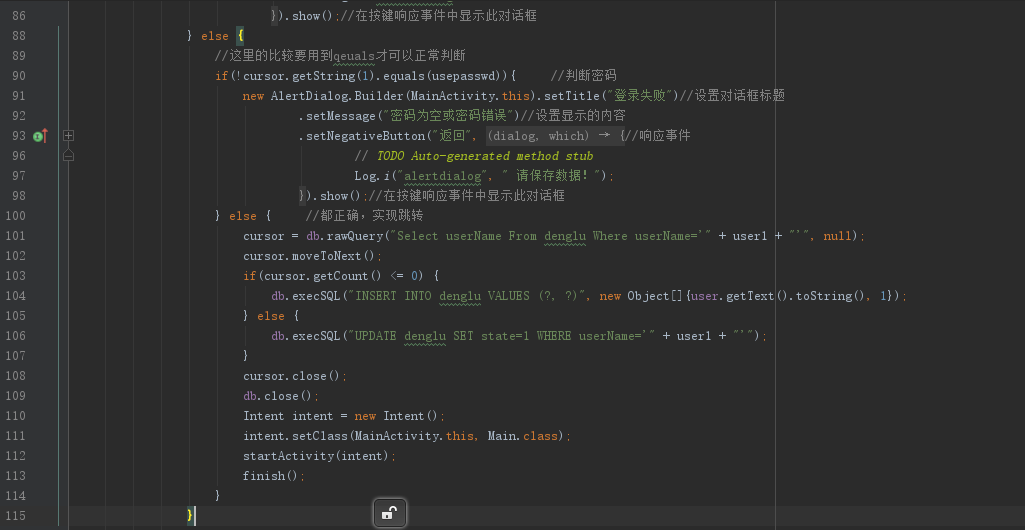
各个按钮的定义，还有自定义actionBar的定义和显示，数据库打开主要是该页面是第一个页面，要是在data/data/项目名字/database/下没有数据库要先把资源文件里面的数据库导进去。



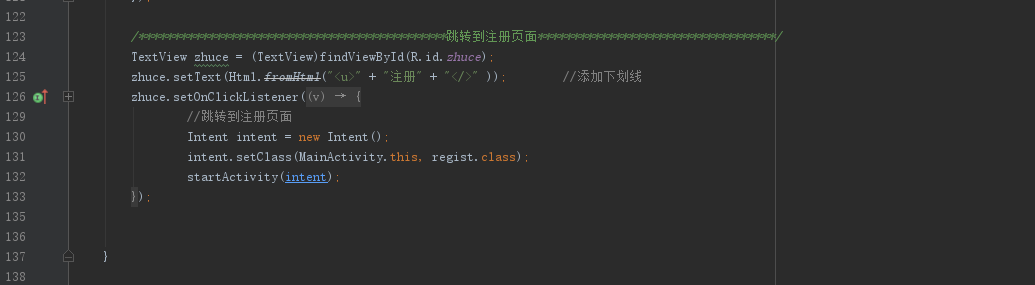
在数据库建了一张状态表，当登陆时，用户的状态值变为1，然后每一次打开应用先判断是否有用户已登录，如果是，直接进入主页面，不用重复登录。



登录事件，监听登录按钮，然后具体实现。先打开数据库，并查询输入用户名，用Cursor存储查询返回的结果，cursor查询的结果第一行都是指向-1的，所以要用moveToNext()线取第一行再判断，判断是否小于等于0，是的话那就是没有该用户。用AlertDialog弹出错误。



不是的话那就是有该用户，那接着对密码进行判断。如果都错误，再弹出AlertDialog弹出框提示错误，都正确，把该用户写入到登录表，并把状态改为1，用Intent实现页面的的跳转，并用finish结束该activity，不然按返回键还是可以回到登录界面的。



监听事件跳转到注册页面



打开数据库文件，先判断是否存在该数据库是的话，直接打开数据库，不在的话，获取资源文件下面的数据库文件，并用文件流写入一个数据库文件到指定的位置。

Activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<!--登陆界面-->  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/activity\_main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 android:background="@color/blue"  
 android:clipToPadding="true"  
 android:fitsSystemWindows="true"  
  
 tools:context="ham.andy.jikezuoye.MainActivity">  
  
 <ImageView  
 android:id="@+id/touxiang"  
 android:layout\_width="70dp"  
 android:layout\_height="70dp"  
 android:layout\_marginTop="10dp"  
 android:layout\_centerHorizontal="true"  
 android:src="@drawable/a"  
 />  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/shape"  
 android:layout\_marginBottom="10dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:id="@+id/zhang"  
 android:layout\_marginTop="100dp">  
 <TextView  
 android:id="@+id/zhanghao"  
 android:layout\_width="40dp"  
 android:layout\_height="wrap\_content"  
 android:text="帐号"  
 android:textSize="15sp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignTop="@+id/user"  
 android:layout\_above="@+id/password" />  
 <View  
 android:id="@+id/view1"  
 android:layout\_width="2dp"  
 android:layout\_height="40dp"  
 android:layout\_centerVertical="true"  
 android:textSize="15sp"  
 android:layout\_gravity="center\_horizontal"  
 android:layout\_marginLeft="1dp"  
 android:layout\_marginRight="2dp"  
 android:layout\_toRightOf="@+id/textView1"  
 android:background="#EEEFFF" />  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:inputType="textPersonName"  
 android:textSize="15sp"  
 android:hint="请输入帐号"  
 android:background="@drawable/shape"  
 android:id="@+id/user"  
 android:layout\_below="@+id/linearLayout"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 </LinearLayout>  
  
 <LinearLayout  
 android:id="@+id/zhang1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/shape"  
 android:layout\_marginBottom="10dp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_below="@id/zhang">  
 <TextView  
 android:id="@+id/text2"  
 android:layout\_width="40dp"  
 android:layout\_height="wrap\_content"  
 android:text="密码"  
 android:textSize="15sp"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignTop="@+id/text1"  
 android:layout\_above="@+id/password" />  
 <View  
 android:id="@+id/view2"  
 android:layout\_width="2dp"  
 android:layout\_height="40dp"  
 android:layout\_centerVertical="true"  
 android:textSize="15sp"  
 android:layout\_gravity="center\_horizontal"  
 android:layout\_marginLeft="1dp"  
 android:layout\_marginRight="2dp"  
 android:layout\_toRightOf="@+id/textView1"  
 android:background="#EEEFFF" />  
  
 <EditText  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:inputType="textPassword"  
 android:textSize="15sp"  
 android:hint="请输入密码"  
 android:background="@drawable/shape"  
 android:id="@+id/passwd"  
 android:layout\_below="@+id/linearLayout"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
</LinearLayout>  
  
 <Button  
 android:text="登录"  
 android:layout\_width="200dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@id/zhang1"  
 android:id="@+id/sublime"  
 android:layout\_centerHorizontal="true"  
 android:background="@drawable/button"  
 />  
  
 <TextView  
 android:text="注册"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="50dp"  
 android:layout\_marginTop="14dp"  
 android:id="@+id/zhuce"  
 android:textSize="20dp"  
 android:gravity="center"  
 android:textColor="#262aff"  
 android:layout\_below="@+id/sublime"  
 android:layout\_toRightOf="@+id/sublime"  
 android:layout\_alignParentRight="true"  
 android:layout\_alignParentEnd="true" />  
  
</RelativeLayout>

主页面的布局文件。

Main\_title.xml

<?xml version="1.0" encoding="utf-8"?>  
<!--登录界面的title-->  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerInParent="true"  
 android:text="用户登录"  
 android:textColor="#FFF"  
 android:textSize="22sp" />  
</RelativeLayout>  
  
<!--  
自定义actionbar的xml文件-->

actionBar布局文件

shape.xml

<?xml version="1.0" encoding="utf-8"?>  
<shape xmlns:android="http://schemas.android.com/apk/res/android"  
 android:shape="rectangle" >  
  
 <solid android:color="#ffffff" />  
 <!-- 内填充颜色-->  
 <stroke  
 android:width="1dip"  
 android:color="#ffffff" />  
 <!-- 边框 -->  
 <padding  
 android:bottom="10dp"  
 android:left="10dp"  
 android:right="10dp"  
 android:top="10dp" />  
 <!-- 圆角 -->  
 <corners android:radius="6dp" />  
  
</shape>  
  
 <!--这是创建自定义的shape格式， 放在drawable，作为其他空间的背景来使用，android:background来使用-->

Shape样式文件

Style.xml

<!-- 自定义首页的actionbar -->  
<style name="myAppTheme" parent="@android:style/Theme.Holo.Light">  
 <item name="android:actionBarStyle">@style/myActionBar</item> <!--设置ActionBarStyle-->  
</style>  
<style name="myActionBar" parent="@android:style/Widget.Holo.Light.ActionBar">  
 <item name="android:height">50dp</item>  
 <item name="android:background">@color/colorPrimary</item>  
 <item name="android:backgroundStacked">@color/black</item>  
</style>  
  
<style name="myAppTheme1" parent="@android:style/Theme.Holo.Light">  
 <item name="android:actionBarStyle">@style/myActionBar\_1</item> <!--设置ActionBarStyle-->  
</style>  
<style name="myActionBar\_1" parent="@android:style/Widget.Holo.Light.ActionBar">  
 <item name="android:height">50dp</item>  
 <item name="android:background">@color/blue2</item>  
 <item name="android:backgroundStacked">@color/black</item>  
</style>



### 注册界面

1. public class regist extends Activity {  
     
    private SQLiteDatabase db = null;  
    private Context mContext = null;  
     
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.*regist*);  
    getActionBar().hide(); //不显示actionBar  
    mContext = this;  
     
    */\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*返回登录界面\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/* ImageView fanhui\_img = (ImageView)findViewById(R.id.*regist\_img*);  
    fanhui\_img.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
    fanghui();  
    }  
    });  
    TextView fanhui\_text = (TextView)findViewById(R.id.*regist\_fanhui*);  
    fanhui\_text.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
    fanghui();  
    }  
    });

这里不再使用actionBar，因为加了之后很难加入监听事件，所以直接写样式使样式和actionBar一样，然后这里是返回操作，监听按钮，实现跳转。Fanghui()函数里面是一个intent的跳转

public void fanghui() {  
 Intent logoutIntent = new Intent(regist.this, MainActivity.class);  
 logoutIntent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 startActivity(logoutIntent);  
 finish();  
}

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*注册事件\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*final EditText user = (EditText)findViewById(R.id.*user*); //获取帐号名  
final EditText passwd = (EditText)findViewById(R.id.*passwd*); //获取密码  
final EditText passwd1 = (EditText)findViewById(R.id.*passwd1*);//获取确认密码  
Button sublime = (Button)findViewById(R.id.*sublime*); //获取确认按钮  
  
sublime.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Toast toast = null;  
 getShujuku();  
 final Cursor cursor = db.rawQuery("Select userName From user Where userName='" + user.getText() + "'", null);  
 if(cursor.getCount() > 0) {  
 toast = Toast.*makeText*(getApplicationContext(), "账户已存在，请重新填写", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 } else {  
 if(passwd.getText().toString().equals("")) {  
 toast = Toast.*makeText*(getApplicationContext(), "密码为空", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 passwd.requestFocus(); //获取焦点  
 cursor.close();  
 db.close();  
 } else if(!passwd1.getText().toString().equals(passwd.getText().toString())) {  
 toast = Toast.*makeText*(getApplicationContext(), "输入的两次密码不一样", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 passwd.requestFocus(); //获取焦点  
 cursor.close();  
 db.close();  
 } else {  
 db.execSQL("INSERT INTO user VALUES (?, ?)", new Object[]{user.getText().toString(), passwd.getText().toString()});  
 cursor.close();  
 db.close();  
 final TextView inputServer = new TextView(mContext);  
 inputServer.setText("注册成功！");  
 inputServer.setPadding(10, 50, 0, 0);  
 inputServer.setGravity(Gravity.*CENTER*);  
 inputServer.setTextSize(20);  
 AlertDialog.Builder builder = new AlertDialog.Builder(mContext);  
 builder.setView(inputServer);  
 builder.setPositiveButton("确定", new DialogInterface.OnClickListener() { //点确定的相应事件  
 public void onClick(DialogInterface dialog, int which) {  
 fanghui();  
 }  
 });  
 builder.show();  
 }  
 }  
 }  
});

这里主要是判断多，对确定按钮做了多个判断，首先是用户名判断，不能是已存在的用户，不用为空，接着是对密码的判断，输入的两次都要一样。

### 主界面

1. public class Main extends Activity {  
    private PopupWindow popupWindow; //从右出来的弹出框  
    private Context mContext = null; //上下文  
    private List<Map<String, Object>> myData; //listitem用  
    //List<Map<String, Object>> items = new ArrayList<Map<String, Object>>(); 是定义一个List类型的变量，list里面存放的是一个Map，而Map的key是一个String类型，Map的value是Object类型,Map是一个接口 代表一个key-value 键值对  
    private List<Map<String, Object>> list = new ArrayList<Map<String, Object>>(); //用全局来保存，实现增加和删除  
    private Map<String, Object> map = new HashMap<String, Object>(); //基于哈希表的map，用来存取数据，  
    private SQLiteDatabase db = null;  
    private MyAdapter adapter = null;  
    String now\_use = null;  
     
     
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    //getSupportActionBar().hide(); 去掉actionbar  
    //setTitle("用户登录");  
    //自定义actionbar用的  
    ActionBar actionBar=getActionBar();  
    actionBar.setCustomView(R.layout.*main\_title*); //获取自定义actionbar文件  
    actionBar.setDisplayOptions(ActionBar.*DISPLAY\_SHOW\_CUSTOM*); //显示出来  
    setContentView(R.layout.*main*);  
     
    mContext = this; //获取主Main的this，有几个地方会用到  
    adapter = new MyAdapter(this);  
     
    getShujuku();  
    Cursor cursor = db.rawQuery("Select userName From denglu Where state=1", null); //查找是否存在状态为1的

cursor.moveToNext();  
 now\_use = cursor.getString(0).toString();  
 cursor.close();  
 db.close();

首先的一些变量的定义，actionBar的显示，然后取当前的的登录用户名给now\_use变量。

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*创建好有列表表\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*getShujuku();  
cursor = db.rawQuery("SELECT count(\*) FROM sqlite\_master WHERE type='table' AND name='" + now\_use + "'", null);  
//判断数据库的存在，这里一定是有数据返回的，存在的话是1，不存在的话是0，所以要用geiInt(0)获取这个值来判断  
cursor.moveToNext();  
if(cursor.getInt(0) == 0) {  
 String sql = "CREATE TABLE " + now\_use + " (\"id\" INTEGER PRIMARY KEY AUTOINCREMENT, \"userName\" TEXT(20), CONSTRAINT \"userName\" FOREIGN KEY (\"userName\") REFERENCES \"user\" (\"userName\"))";  
 db.execSQL(sql);  
 cursor.close();  
 db.close();  
}

每个用户是对应一张好有列表表的，所以在一开始的时候就要创建好友表。

*/\*\*\*\*\*\*\*\*点击头像右边弹出框的实现\*\*\*\*\*\*\*\*\*\*\*/*ImageView touxiang = (ImageView)findViewById(R.id.*touxiang*);  
touxiang.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 initPopupWindow(); //初始化类  
 }  
});

具体实现如下代码

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*右边弹出框相关的函数开始\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*右边弹出框相关的函数开始\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
/\*\*  
 \* 添加新笔记时弹出的popWin关闭的事件，主要是为了将背景透明度改回来  
 \*  
 \*/*class popupDismissListener implements PopupWindow.OnDismissListener{  
  
 @Override  
 public void onDismiss() {  
 backgroundAlpha(1f);  
 }  
  
}  
//右弹出框的初始化代码。  
protected void initPopupWindow(){  
 View popupWindowView = getLayoutInflater().inflate(R.layout.*youtanchukuang*, null); //LayoutInflater是用来找res/layout/下的xml布局文件，并且实例化,因此如果你的Activity里如果用到别的layout,比如对话框上的layout,你还要设置对话框上的layout里的组件(像图片ImageView,文字TextView)上的内容,你就必须用inflate()先将对话框上的layout找出来,然后再用这个layout对象去找到它上面的组件  
 //内容，高度，宽度  
 //4个参数分别为要显示的view，width和height为宽和高，值为像素值，是否获取焦点。View的职责，根据测量模式和ViewGroup给出的建议的宽和高，计算出自己的宽和高；同时还有个更重要的职责是：在ViewGroup为其指定的区域内绘制自己的形态。  
 popupWindow = new PopupWindow(popupWindowView, ViewGroup.LayoutParams.*WRAP\_CONTENT*, ViewGroup.LayoutParams.*FILL\_PARENT*, true);  
 //动画效果，在style文件里面  
 popupWindow.setAnimationStyle(R.style.*AnimationLeftFade*);  
 //菜单背景色  
 ColorDrawable dw = new ColorDrawable(0xffffffff);  
 popupWindow.setBackgroundDrawable(dw);  
 //宽度  
 popupWindow.setWidth(650);  
 //高度  
 //popupWindow.setHeight(LayoutParams.FILL\_PARENT);  
 //显示位置  
 popupWindow.showAtLocation(getLayoutInflater().inflate(R.layout.*activity\_main*, null), Gravity.*LEFT*, 0, 500); //设置弹出框，gravity是弹出的方向  
 //设置背景半透明  
 backgroundAlpha(0.5f);  
  
  
 //获取数据更改用户名  
 final TextView yonghuming = (TextView)popupWindowView.findViewById(R.id.*yonghuming*);  
 yonghuming.setText(now\_use);  
 db.close();  
 popupWindowView.setFocusableInTouchMode(true); //获取淡出框的焦点  
 /\*注销按钮，右弹出框的监听时间，如果只是一般的用inflate来获取的话是不行的，inflate的作用就是获取到xml，  
 这里是获取别的xml文件，取出对应的id，但是inflate就但存是获取对应的xml，是不能对其操作的，所以用setFocusableInTouchMode方法，  
 先获取到这个弹出框的焦点，然后再做监听事件\*/  
 TextView zhuxiao = (TextView)popupWindowView.findViewById(R.id.*zhanghaozhuxiao*);  
 zhuxiao.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //注销的时候把登录的状态改为0  
 getShujuku();  
 db.execSQL("UPDATE denglu SET state=0 WHERE userName='" + yonghuming.getText().toString() + "'");  
 db.close();  
 Intent logoutIntent = new Intent(Main.this, MainActivity.class);  
 logoutIntent.setFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TASK* | Intent.*FLAG\_ACTIVITY\_NEW\_TASK*);  
 startActivity(logoutIntent);  
 finish();  
 }  
 });  
 TextView tuichu = (TextView)popupWindowView.findViewById(R.id.*tuichu*);  
 tuichu.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 System.*exit*(0);  
 }  
 });  
  
 //关闭事件  
 popupWindow.setOnDismissListener(new Main.popupDismissListener()); //将透明度改回来  
 popupWindowView.setOnTouchListener(new View.OnTouchListener() {  
 @Override  
 public boolean onTouch(View v, MotionEvent event) {  
 /\*if( popupWindow!=null && popupWindow.isShowing()){  
 popupWindow.dismiss();  
 popupWindow=null;  
 }\*/  
 // 这里如果返回true的话，touch事件将被拦截  
 // 拦截后 PopupWindow的onTouchEvent不被调用，这样点击外部区域无法dismiss  
 return false;  
 }  
 });  
  
}  
*/\*\*  
 \* 设置添加屏幕的背景透明度  
 \** ***@param*** *bgAlpha  
 \*/*public void backgroundAlpha(float bgAlpha)  
{  
 WindowManager.LayoutParams lp = getWindow().getAttributes(); //得到透明属性值  
 lp.alpha = bgAlpha; //0.0-1.0  
 getWindow().setAttributes(lp); //setAttributes设置透明属性  
}  
*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*右边弹出框相关的函数结束\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*右边弹出框相关的函数结束\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

主要是popupwindow的使用，唯一的注意点是popupWindowView.setFocusableInTouchMode获取焦点的函数要加上，不然不能捕抓到控件，注销是把当前登录的用户状态表改为0，然手用intent来跳转，退出是用System。Exit(0)，关闭弹出框是用return false;

*/\*\*\*\*\*\*\*\*点击右边的十字架出现操作按钮\*\*\*\*\*\*\*/*final ImageView caozuo = (ImageView)findViewById(R.id.*caozuo*);  
caozuo.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 showPopupMenu(caozuo);  
 }  
});

private void showPopupMenu(View view) {  
 // View当前PopupMenu显示的相对View的位置  
 PopupMenu popupMenu = new PopupMenu(this, view);  
 // menu布局  
 popupMenu.getMenuInflater().inflate(R.menu.*caozuo*, popupMenu.getMenu());  
 // menu的item点击事件  
 popupMenu.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {  
 @Override  
 public boolean onMenuItemClick(MenuItem item) {  
 //创建一个带输入框的弹出框  
 final EditText inputServer = new EditText(mContext);  
 inputServer.setBackgroundResource(R.drawable.*shape*);  
 AlertDialog.Builder builder = new AlertDialog.Builder(mContext);  
 builder.setTitle("请输入要添加的用户名").setView(inputServer)  
 .setNegativeButton("取消", null); //取消就取消  
 builder.setPositiveButton("确定", new DialogInterface.OnClickListener() { //点确定的相应事件  
 public void onClick(DialogInterface dialog, int which) {  
 String yonghuming = null;  
 yonghuming = inputServer.getText().toString(); //获取输入的数据  
 Toast toast = null;  
 getShujuku();  
 Cursor cursor = db.rawQuery("Select userName From " + now\_use +" Where userName = '" + yonghuming + "'", null);  
 cursor.moveToNext();  
 if(cursor.getCount() <= 0) {  
 cursor = db.rawQuery("Select userName From user Where userName = '" + yonghuming + "'", null);  
 cursor.moveToNext();  
 if(cursor.getCount() <= 0) {  
 toast = Toast.*makeText*(getApplicationContext(), "没有该用户", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 cursor.close();  
 db.close();  
 } else {  
 if(cursor.getString(0).equals(now\_use)) {  
 toast = Toast.*makeText*(getApplicationContext(), "不能添加自己", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 cursor.close();  
 db.close();  
 } else {  
 db.execSQL("Insert into " + now\_use + " Values(null, '" + yonghuming + "')");  
 db.execSQL("Insert into " + yonghuming + " Values(null, '" + now\_use + "')");  
 map = new HashMap<String, Object>();  
 map.put("title", yonghuming);  
 map.put("img", R.drawable.*a*);  
 list.add(map);  
 adapter.notifyDataSetChanged();  
 toast = Toast.*makeText*(getApplicationContext(), "好友添加成功", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 cursor.close();  
 db.close();  
 }  
 }  
  
 } else {  
 toast = Toast.*makeText*(getApplicationContext(), "该用户已在你的好友列表里面", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 cursor.close();  
 db.close();  
 }  
  
 }  
 });  
 builder.show();  
 return false;  
 }  
 });  
 popupMenu.show();  
}

添加按钮，用builder.setTitle("请输入要添加的用户名").setView(inputServer).setNegativeButton("取消", null);来实现有输入框的弹出框，然后在添加的时候对数据库做操作，首先，不能重复添加好友，接着该用户要存在的，再者不能添加自己为好友，都满足了后就能添加好友了，添加的时候要对好友表做添加，数据库操作用execSql就能加入语句了。然后要对listView进行更新，加入新的haxmap，然后通知适配器数据更改了notifyDataSetChanged。

*/\*\*\*\*\*\*\*\*点击用户跳转到不同的聊天页面\*\*\*\*\*\*\*/*ListView listview = null; //生成设配器，数组===》listItem  
listview = (ListView)findViewById(R.id.*MyListView*);  
  
/\* SimpleAdapter adapter = new SimpleAdapter(this, getData(), //数据来源  
 R.layout.my\_listitem, //ListItem的xml实现  
 new String[]{"title", "img"}, //动态数组与ListItem对应的子项  
 new int[]{R.id.title, R.id.img}); //ListItem的XML文件里面的id  
listview.setAdapter(adapter);//启动并显示设配器\*/  
//用BaseAdapter并重写BaseAdapter类来实现  
myData = getData();  
MyAdapter adapter = new MyAdapter(this);  
listview.setAdapter(adapter);

好友列表的显示和跳转，用了baseAdapter来实现，具体在重构的适配器Myadapter里面。

private List<Map<String, Object>> getData() {  
 //获取数据用的，一个哈希map  
  
 //查找出该用户的好友表，遍历显示出来  
 getShujuku();  
 Cursor cursor = db.rawQuery("Select userName From " + now\_use, null);  
 while(cursor.moveToNext()) {  
 map = new HashMap<String, Object>();  
 map.put("title", cursor.getString(0));  
 map.put("img", R.drawable.*a*);  
 list.add(map);  
 }  
 cursor.close();  
 db.close();  
 return list;  
}

先是取出好友列表的所有好友，然后通过循环cursor所有行，把所有数据添加到list的哈希表中。

static class ViewHolder {  
 public ImageView img;  
 public TextView title;  
 public Button button;  
 } //创建一个静态类，后面要用

定义一个对象类

public class MyAdapter extends BaseAdapter {  
 private LayoutInflater yonghu;  
  
 public MyAdapter(Context context) {  
 this.yonghu = LayoutInflater.*from*(context);  
 }  
 @Override  
 public int getCount() {  
 // *TODO Auto-generated method stub* //在此适配器中所代表的数据集中的条目数  
 return myData.size();  
 }  
  
 @Override  
 public Object getItem(int arg0) {  
 // *TODO Auto-generated method stub* //获取数据集中与指定索引对应的数据项  
 return null;  
 }  
  
 @Override  
 public long getItemId(int arg0) {  
 // *TODO Auto-generated method stub* //取在列表中与指定索引对应的行id  
 return 0;  
 }  
  
 @Override  
 public View getView(final int position, View convertView, ViewGroup parent) {  
 //获取一个在数据集中指定索引的视图来显示数据  
 ViewHolder holder = null;  
 //如果缓存converView为空，则需创建View  
 if(convertView == null) {  
 holder = new ViewHolder();  
 convertView = yonghu.inflate(R.layout.*my\_listitem*, null);  
 holder.img = (ImageView)convertView.findViewById(R.id.*img*);  
 holder.title = (TextView)convertView.findViewById(R.id.*title*);  
 holder.button = (Button)convertView.findViewById(R.id.*view\_btn*);  
 //将设置好的布局保存到缓存中，并将其设置在Tag里，以便取出Tag  
 convertView.setTag(holder);  
 } else {  
 //缓存有，直接取出  
 holder = (ViewHolder)convertView.getTag();  
 }  
 //设置创建的img数据为myData的第一个数据  
 holder.img.setBackgroundResource((Integer)myData.get(position).get("img"));  
 holder.title.setText((String)myData.get(position).get("title"));  
 final String aaa = (String)myData.get(position).get("title"); //获取当前的title，也就是该好友  
 holder.button.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //删除对应的view，list.remove(position)是删除哈希表的数据，notifyDataSetChanged()是告知数据被改变需要更新。  
 new AlertDialog.Builder(Main.this).setTitle("删除好友")//设置对话框标题  
 .setMessage("你确定要删除吗？")//设置显示的内容  
 .setNegativeButton("返回", new DialogInterface.OnClickListener() {//添加返回按钮  
 @Override  
 public void onClick(DialogInterface dialog, int which) {//响应事件  
 // *TODO Auto-generated method stub* Log.*i*("alertdialog", " 请保存数据！");  
 }  
 }).setPositiveButton("确定",new DialogInterface.OnClickListener() { //确定  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
 //删除数据  
 getShujuku();  
 db.execSQL("Delete From " + now\_use + " Where userName='" + aaa + "'");  
 db.execSQL("Delete From " + aaa + " Where userName='" + now\_use + "'");  
 db.close();  
 Toast toast = null;  
 toast = Toast.*makeText*(getApplicationContext(), "删除陈功", Toast.*LENGTH\_LONG*);  
 toast.setGravity(Gravity.*CENTER*, 0, -200);  
 toast.show();  
 list.remove(position);  
 notifyDataSetChanged();  
 }  
 }).show();//在按键响应事件中显示此对话框  
 }  
 });  
 //监视每一个listview，点击的时候跳转到对应的activity  
 holder.img.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent();  
 //传递参数  
 intent.putExtra("title", aaa);  
 intent.setClass(Main.this, chat.class);  
 startActivity(intent);  
 }  
 });  
 //监视每一个listview，点击的时候跳转到对应的activity  
 holder.title.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent();  
 //传递参数  
 intent.putExtra("title", aaa);  
 intent.setClass(Main.this, chat.class);  
 startActivity(intent);  
 }  
 });  
 return convertView;  
 }

设配器的编写，主要在

public View getView(final int position, View convertView, ViewGroup parent)

这个方法里面，用if判断缓存，增加效率，取出相对应的所有控件名字，然后把哈希表里面的数据给控件，布局文件单独写出来。还有两个事件，删除的监听事件，主要就是删除也要连表里面的一起删除，而且是相互的。然后就是跳转，因为跳转是不同的人的，所以这里增加了跳转的参数

intent.putExtra("title", aaa);

把aaa传递给了聊天界面。

Main。Xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:background="#fff"  
 android:id="@+id/linear">  
  
 <ListView android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/MyListView">  
 </ListView>  
  
  
</LinearLayout>

Main\_title.xml

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <ImageView  
 android:id="@+id/touxiang"  
 android:layout\_width="30dp"  
 android:layout\_height="30dp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="10dp"  
 android:src="@drawable/a"  
 />  
  
 <ImageView  
 android:id="@+id/caozuo"  
 android:layout\_width="30dp"  
 android:layout\_height="30dp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginRight="10dp"  
 android:layout\_alignParentRight="true"  
 android:src="@drawable/b"  
 />  
</RelativeLayout>  
 <!--  
 自定义actionbar的xml文件-->

My\_listview.xml 好友列表每一项的布局

<?xml version="1.0" encoding="utf-8"?>  
<!--listview的自定义样式，用的simpleAdapter-->  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:padding="10dp">  
  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="1"  
 >  
 <ImageView  
 android:id="@+id/img"  
 android:layout\_width="50dp"  
 android:layout\_height="50dp"  
 android:layout\_margin="5px"  
 android:background="@drawable/a"  
 />  
 </LinearLayout>  
  
  
 <TextView android:id="@+id/title"  
 android:layout\_width="230dp"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="1.5"  
 android:paddingLeft="20dp"  
 android:layout\_gravity="center\_vertical"  
 android:gravity="center\_vertical"  
 android:textSize="16dp"  
 />  
 <Button android:id="@+id/view\_btn"  
 android:layout\_width="80dp"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="1.5"  
 android:text="删除"  
 android:layout\_gravity="center\_vertical|right"  
 />  
  
  
</LinearLayout>

Youtanchukuang.xml 右弹出框的布局

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:orientation="vertical" >  
  
 <RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="140dp"  
 android:background="@color/blue2">  
 <ImageView  
 android:id="@+id/touxiang"  
 android:layout\_width="60dp"  
 android:layout\_height="60dp"  
 android:layout\_marginTop="30dp"  
 android:layout\_marginLeft="20dp"  
 android:src="@drawable/a"  
 />  
  
 <TextView  
 android:id="@+id/yonghuming"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="用户名"  
 android:layout\_below="@+id/touxiang"  
 android:layout\_alignLeft="@+id/touxiang"  
 android:layout\_alignStart="@+id/touxiang"  
 android:layout\_marginTop="15dp"  
 android:textSize="15dp"/>  
 </RelativeLayout>  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:text="设置功能项"  
 android:gravity="center\_vertical"  
 android:textSize="15dp"  
 android:layout\_marginLeft="20dp"/>  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:text="设置功能项"  
 android:gravity="center\_vertical"  
 android:textSize="15dp"  
 android:layout\_marginLeft="20dp"/>  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:text="设置功能项"  
 android:gravity="center\_vertical"  
 android:textSize="15dp"  
 android:layout\_marginLeft="20dp"/>  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:text="设置功能项"  
 android:gravity="center\_vertical"  
 android:textSize="15dp"  
 android:layout\_marginLeft="20dp"/>  
  
 <!-- 分割线-->  
 <View  
 android:layout\_width="fill\_parent"  
 android:layout\_height="1px"  
 android:layout\_marginTop="15dp"  
 android:layout\_marginBottom="15dp"  
 android:background="@android:color/darker\_gray" />  
 <TextView  
 android:id="@+id/zhanghaozhuxiao"  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:text="帐号注销"  
 android:gravity="center\_vertical"  
 android:textSize="15dp"  
 android:layout\_marginLeft="20dp"/>  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:text="设置"  
 android:gravity="center\_vertical"  
 android:textSize="15dp"  
 android:layout\_marginLeft="20dp"/>  
 <TextView  
 android:id="@+id/tuichu"  
 android:layout\_width="match\_parent"  
 android:layout\_height="40dp"  
 android:text="退出"  
 android:gravity="center\_vertical"  
 android:textSize="15dp"  
 android:layout\_marginLeft="20dp"/>  
  
  
  
</LinearLayout>

in\_lefttoright.xml 动画文件

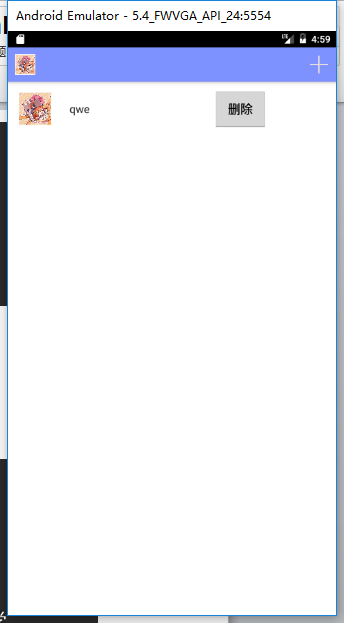
<?xml version="1.0" encoding="utf-8"?>  
<set xmlns:android="http://schemas.android.com/apk/res/android">  
  
 <translate  
 android:fromXDelta="-100%"  
 android:toXDelta="0"  
 android:duration="200"/>  
  
</set>

Out\_leftoright.xml 动画文件

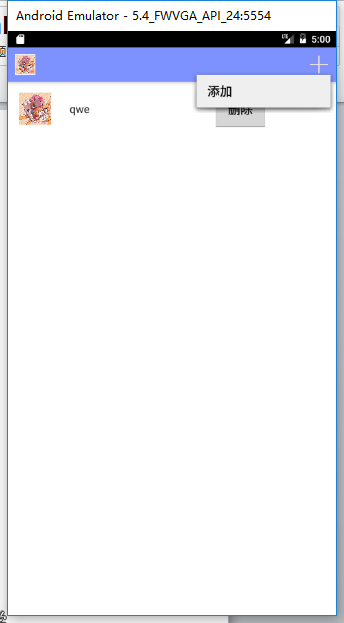
<?xml version="1.0" encoding="utf-8"?>  
<set xmlns:android="http://schemas.android.com/apk/res/android">  
  
 <translate android:fromXDelta="0"  
 android:toXDelta="-100%"  
 android:duration="200"/>  
  
</set>

Caozuo.xml 添加按钮布局

<?xml version="1.0" encoding="utf-8"?>  
<menu xmlns:android="http://schemas.android.com/apk/res/android" >  
  
 <item  
 android:id="@+id/open"  
 android:title="添加"/>  
  
</menu>





### 聊天界面

1. public class chat extends Activity{  
     
    private Context mContext = null; //上下文  
    String now\_use = null;  
    private SQLiteDatabase db = null;  
    private String aaa = null; //保存和谁聊天  
     
    ArrayList<HashMap<String, Object>> chatList = null;  
    private ListView chatListView = null; //获取listview值  
    private MyChatAdapter adapter = null; //适配器  
    private String a = null; //输入框的值，发送用的  
     
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.*chat*);  
    getActionBar().hide(); //不显示actionBar  
     
    mContext = this;

常规的定义

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*获取当前的登录用户名\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*getShujuku();  
Cursor cursor = db.rawQuery("Select userName From denglu Where state=1", null); //查找是否存在状态为1的，有那就是登录过，直接跳转，没有那就继续下面的操作  
cursor.moveToNext();  
now\_use = cursor.getString(0).toString();  
cursor.close();  
db.close();

获取当前登录的人是谁，并给了now\_use

*/\*\*\*\*\*\*\*\*\*\*更改标题\*\*\*\*\*\*\*\*\*/*Intent intent = getIntent(); //获取传过来的数据  
aaa = intent.getStringExtra("title");  
TextView chat\_title = (TextView)findViewById(R.id.*chat\_title*);//根据传过来的数据更改标题  
chat\_title.setText(aaa);

更改标题，用intent.getStringExtra获取从主页面得来的好友名，并赋给aaa变量。设置标题也就是当前和谁聊天。

*/\*\*\*\*\*\*\*\*\*\*放回按钮\*\*\*\*\*\*\*\*\*/*TextView fanghui = (TextView)findViewById(R.id.*chat\_fanhui*);  
fanghui.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 finish();  
 }  
});  
ImageView fanghui1 = (ImageView)findViewById(R.id.*chat\_img*);  
fanghui1.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 finish();  
 }  
});

返回主页面

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*聊天记录显示\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*chatList = new ArrayList<HashMap<String, Object>>();  
addTextToList(); //执行显示函数

聊天巨鹿的显示，也是用到了设配器主要函数如下

protected void addTextToList(){  
 getShujuku();  
 String sql = "Select message, sendName, receiveName From message where sendName='" + now\_use + "' and receiveName='" + aaa + "' OR sendName='" + aaa + "' and receiveName='" + now\_use + "'";  
 Cursor cursor = db.rawQuery(sql, null);  
 while(cursor.moveToNext()) {  
 if(cursor.getString(1).equals(now\_use)) {  
 HashMap<String,Object> map=new HashMap<String,Object>();  
 map.put("person", 0);  
 map.put("text", cursor.getString(0));  
 chatList.add(map);  
 } else {  
 HashMap<String,Object> map=new HashMap<String,Object>();  
 map.put("person", 1);  
 map.put("text", cursor.getString(0));  
 chatList.add(map);  
 }  
 }  
}

信息存储在了信息表，用查询语句把信息全部查询出来，然后循环cursor对象，这里面因为自己发的显示在右边，好友发的显示在左边，所以给了一个person的键，为了区分到底是自己发的还是好友发的，自己发的标记为0，好友发的标记为1，然后吸入到chatList哈希表。

public class MyChatAdapter extends BaseAdapter {  
 Context context = null;  
 ArrayList<HashMap<String, Object>> chatList = null;  
 public MyChatAdapter(Context context, ArrayList<HashMap<String, Object>> chatList) {  
 super();  
 this.chatList = chatList;  
 this.context = context;  
 }  
 @Override  
 public int getCount() {  
 // *TODO Auto-generated method stub* //在此适配器中所代表的数据集中的条目数  
 return chatList.size();  
 }  
  
 @Override  
 public Object getItem(int arg0) {  
 // *TODO Auto-generated method stub* //获取数据集中与指定索引对应的数据项  
 return null;  
 }  
  
 @Override  
 public long getItemId(int position) {  
 // *TODO Auto-generated method stub* //取在列表中与指定索引对应的行id  
 return position;  
 }  
  
 class ViewHolder {  
 public TextView textView = null;  
 }  
  
 @Override  
 public View getView(final int position, View convertView, ViewGroup parent) {  
 //设置是对面发来的还是自己发的  
 /\* 这里不能加缓存，找了一晚上的错误。。。。因为像第二个一样加了缓存的话，不能准确的判断是要用哪一个  
 convertView = LayoutInflater.from(context).inflate(layout[a==0?0:1], null);造成错误 \*/  
 ViewHolder holder = null;  
 //两个页面  
 int[] layout = {R.layout.*chat\_listview\_right*,R.layout.*chat\_listview\_left*};  
 //两个页面对应的不同的text名字  
 int[] to = {R.id.*chatlist\_text\_me*, R.id.*chatlist\_text\_other*};  
 String[] from = {"chat\_list\_right\_text", "chat\_list\_left\_text"};  
 int a = (Integer)chatList.get(position).get("person");  
 //用person来判断，是0的话就是自己发的了  
 convertView = LayoutInflater.*from*(context).inflate(layout[a==0?0:1], null);  
 holder = new ViewHolder();  
 holder.textView = (TextView) convertView.findViewById(to[a]);  
 //将设置好的布局保存到缓存中，并将其设置在Tag里，以便取出Tag  
 convertView.setTag(holder);  
  
 //设置创建的img数据为myData的第一个数据  
 holder.textView.setText((String) chatList.get(position).get("text"));  
 return convertView;  
 }  
}

设配器和主页面差不多，不同的是不能缓存，因为不同的发送者要随时更加要用的样式套

convertView = LayoutInflater.*from*(context).inflate(layout[a==0?0:1], null);

这句就是判断倒是是谁发的，为了应用两套不同的布局文件。

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*发送事件\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/* Button send = (Button)findViewById(R.id.*chat\_send*);  
  
 send.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //获取到输入框的值  
 a = editText.getText().toString();  
 if (a.length() == 0)  
 return;  
 editText.setText("");  
 //添加到哈希表  
  
 getShujuku();  
 db.execSQL("Insert Into message Values(null, '" + a + "', '" + now\_use + "', '" + aaa + "')");  
 db.close();  
 HashMap<String, Object> map = new HashMap<String, Object>();  
 map.put("person", 0);  
 map.put("text", a);  
 chatList.add(map);  
  
 //通知说数据更改了，  
 adapter.notifyDataSetChanged();  
 //定位到到listview中的view中的最后一个  
 chatListView.setSelection(chatList.size() - 1);  
  
 }  
 });  
  
  
 chatListView.setAdapter(adapter);

最后一个发送功能，获取输入的内容，插入消息到消息表，插入数据到哈希表，告知适配器数据更改了，用setselection让屏幕停留在最下面。

Chat.xml

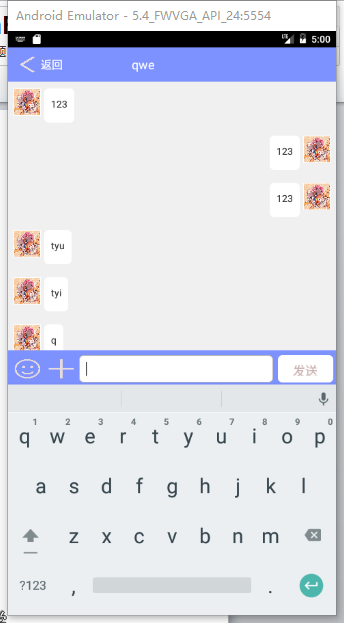
<?xml version="1.0" encoding="utf-8"?>  
<!--聊天窗口主界面-->  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 >  
  
 <!-- 最上面的标题栏-->  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dip"  
 android:id="@+id/title"  
 android:layout\_alignParentTop="true"  
 android:background="@color/blue2">  
 <ImageView  
 android:id="@+id/chat\_img"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="4.1"  
 android:layout\_marginLeft="8dip"  
 android:layout\_gravity="center\_vertical"  
 android:textColor="@android:color/white"  
 android:background="@drawable/c"  
 android:layout\_marginTop="13dp"  
 android:layout\_marginBottom="13dp"  
 />  
 <TextView  
 android:id="@+id/chat\_fanhui"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="4"  
 android:gravity="center\_vertical"  
 android:layout\_gravity="center\_vertical"  
 android:text="返回"  
 android:textColor="@android:color/white"  
 android:textSize="7pt" />  
 <TextView  
 android:id="@+id/chat\_title"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="1"  
 android:textSize="8pt"  
 android:layout\_marginLeft="75dp"  
 android:gravity="center\_vertical"  
 android:textColor="@android:color/white"  
 android:layout\_gravity="center\_vertical"/>  
  
 </LinearLayout>  
 <!--最下面的发送框-->  
 <LinearLayout  
 android:id="@+id/chat\_bottom\_linear"  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dip"  
 android:orientation="horizontal"  
 android:layout\_alignParentBottom="true"  
 android:paddingTop="7dip"  
 android:background="@color/blue2"  
 android:paddingBottom="3dip">  
  
 <ImageButton  
 android:id="@+id/chat\_bottom\_look"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="3.5"  
 android:layout\_marginLeft="7dip"  
 android:layout\_marginTop="4dip"  
 android:layout\_marginBottom="4dp"  
 android:layout\_gravity="center\_vertical"  
 android:background="@drawable/biaoqing"/>  
 <ImageButton  
 android:id="@+id/chat\_bottom\_add"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="3.5"  
 android:layout\_marginLeft="7dip"  
 android:layout\_marginTop="4dip"  
 android:layout\_marginBottom="4dp"  
 android:layout\_gravity="center\_vertical"  
 android:background="@drawable/b"/>  
 <EditText  
 android:id="@+id/chat\_edittext"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginLeft="5dip"  
 android:layout\_marginRight="7dip"  
 android:layout\_weight="1.5"  
 android:background="@drawable/shape1"/>  
 <Button  
 android:id="@+id/chat\_send"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="3.2"  
 android:layout\_marginRight="4dip"  
 android:background="@drawable/shape"  
 android:text="发送"  
 android:textColor="#ccb4b4"/>  
  
  
 </LinearLayout>  
  
 <!--显示数据的listView-->  
 <ListView  
 android:id="@+id/chat\_list"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@id/title"  
 android:layout\_above="@id/chat\_bottom\_linear"  
 android:fadingEdge="none"  
 android:background="#f0f0f0"  
 android:divider="#aaaaaa"  
 android:stackFromBottom="true"  
 android:dividerHeight="0px">  
 </ListView>  
  
  
  
  
</RelativeLayout>

chat\_listview\_left.xml 好友发的布局

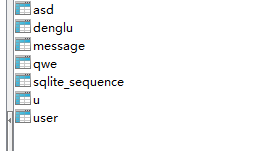
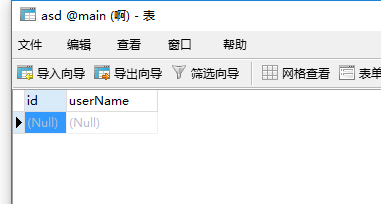
<?xml version="1.0" encoding="utf-8"?>  
<!--对方发出信息的listview布局-->  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="#f0f0f0"  
 android:paddingRight="80dip"  
 android:paddingLeft="8dip"  
 android:paddingBottom="10dip"  
 android:paddingTop="10dip">  
  
 <ImageView  
 android:id="@+id/chatlist\_image\_other"  
 android:layout\_width="40dip"  
 android:layout\_height="40dip"  
 android:layout\_marginRight="5dip"  
 android:layout\_alignParentLeft="true"  
 android:background="@drawable/a"/>  
 <TextView  
 android:id="@+id/chatlist\_text\_other"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_toRightOf="@id/chatlist\_image\_other"  
 android:layout\_alignTop="@id/chatlist\_image\_other"  
 android:text="hewdfb"  
 android:background="@drawable/shape"  
 android:paddingTop="13dip"  
 android:paddingBottom="18dip" />  
  
  
  
  
</RelativeLayout>

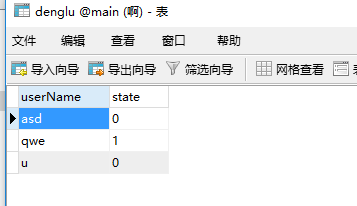
Chat\_listview\_right.xml 自己发的布局

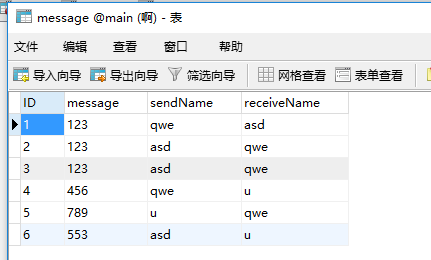
<?xml version="1.0" encoding="utf-8"?>  
<!--我发出信息的listview布局-->  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="#f0f0f0"  
 android:paddingLeft="80dip"  
 android:paddingRight="8dip"  
 android:paddingBottom="10dip"  
 android:paddingTop="10dip">  
  
 <ImageView  
 android:id="@+id/chatlist\_image\_me"  
 android:layout\_width="40dip"  
 android:layout\_height="40dip"  
 android:layout\_marginLeft="5dip"  
 android:layout\_alignParentRight="true"  
 android:background="@drawable/a"/>  
 <TextView  
 android:id="@+id/chatlist\_text\_me"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_toLeftOf="@id/chatlist\_image\_me"  
 android:layout\_alignTop="@id/chatlist\_image\_me"  
 android:text="会忽悠了gyughiulhlluihui过"  
 android:background="@drawable/shape"  
 android:paddingTop="13dip"  
 android:paddingBottom="18dip" />  
  
  
  
  
</RelativeLayout>

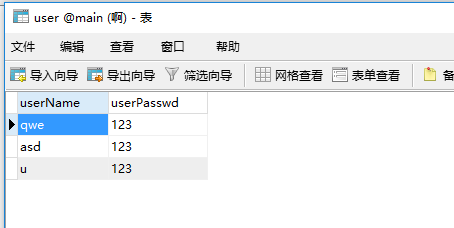


## 数据库表







## 心得体会

这次编写用了一个星期有，因为不是很懂java和安卓的函数，各种不知道功能，找了无数的博客，一个一个按钮组件功能敲出来的，真的费了很多时间，各种奇葩的问题也是层出不穷，但还好基本都是布局或者数据库判断。都能够顺利的解决，也学到了很多很多东西，像布局，能够自定义actionBar，知道了可以自定义主题，知道了shape的布局可以当背景实现输入框的美化，知道了数据库的增删改查的基本方法，知道了文件流的输入输出，知道了totast弹出框，熟练了intent的跳转，知道了popupwindow的弹出框，知道了popupmenu的弹出框，知道了listview的显示布局，知道了适配器adapter等还有好多好多小知识。可以说对安卓基本的布局和数据库操作有所了解了，打下了安卓app开发的基础。

附项目git地址：https://github.com/CLnagisa/My-Android/tree/master/jikezuoye