Brief Introduct

1. 小组分工

yuyao

zsh

zly

zqk

2. 项目需求

1. 用 Java 实现一个后台服务程序，负责消息分发。客户端和服务器连接必须使用长连接（socket），不能使用 HTTP。

2. 手机端实现通讯录管理，在好友上线后能获取通知。

3. 好友之间的文本通讯和短语音通讯（经过服务器转发），要求在本地 Sqlite 数据库中保存通讯记录。

扩展要求：（选做）

1). 聊天记录管理、搜索等

2). 图片等多媒体资源的通讯

3). 更丰富的服务器功能（如是否在线等）

4). 类似微信朋友圈功能

5). 其他功能

3. 项目要点

3.1 用户注册及登陆

将注册的用户名及对应密码放入数据库，每次登录时将用户名和密码与数据库信息互相匹配，匹配成功时登陆。

3.2 消息发送及接受

用 http 的 get 方法请求指定的页面信息并返回实体信息，用第三方平台发送信息。

3.3 发现模块的音乐插件

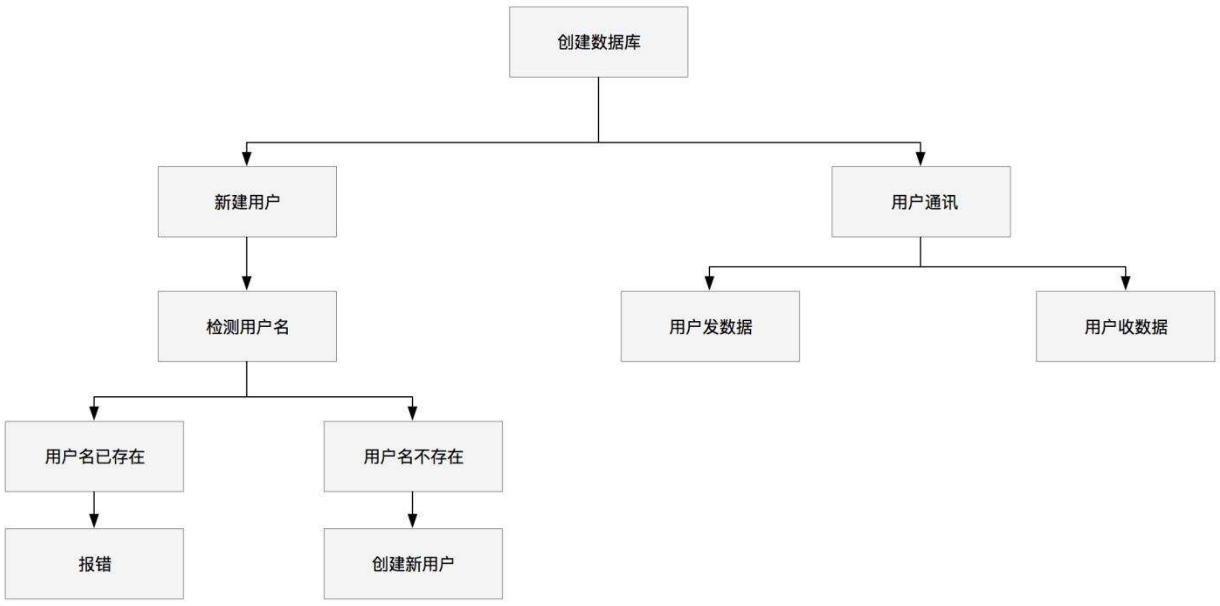
一键打开游戏中的音乐插件，播放喜好音乐

3.4 界面设计

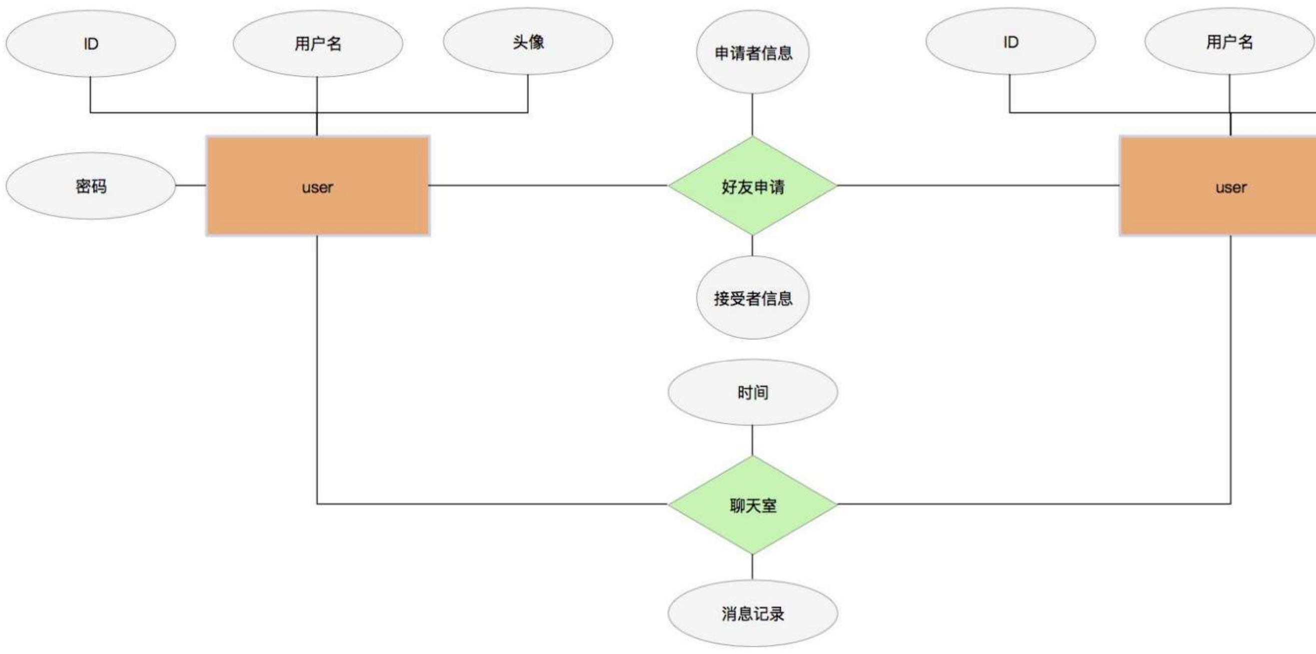
界面参考微信界面设计，减少了喜好表情、附加程序等功能

4. 设计思想

功能框图

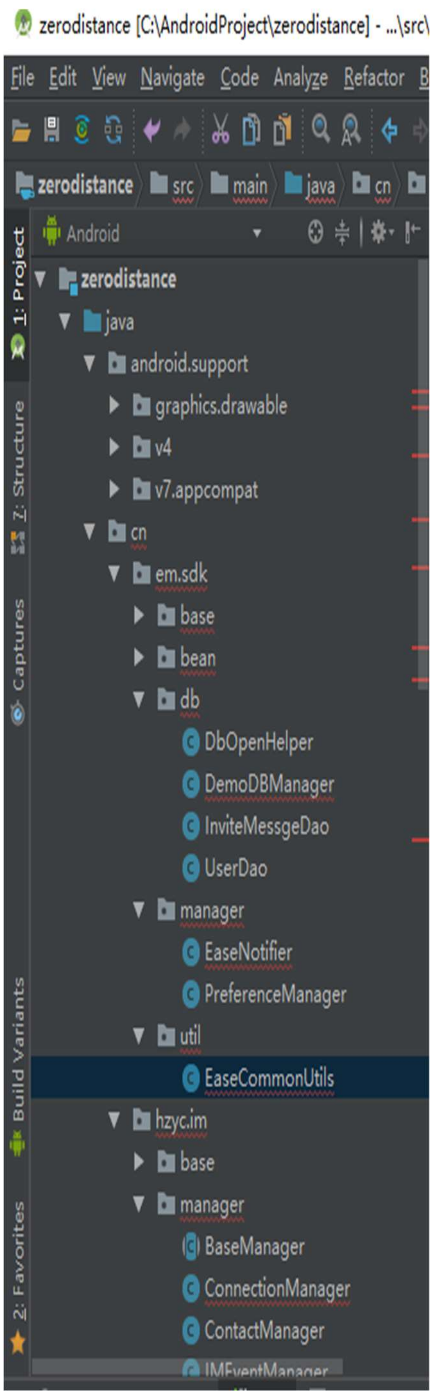

创建数据库
新建用户
用户通讯
检测用户名
用户发数据
用户收数据
用户名已存在
用户名不存在
报错
创建新用户

ER 图：

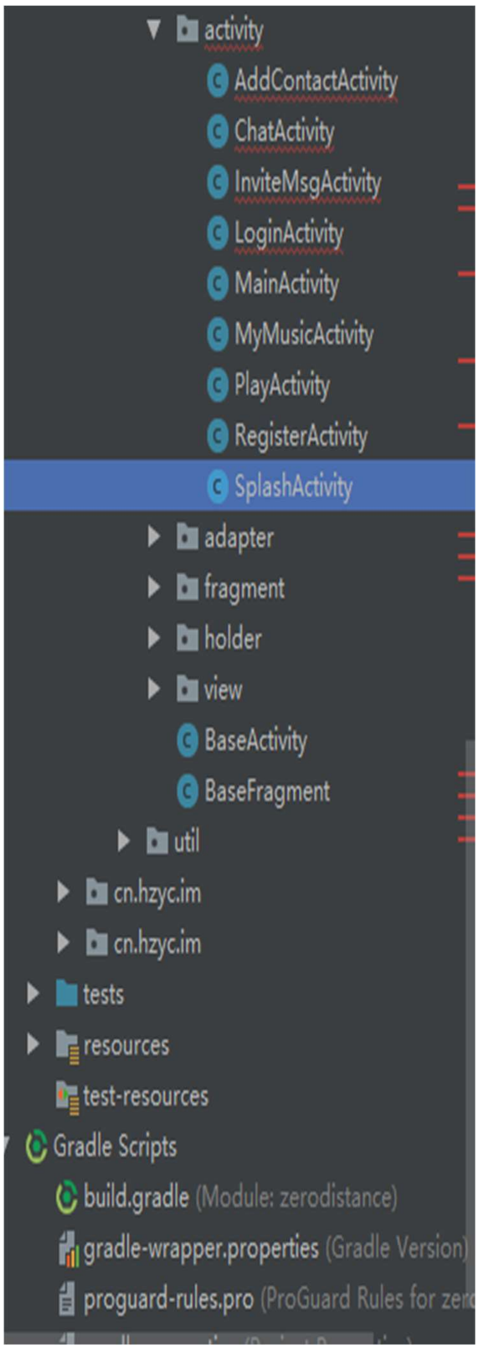

ID
用户名
头像
申请者信息
ID
用户名
密码
user
好友申请
user
接受者信息
时间
聊天室
消息记录

5. 代码示例

文件结构：

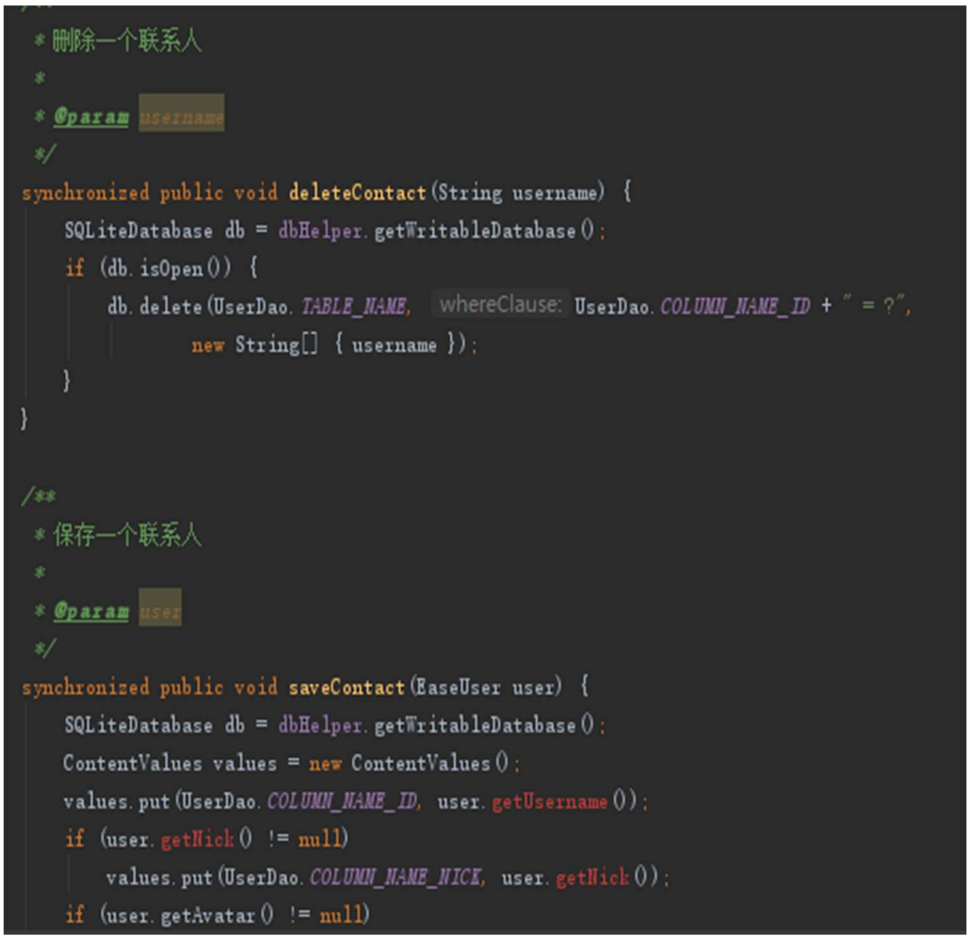


```
zerodistance [C:\AndroidProject\zerodistance] - ...\src\
File Edit View Navigate Code Analyze Refactor B
zerodistance > src > main > java > cn >
Android
▼ zerodistance
  ▼ java
    ▼ android.support
      ▶ graphics.drawable
      ▶ v4
      ▶ v7.appcompat
    ▼ cn
      ▼ em.sdk
        ▶ base
        ▶ bean
        ▼ db
            DbOpenHelper
            DemoDBManager
            InviteMessgeDao
            UserDao
        ▼ manager
            EaseNotifier
            PreferenceManager
        ▼ util
            EaseCommonUtils
      ▼ hzyc.im
        ▶ base
        ▼ manager
            BaseManager
            ConnectionManager
            ContactManager
            IMEventManager
```



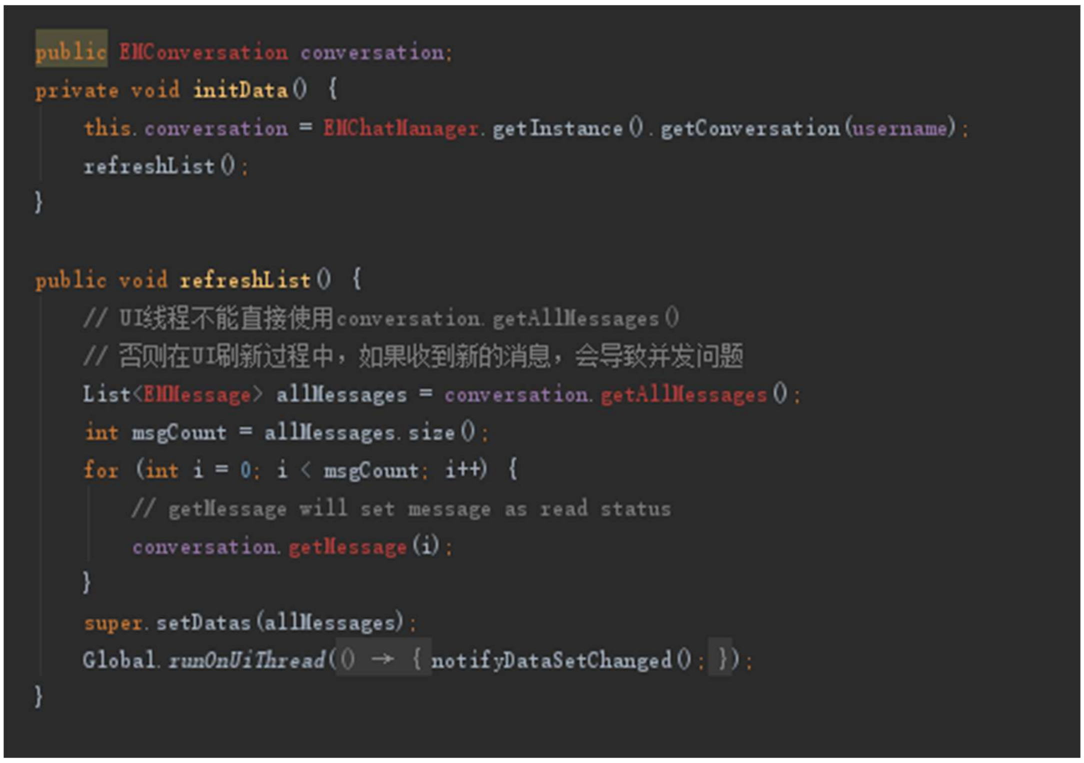
```
        ▼ activity
            AddContactActivity
            ChatActivity
            InviteMsgActivity
            LoginActivity
            MainActivity
            MyMusicActivity
            PlayActivity
            RegisterActivity
            SplashActivity
        ▶ adapter
        ▶ fragment
        ▶ holder
        ▶ view
          BaseActivity
          BaseFragment
      ▶ util
    ▶ cn.hzyc.im
    ▶ cn.hzyc.im
  ▶ tests
  ▶ resources
    test-resources
Gradle Scripts
    build.gradle (Module: zerodistance)
    gradle-wrapper.properties (Gradle Version)
    proguard-rules.pro (ProGuard Rules for zer
```

数据库中联系人操作：



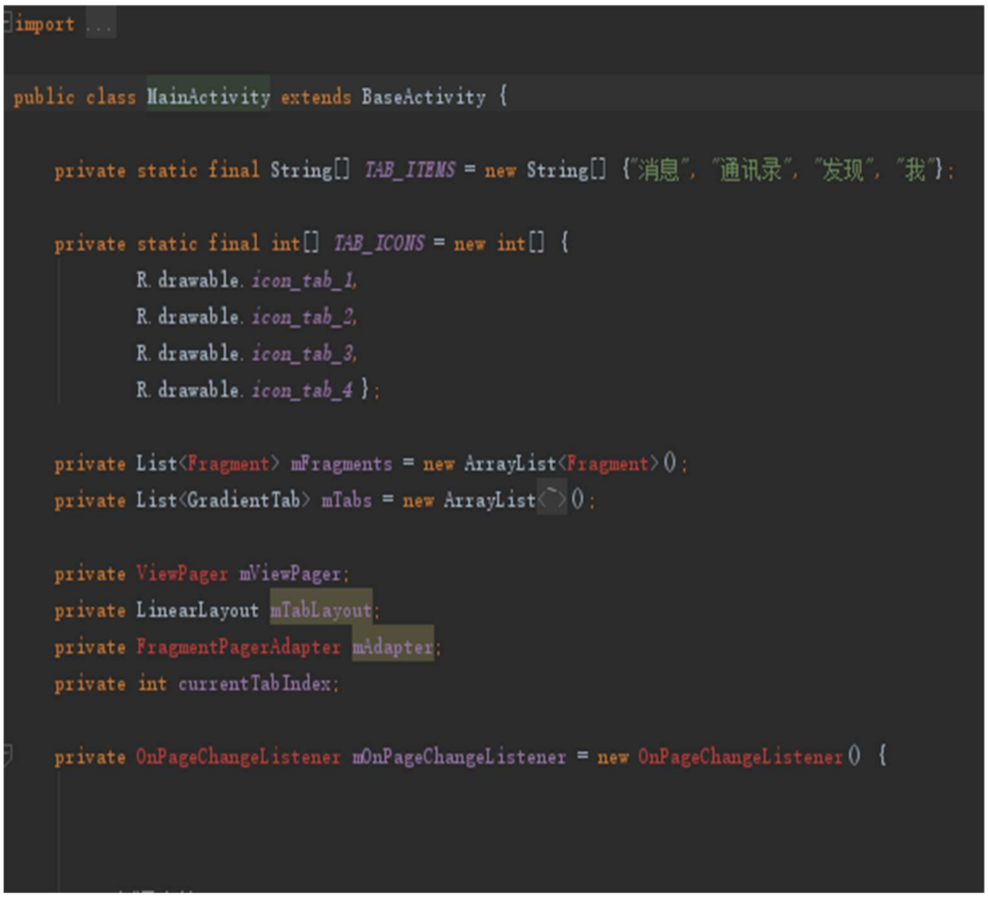
```
 * 删除一个联系人
 *
 * @param username
 */
synchronized public void deleteContact(String username) {
    SQLiteDatabase db = dbHelper.getWritableDatabase();
    if (db.isOpen()) {
        db.delete(UserDao.TABLE_NAME,  whereClause: UserDao.COLUMN_NAME_ID + " = ?",
                new String[] { username });
    }
}

/**
 * 保存一个联系人
 *
 * @param user
 */
synchronized public void saveContact(EaseUser user) {
    SQLiteDatabase db = dbHelper.getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put(UserDao.COLUMN_NAME_ID, user.getUsername());
    if (user.getNick() != null)
        values.put(UserDao.COLUMN_NAME_NICK, user.getNick());
    if (user.getAvatar() != null)
```

Chatadpte 功能 r:



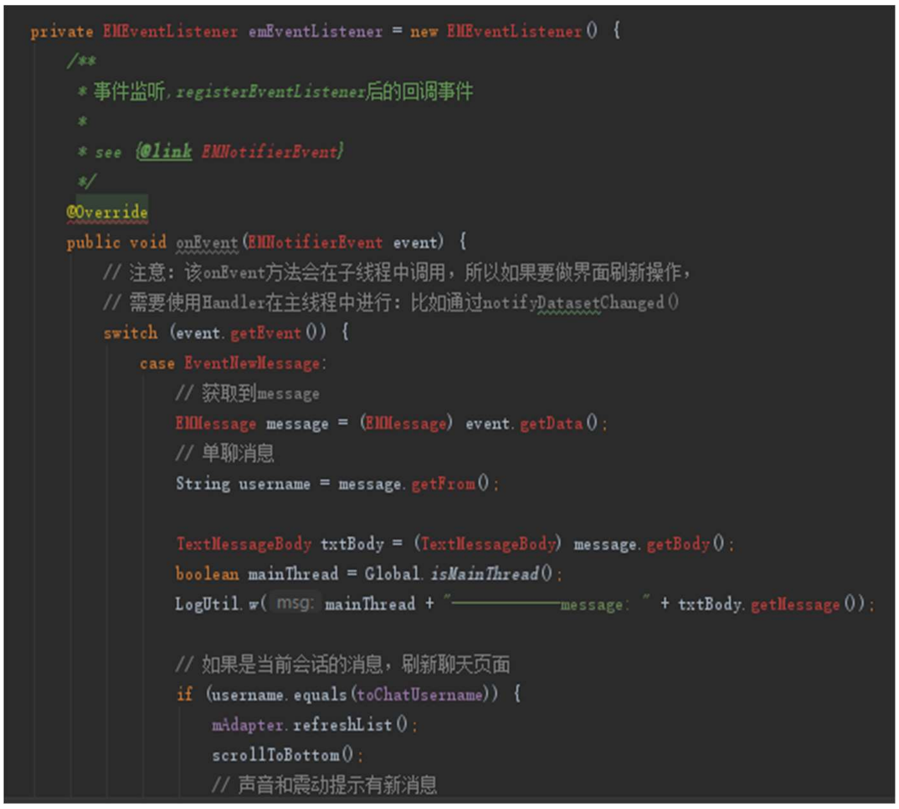
```
public EMConversation conversation;
private void initData() {
    this.conversation = EMChatManager.getInstance().getConversation(username);
    refreshList();
}

public void refreshList() {
    // UI线程不能直接使用conversation.getAllMessages()
    // 否则在UI刷新过程中，如果收到新的消息，会导致并发问题
    List<EMMessage> allMessages = conversation.getAllMessages();
    int msgCount = allMessages.size();
    for (int i = 0; i < msgCount; i++) {
        // getMessage will set message as read status
        conversation.getMessage(i);
    }
    super.setDatas(allMessages);
    Global.runOnUiThread(() -> { notifyDataSetChanged(); });
}
```

main 界面：



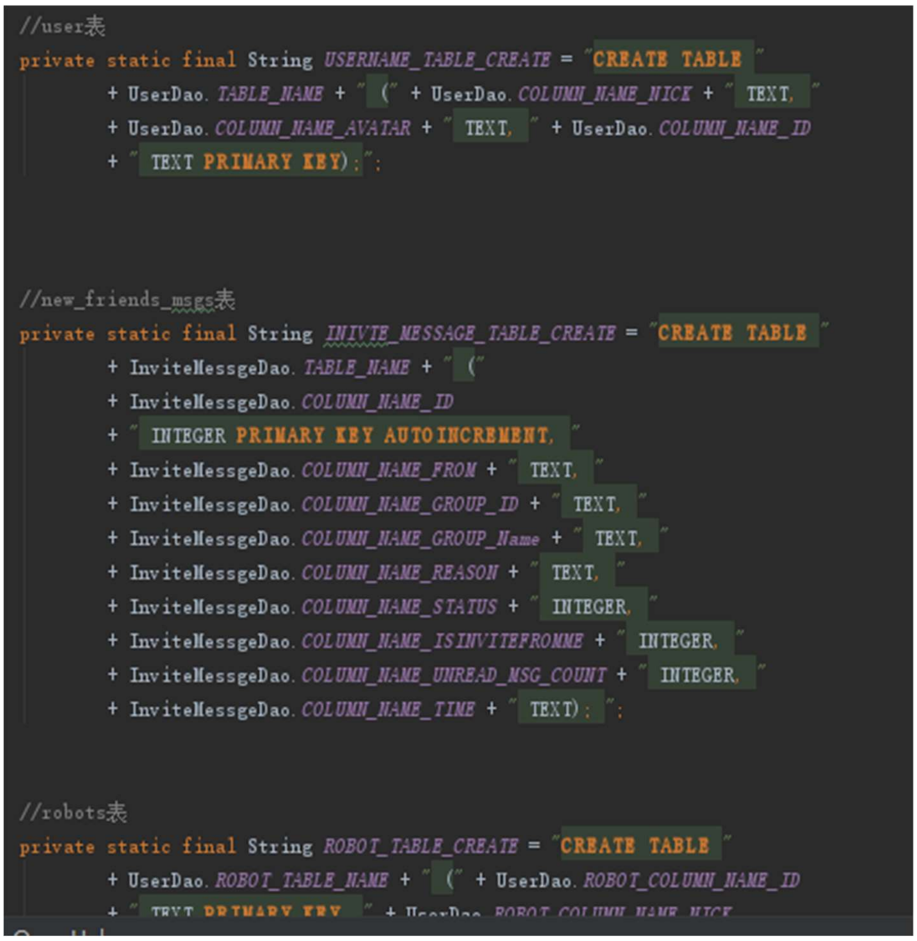
```
import ...

public class MainActivity extends BaseActivity {

    private static final String[] TAB_ITEMS = new String[] {"消息", "通讯录", "发现", "我"};

    private static final int[] TAB_ICONS = new int[] {
            R.drawable.icon_tab_1,
            R.drawable.icon_tab_2,
            R.drawable.icon_tab_3,
            R.drawable.icon_tab_4 };

    private List<Fragment> mFragments = new ArrayList<Fragment>();
    private List<GradientTab> mTabs = new ArrayList<~>();

    private ViewPager mViewPager;
    private LinearLayout mTabLayout;
    private FragmentPagerAdapter mAdapter;
    private int currentTabIndex;

    private OnPageChangeListener mOnPageChangeListener = new OnPageChangeListener() {
```

Message　　　　　　　　　接　　　　　　　　受　　　　　　　　：



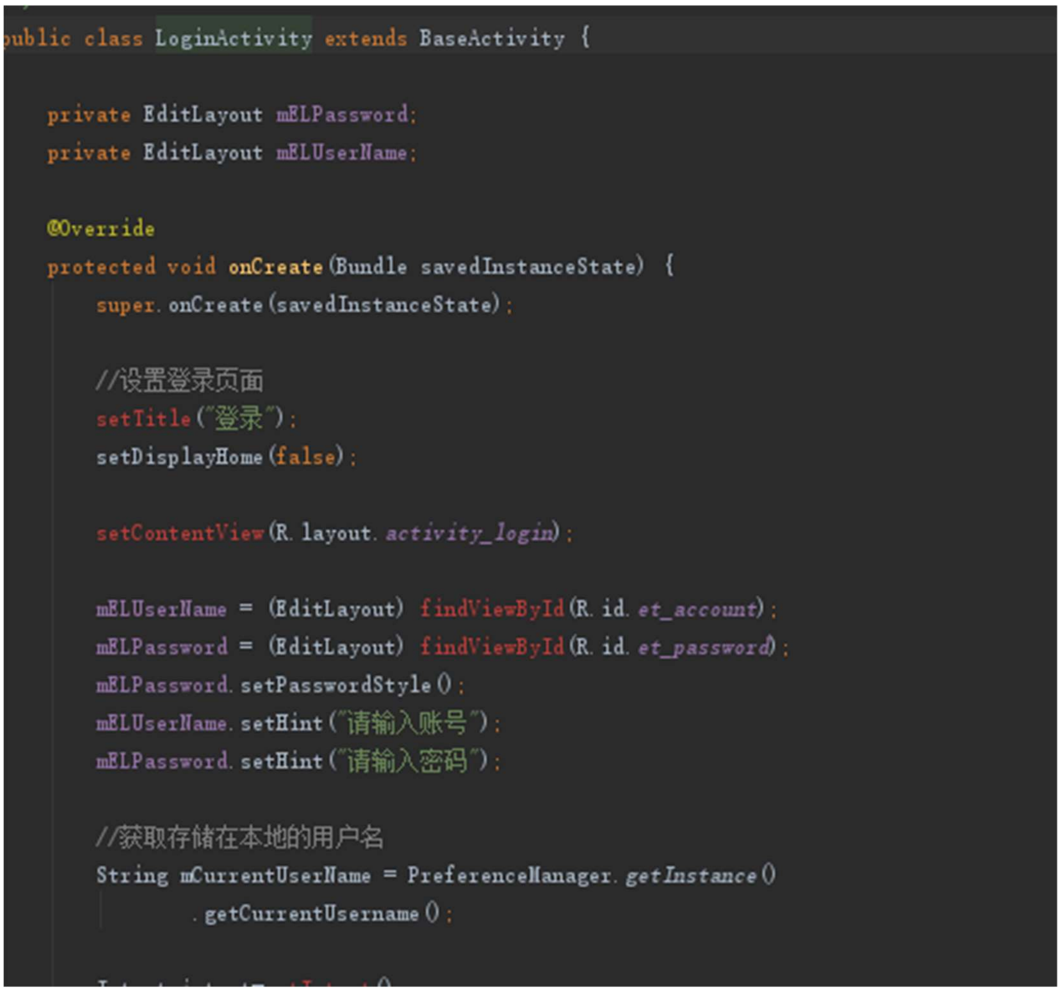
```java
private EMEventListener emEventListener = new EMEventListener() {
    /**
     * 事件监听, registerEventListener后的回调事件
     *
     * see {@link EMNotifierEvent}
     */
    @Override
    public void onEvent(EMNotifierEvent event) {
        // 注意：该onEvent方法会在子线程中调用，所以如果要做界面刷新操作，
        // 需要使用Handler在主线程中进行：比如通过notifyDatasetChanged()
        switch (event.getEvent()) {
            case EventNewMessage:
                // 获取到message
                EMMessage message = (EMMessage) event.getData();
                // 单聊消息
                String username = message.getFrom();

                TextMessageBody txtBody = (TextMessageBody) message.getBody();
                boolean mainThread = Global.isMainThread();
                LogUtil.w( msg: mainThread + "————————message: " + txtBody.getMessage());

                // 如果是当前会话的消息，刷新聊天页面
                if (username.equals(toChatUsername)) {
                    mAdapter.refreshList();
                    scrollToBottom();
                    // 声音和震动提示有新消息
```

User 表：



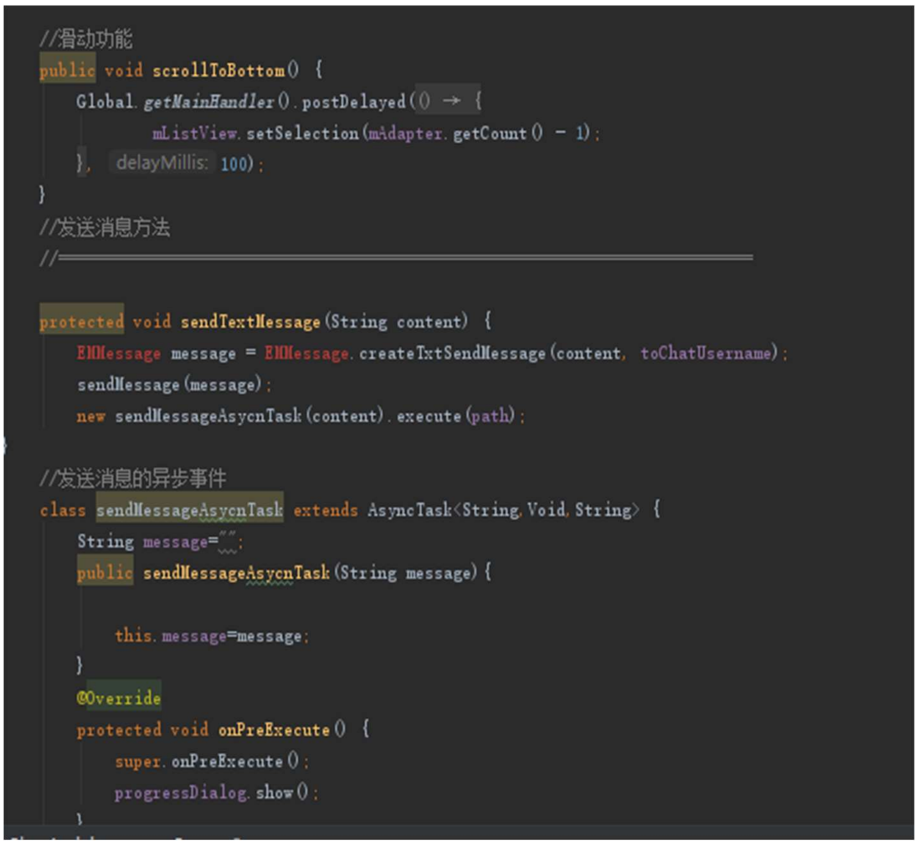
```java
//user表
private static final String USERNAME_TABLE_CREATE = "CREATE TABLE "
        + UserDao.TABLE_NAME + " (" + UserDao.COLUMN_NAME_NICK + " TEXT, "
        + UserDao.COLUMN_NAME_AVATAR + " TEXT, " + UserDao.COLUMN_NAME_ID
        + " TEXT PRIMARY KEY);";



//new_friends_msgs表
private static final String INIVTE_MESSAGE_TABLE_CREATE = "CREATE TABLE "
        + InviteMessgeDao.TABLE_NAME + " ("
        + InviteMessgeDao.COLUMN_NAME_ID
        + " INTEGER PRIMARY KEY AUTOINCREMENT, "
        + InviteMessgeDao.COLUMN_NAME_FROM + " TEXT, "
        + InviteMessgeDao.COLUMN_NAME_GROUP_ID + " TEXT, "
        + InviteMessgeDao.COLUMN_NAME_GROUP_Name + " TEXT, "
        + InviteMessgeDao.COLUMN_NAME_REASON + " TEXT, "
        + InviteMessgeDao.COLUMN_NAME_STATUS + " INTEGER, "
        + InviteMessgeDao.COLUMN_NAME_ISINVITEFROMME + " INTEGER, "
        + InviteMessgeDao.COLUMN_NAME_UNREAD_MSG_COUNT + " INTEGER, "
        + InviteMessgeDao.COLUMN_NAME_TIME + " TEXT); ";


//robots表
private static final String ROBOT_TABLE_CREATE = "CREATE TABLE "
        + UserDao.ROBOT_TABLE_NAME + " (" + UserDao.ROBOT_COLUMN_NAME_ID
        + " TEXT PRIMARY KEY, " + UserDao.ROBOT_COLUMN_NAME_NICK
```

登陆：

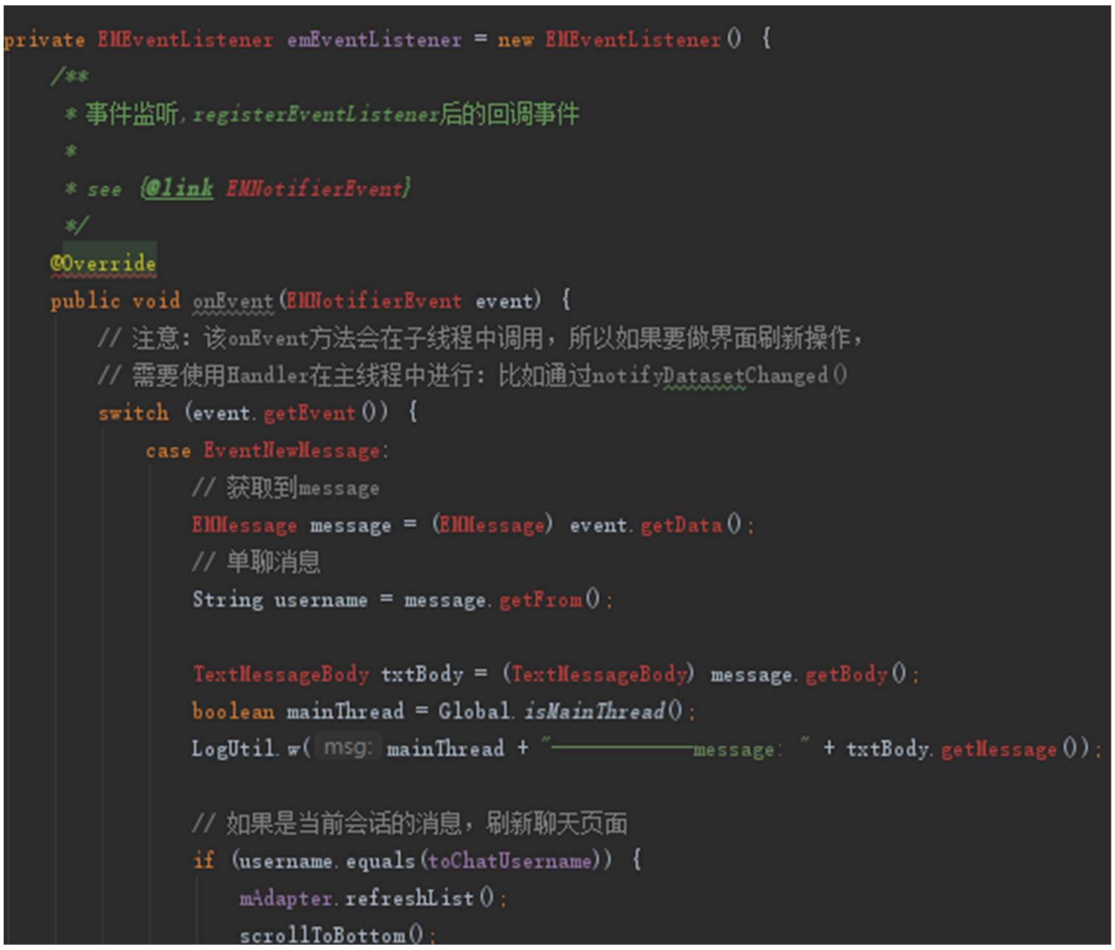


```
public class LoginActivity extends BaseActivity {

    private EditLayout mELPassword;
    private EditLayout mELUserName;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        //设置登录页面
        setTitle("登录");
        setDisplayHome(false);

        setContentView(R.layout.activity_login);

        mELUserName = (EditLayout) findViewById(R.id.et_account);
        mELPassword = (EditLayout) findViewById(R.id.et_password);
        mELPassword.setPasswordStyle();
        mELUserName.setHint("请输入账号");
        mELPassword.setHint("请输入密码");

        //获取存储在本地的用户名
        String mCurrentUserName = PreferenceManager.getInstance()
                .getCurrentUsername();
```

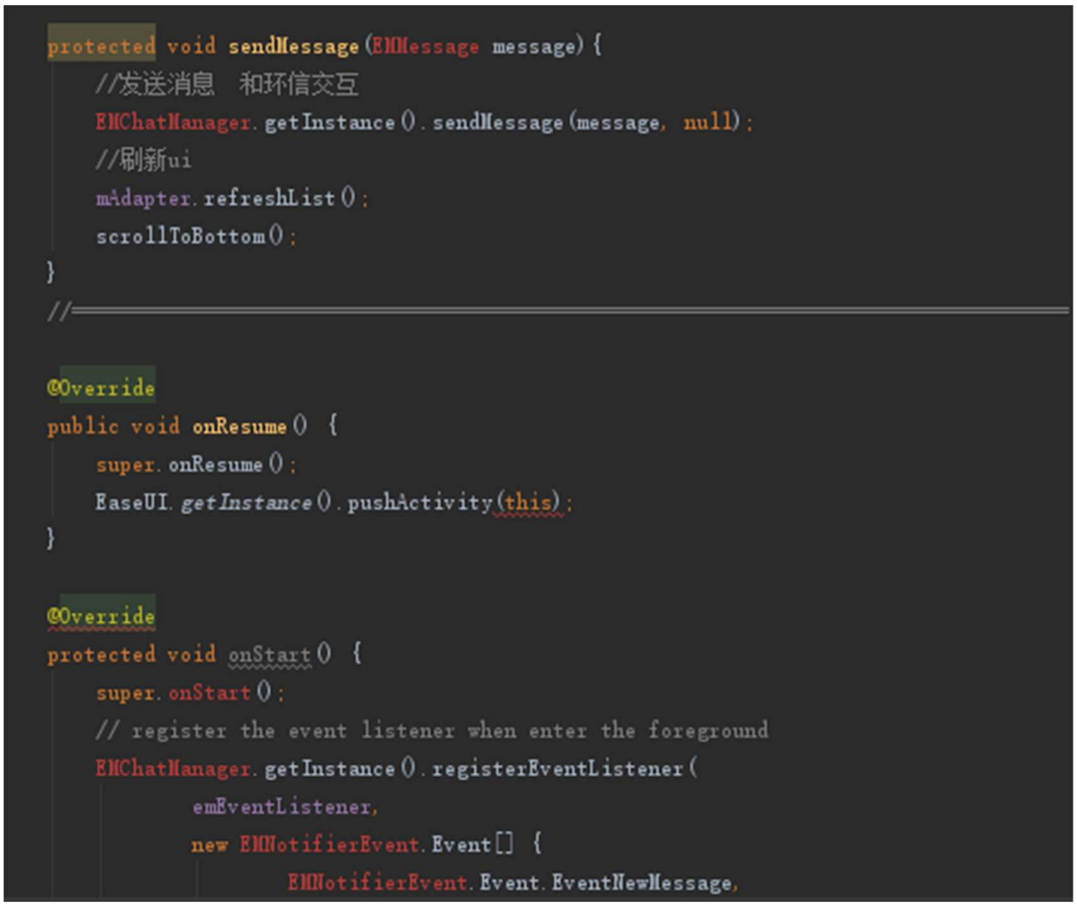
发送消息，与环信交互



```
//滑动功能
public void scrollToBottom() {
    Global.getMainHandler().postDelayed(() -> {
            mListView.setSelection(mAdapter.getCount() - 1);
    }, delayMillis: 100);
}
//发送消息方法
//═══════════════════════════════════════════════════

protected void sendTextMessage(String content) {
    EMMessage message = EMMessage.createTxtSendMessage(content, toChatUsername);
    sendMessage(message);
    new sendMessageAsycnTask(content).execute(path);
}
//发送消息的异步事件
class sendMessageAsycnTask extends AsyncTask<String,Void,String> {
    String message="";
    public sendMessageAsycnTask(String message){

        this.message=message;
    }
    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        progressDialog.show();
    }
```

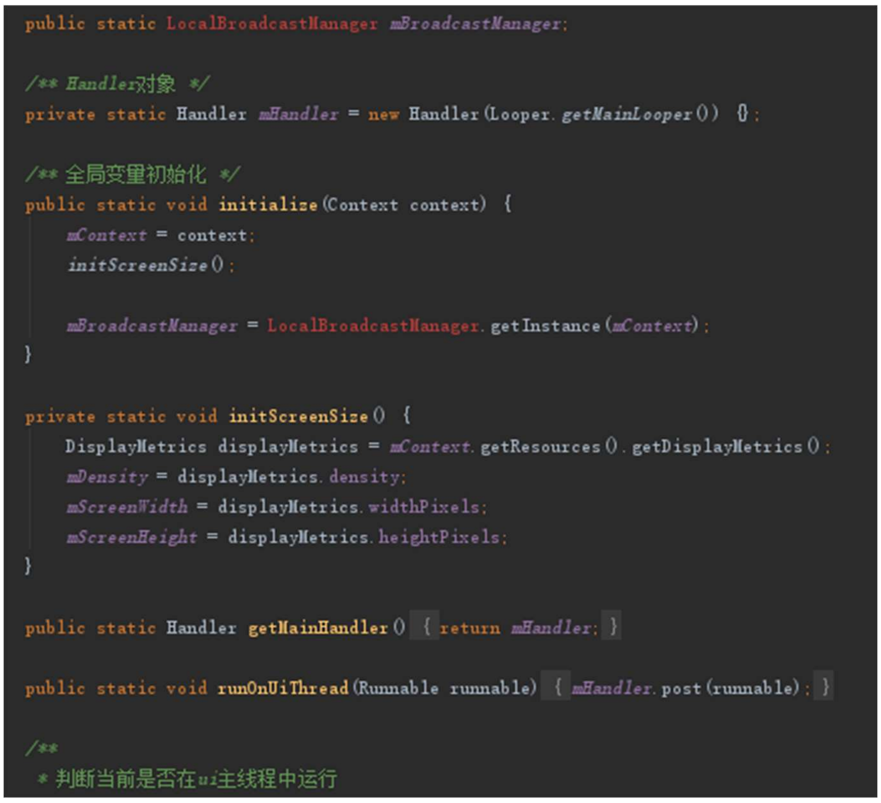


```
private EMEventListener emEventListener = new EMEventListener() {
    /**
     * 事件监听,registerEventListener后的回调事件
     *
     * see {@link EMNotifierEvent}
     */
    @Override
    public void onEvent(EMNotifierEvent event) {
        // 注意：该onEvent方法会在子线程中调用，所以如果要做界面刷新操作，
        // 需要使用Handler在主线程中进行：比如通过notifyDatasetChanged()
        switch (event.getEvent()) {
            case EventNewMessage:
                // 获取到message
                EMMessage message = (EMMessage) event.getData();
                // 单聊消息
                String username = message.getFrom();

                TextMessageBody txtBody = (TextMessageBody) message.getBody();
                boolean mainThread = Global.isMainThread();
                LogUtil.w( msg: mainThread + "————————message: " + txtBody.getMessage());

                // 如果是当前会话的消息，刷新聊天页面
                if (username.equals(toChatUsername)) {
                    mAdapter.refreshList();
                    scrollToBottom();
```

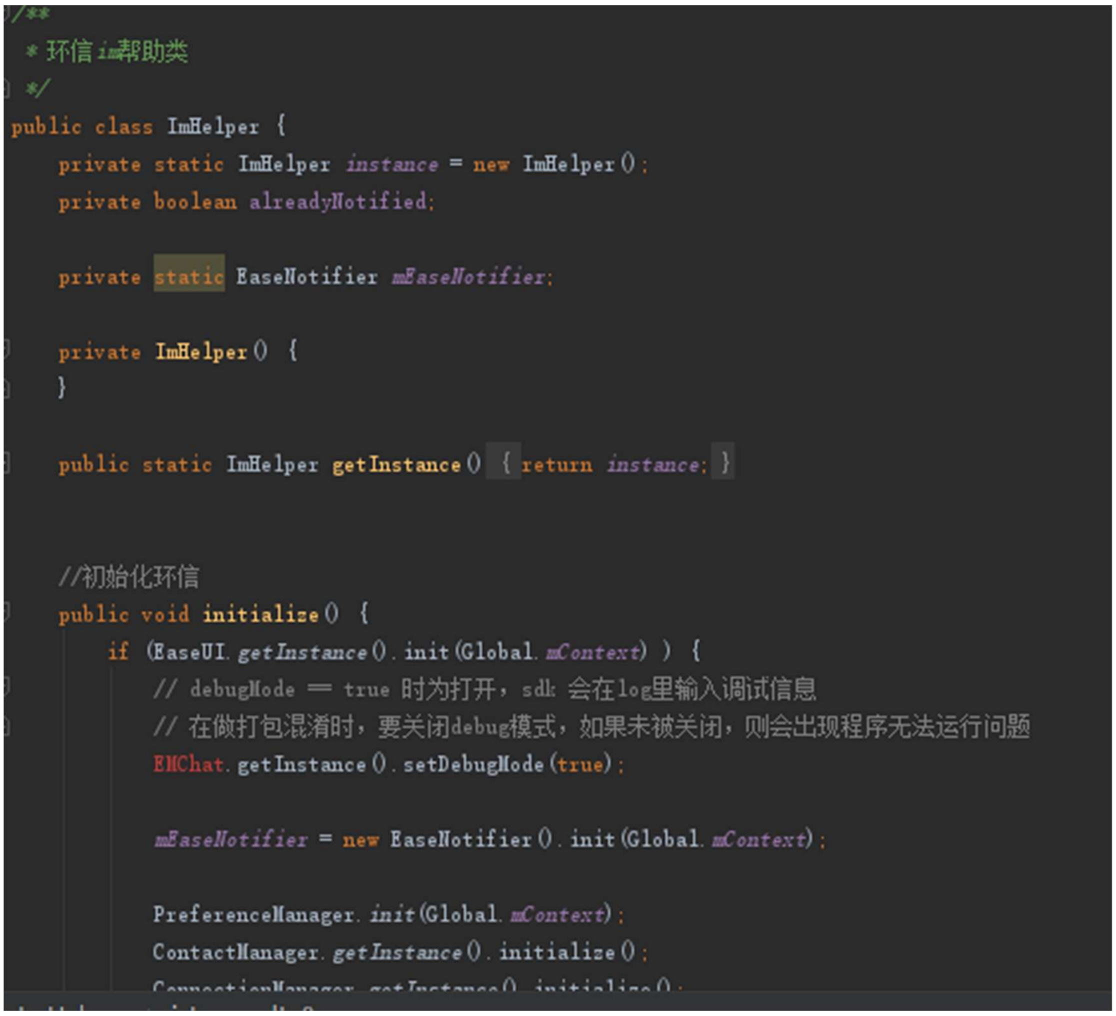


```
protected void sendMessage(EMMessage message){
    //发送消息  和环信交互
    EMChatManager.getInstance().sendMessage(message, null);
    //刷新ui
    mAdapter.refreshList();
    scrollToBottom();
}
//================================================================================

@Override
public void onResume() {
    super.onResume();
    EaseUI.getInstance().pushActivity(this);
}

@Override
protected void onStart() {
    super.onStart();
    // register the event listener when enter the foreground
    EMChatManager.getInstance().registerEventListener(
            emEventListener,
            new EMNotifierEvent.Event[] {
                    EMNotifierEvent.Event.EventNewMessage,
```

环信端交互 global：

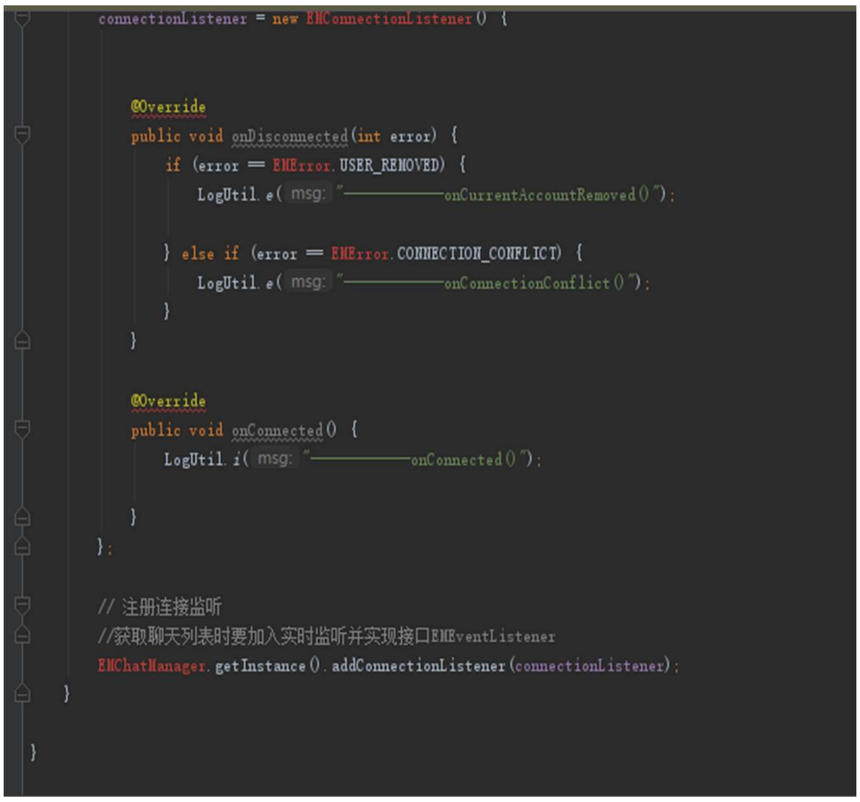


```
public static LocalBroadcastManager mBroadcastManager;

/** Handler对象 */
private static Handler mHandler = new Handler(Looper.getMainLooper()) {};

/** 全局变量初始化 */
public static void initialize(Context context) {
    mContext = context;
    initScreenSize();

    mBroadcastManager = LocalBroadcastManager.getInstance(mContext);
}

private static void initScreenSize() {
    DisplayMetrics displayMetrics = mContext.getResources().getDisplayMetrics();
    mDensity = displayMetrics.density;
    mScreenWidth = displayMetrics.widthPixels;
    mScreenHeight = displayMetrics.heightPixels;
}

public static Handler getMainHandler() { return mHandler; }

public static void runOnUiThread(Runnable runnable) { mHandler.post(runnable); }

/**
 * 判断当前是否在ui主线程中运行
```



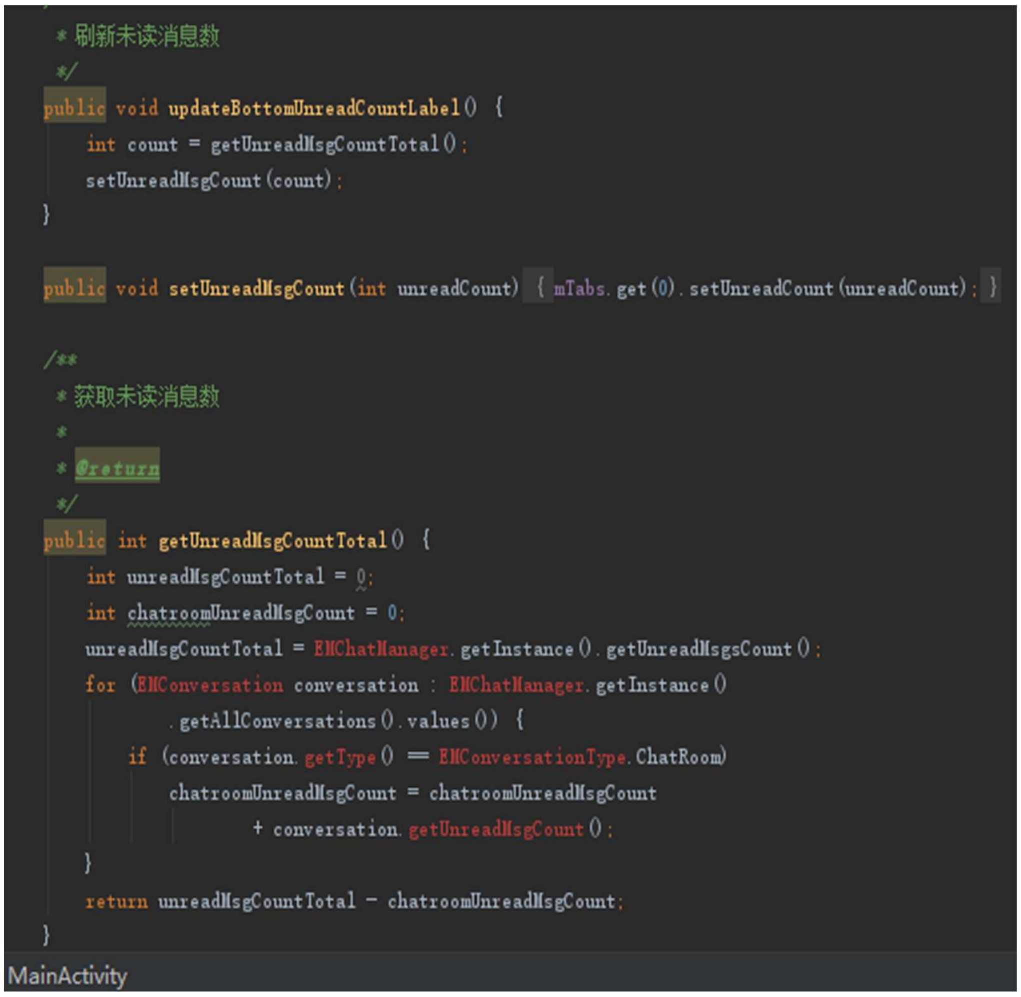
```java
/**
 * 环信 im帮助类
 */
public class ImHelper {
    private static ImHelper instance = new ImHelper();
    private boolean alreadyNotified;

    private static EaseNotifier mEaseNotifier;

    private ImHelper() {
    }

    public static ImHelper getInstance() { return instance; }


    //初始化环信
    public void initialize() {
        if (EaseUI.getInstance().init(Global.mContext) ) {
            // debugMode == true 时为打开，sdk 会在log里输入调试信息
            // 在做打包混淆时，要关闭debug模式，如果未被关闭，则会出现程序无法运行问题
            EMChat.getInstance().setDebugMode(true);

            mEaseNotifier = new EaseNotifier().init(Global.mContext);

            PreferenceManager.init(Global.mContext);
            ContactManager.getInstance().initialize();
            ConnectionManager.getInstance().initialize();
```

事件监听；



```
        connectionListener = new EMConnectionListener() {


            @Override
            public void onDisconnected(int error) {
                if (error == EMError.USER_REMOVED) {
                    LogUtil.e( msg: "——————onCurrentAccountRemoved()");

                } else if (error == EMError.CONNECTION_CONFLICT) {
                    LogUtil.e( msg: "——————onConnectionConflict()");
                }
            }

            @Override
            public void onConnected() {
                LogUtil.i( msg: "——————onConnected()");

            }
        };

        // 注册连接监听
        //获取聊天列表时要加入实时监听并实现接口EMEventListener
        EMChatManager.getInstance().addConnectionListener(connectionListener);
    }

}
```

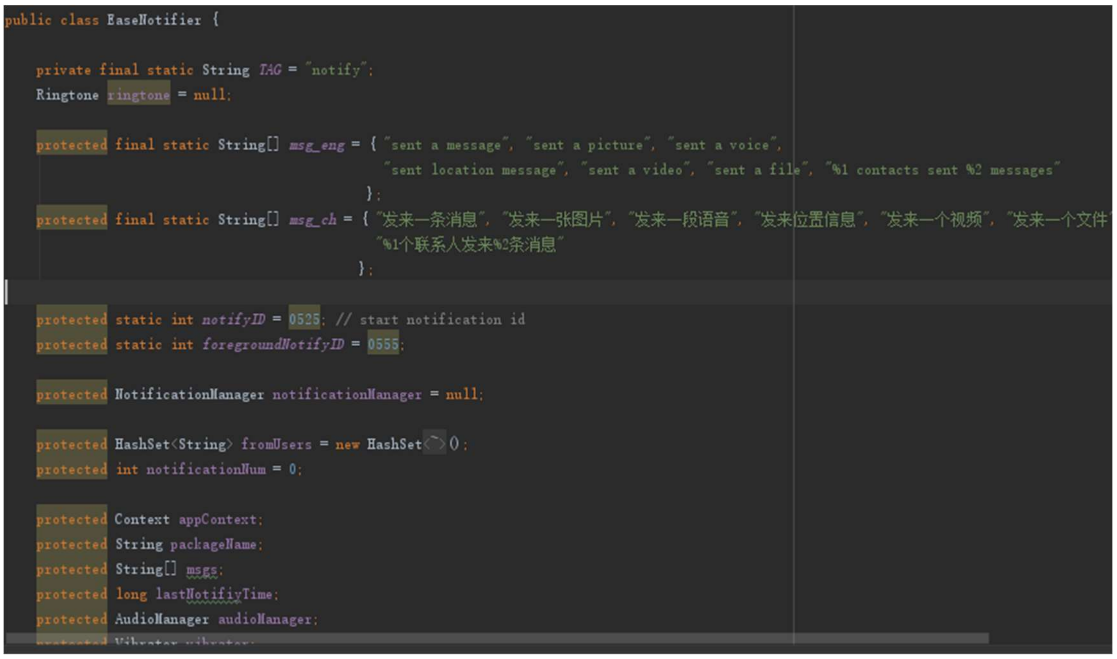
未读消息



```java
    * 刷新未读消息数
    */
   public void updateBottomUnreadCountLabel() {
       int count = getUnreadMsgCountTotal();
       setUnreadMsgCount(count);
   }

   public void setUnreadMsgCount(int unreadCount) { mTabs.get(0).setUnreadCount(unreadCount); }

   /**
    * 获取未读消息数
    *
    * @return
    */
   public int getUnreadMsgCountTotal() {
       int unreadMsgCountTotal = 0;
       int chatroomUnreadMsgCount = 0;
       unreadMsgCountTotal = EMChatManager.getInstance().getUnreadMsgsCount();
       for (EMConversation conversation : EMChatManager.getInstance()
               .getAllConversations().values()) {
           if (conversation.getType() == EMConversationType.ChatRoom)
               chatroomUnreadMsgCount = chatroomUnreadMsgCount
                       + conversation.getUnreadMsgCount();
       }
       return unreadMsgCountTotal - chatroomUnreadMsgCount;
   }
```

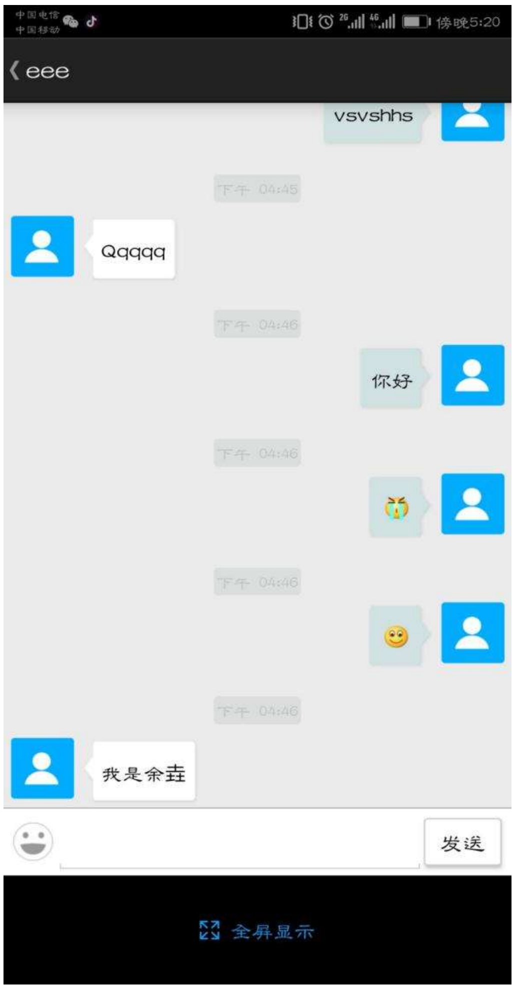
MainActivity

新消息提醒 class：



```java
public class EaseNotifier {

    private final static String TAG = "notify";
    Ringtone ringtone = null;

    protected final static String[] msg_eng = { "sent a message", "sent a picture", "sent a voice",
                                                "sent location message", "sent a video", "sent a file", "%1 contacts sent %2 messages"
                                              };
    protected final static String[] msg_ch = { "发来一条消息", "发来一张图片", "发来一段语音", "发来位置信息", "发来一个视频", "发来一个文件"
                                               "%1个联系人发来%2条消息"
                                             };

    protected static int notifyID = 0525; // start notification id
    protected static int foregroundNotifyID = 0555;

    protected NotificationManager notificationManager = null;

    protected HashSet<String> fromUsers = new HashSet<>();
    protected int notificationNum = 0;

    protected Context appContext;
    protected String packageName;
    protected String[] msgs;
    protected long lastNotifiyTime;
    protected AudioManager audioManager;
    protected Vibrator vibrator;
```

6. 测试


中国电信
中国移动
傍晚5:20

eee

vsvshhs

下午 04:45

Qqqqq

下午 04:46

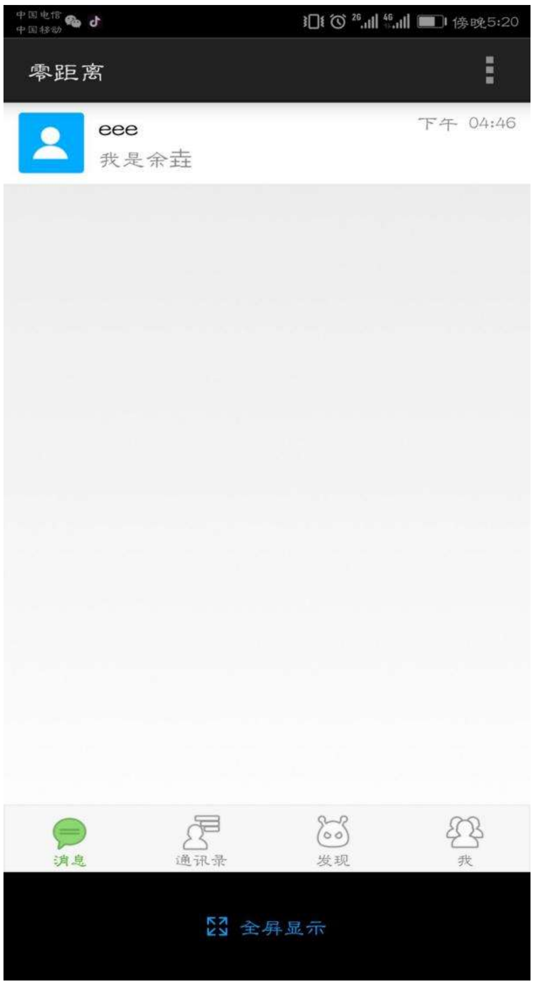
你好

下午 04:46

下午 04:46

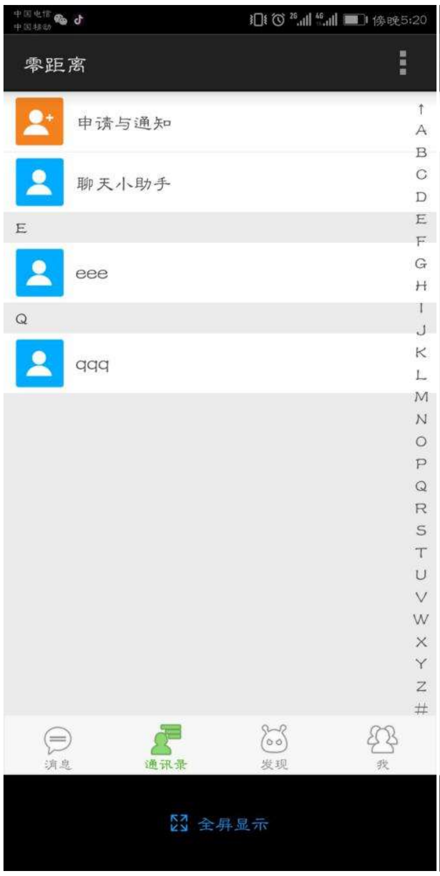
下午 04:46

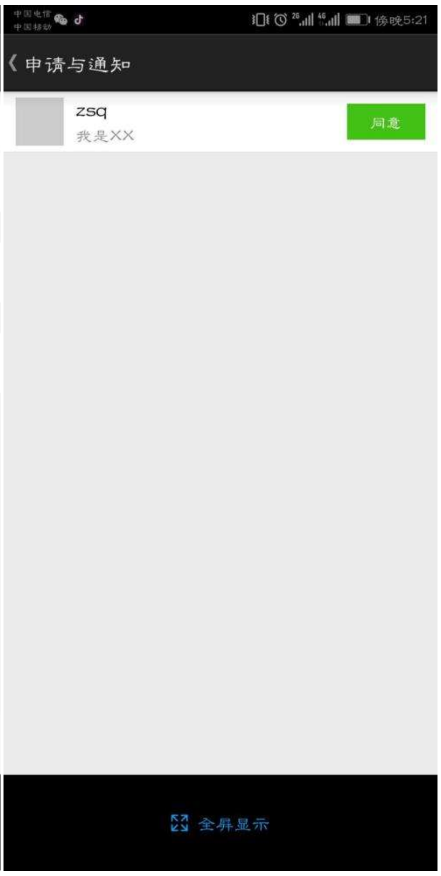
我是余垚

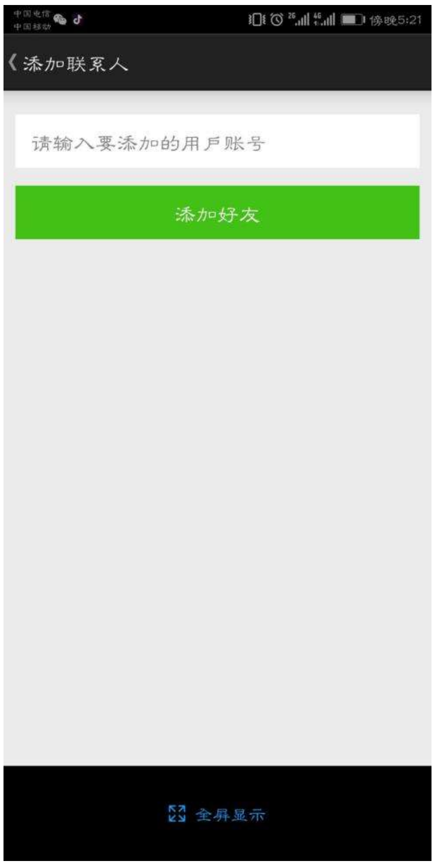

中国电信
中国移动
傍晚5:20

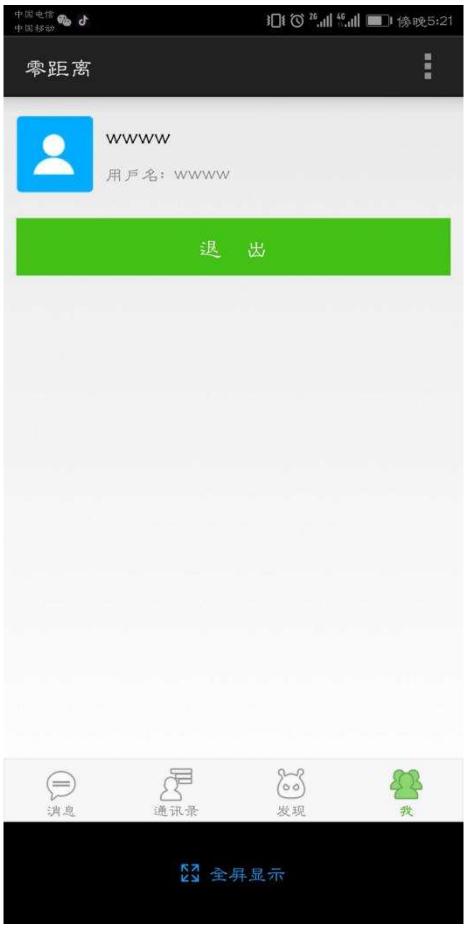
零距离

eee 下午 04:46
我是余垚


零距离
申请与通知
聊天小助手
E
eee
Q
qqq
↑ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #
消息
通讯录
发现
我
全屏显示


申请与通知
zsq
我是XX
同意
全屏显示


添加联系人
请输入要添加的用户账号
添加好友
全屏显示


零距离
wwww
用户名：wwww
退　出
消息
通讯录
发现
我
全屏显示

7.评价及总结

　通过几周学习，了解了 Android 环境的配置，编程的规范等。将 Java 语言运用到了实例当中，理解了 SQL 语句的使用，熟悉了 XML 文件的运用和编写，可以进一步完善功能和美化界面。

程序均为连接自己的安卓手机调试，更直观感觉到功能和界面的不完善，有待学习进步。