程序设计实践训练课程设计报告

姓 名：吴泽锦、全栩剑

班 级：15 通信 1 班

学 号：201521190421

201521190418

指导老师：徐兴，俞龙

日期：2018.4.22～2018.6.15

华南农业大学工程学院

前　言

本课程设计目的在于实现一款在局域网内通信的聊天程序软件，由于笔者早已有类似的程序设计思路，故本课程设计的实现内容仅作为整体框架的一部分，接下来介绍该内容的实现流程。

本设计只考虑在 Android 平台实现该软件，至于 iOS 平台的实现以及 PC 端的实现此处并不做具体分析与介绍。

本设计基于 Android 手机平台，在 windows8.1 操作系统上采用 AndroidStudio 3.1 集成开发平台进行开发,最低兼容的软件运行环境为 Android 4.2,更多关于 Application 的配置信息可参考附件 1。

实现的功能有：局域网内进行通信（私聊、群聊），数据库存取。

目　录

一、 实验要求

基本功能要求

客户端部分：

- 手动输入服务器端 IP 地址和端口号进行连接
- 发送消息给服务器端并显示服务器端回传的消息

服务器端部分：

- 手动建立服务器端与客户端的连接请求
- 接收所有用户发送的消息
- 向所有在线用户群发消息

附加功能

- 在客户端和服务器端分别增加保存聊天记录的功能

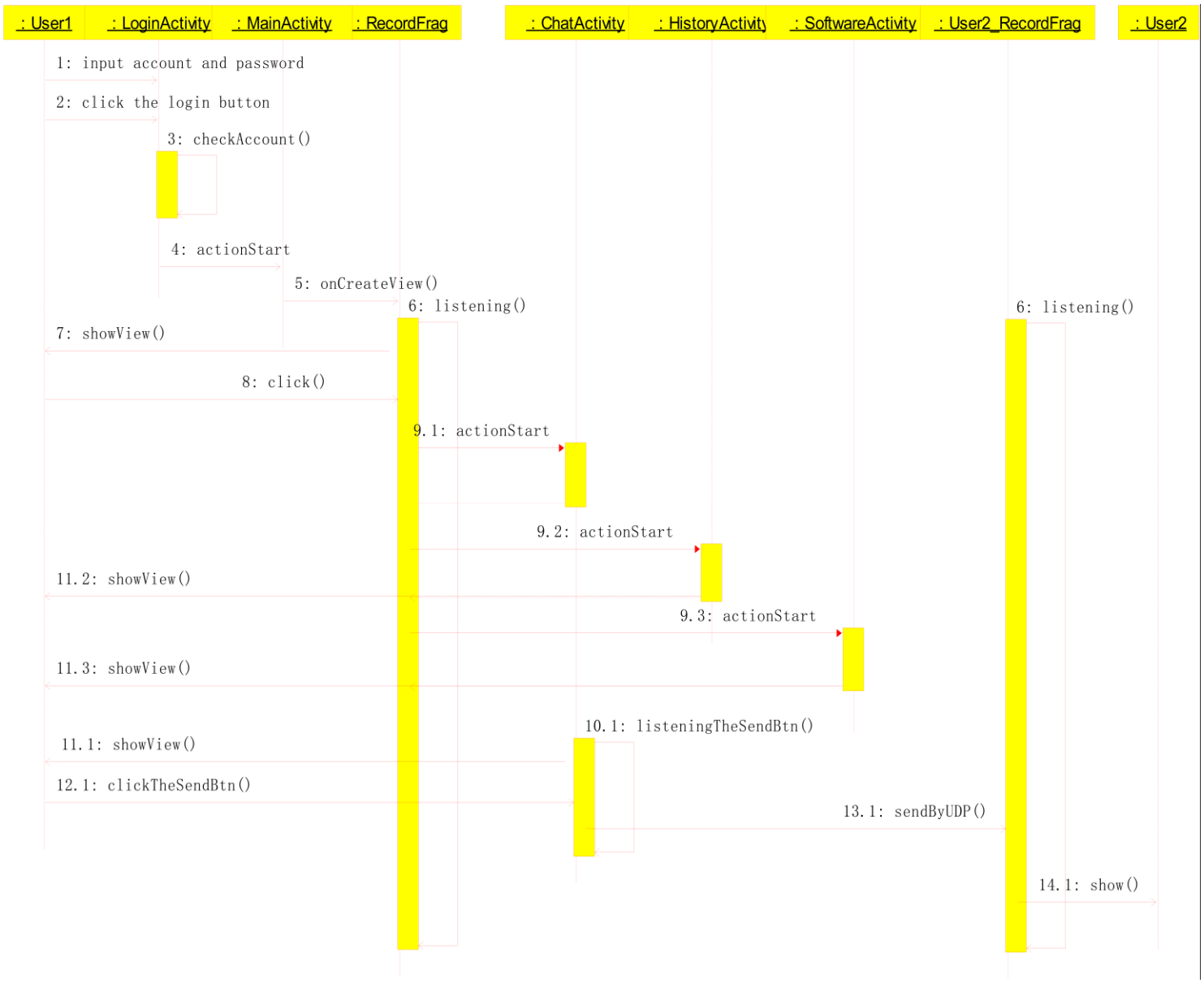

图 1　程序流程图

二、 程序分析与框架搭建

2.1 程序流程分析

如图 1 所示，用户 1 启动软件后，进入 LoginActivity，输入账号和密码（步骤 1），并点击登录按钮后（步骤 2），LoginActivity 调用 checkAccount()，对账号和密码进行验证（步骤 3），验证成功则启动主活动（步骤 4），主活动启动 RecordFrag 碎片（步骤 5），碎片进入监听状态（步骤 6），并向用户 1 展示界面（步骤 7），用户点击按钮（步骤 8），如果按钮是聊天按钮，则启动聊天活动（步骤 9.1），如果按钮是聊天记录查看按钮，则启动历史查看活动（步骤 9.2），如果按钮是查看软件信息，则启动关于软件的活动（步骤 9.3）；

当用户启动聊天活动后，聊天活动进入监听状态（步骤 10.1），此时向用户展示界面（步骤 11.1），用户编辑消息并点击发送按钮后（步骤 12.1），聊天活动启动发送程序，向用户 2 的 RecordFrag 碎片发送消息（步骤 13.1），该碎片收到消息后，向用户展示消息（步骤 14.1）。

以上仅是用户运行程序时的参考步骤，实际情况比这复杂得多，例如：RecordFrag 碎片中尚未包括 Receiver 的消息接收模块，由于二维时序图展示信息有限，并没有列出该步骤；又如：碎片接收到消息后，并非仅展示消息，还需要对数据进行选择性展示（选择展示在私聊界面还是群聊界面），展示前还需要保存数据进相应的数据表中。实际编程实现时需要考虑的因素还有很多，这里给出的时序图仅作参考。

2.2 功能分析

在局域网内聊天的功能并不需要真正地区分服务端与客户端，原因在于并不存在服务对象与被服务对象的相关关系，故本设计不对 Server 和 Client 进行明确的区分，而是采用 P2P 对等关系模型。

针对实验要求与实际情况，本设计实际在于实现一款类似于 WeChat 的软件，在该软件中实现局域网内私聊（局域网网络稳定，不易丢包，故采用 UDP 进行通信）、群聊（根据计算机网络技术，采用特殊 IP 地址进行封包即可实现群聊）、数据存取（采用 Android 自带 MySQL 数据库）。类框架如图 2 所示（自行查看清晰版图片）。

目前包含 6 个包（绿色框图）；1 个测试类（深蓝框图）；6 个活动类，其中包含 1 个活动根类和 5 个普通活动类（黄色框图）；4 个碎片类（橙色框图）；以及 2 个辅助类（浅蓝框图）。接下来根据模块对各大类进行分析。

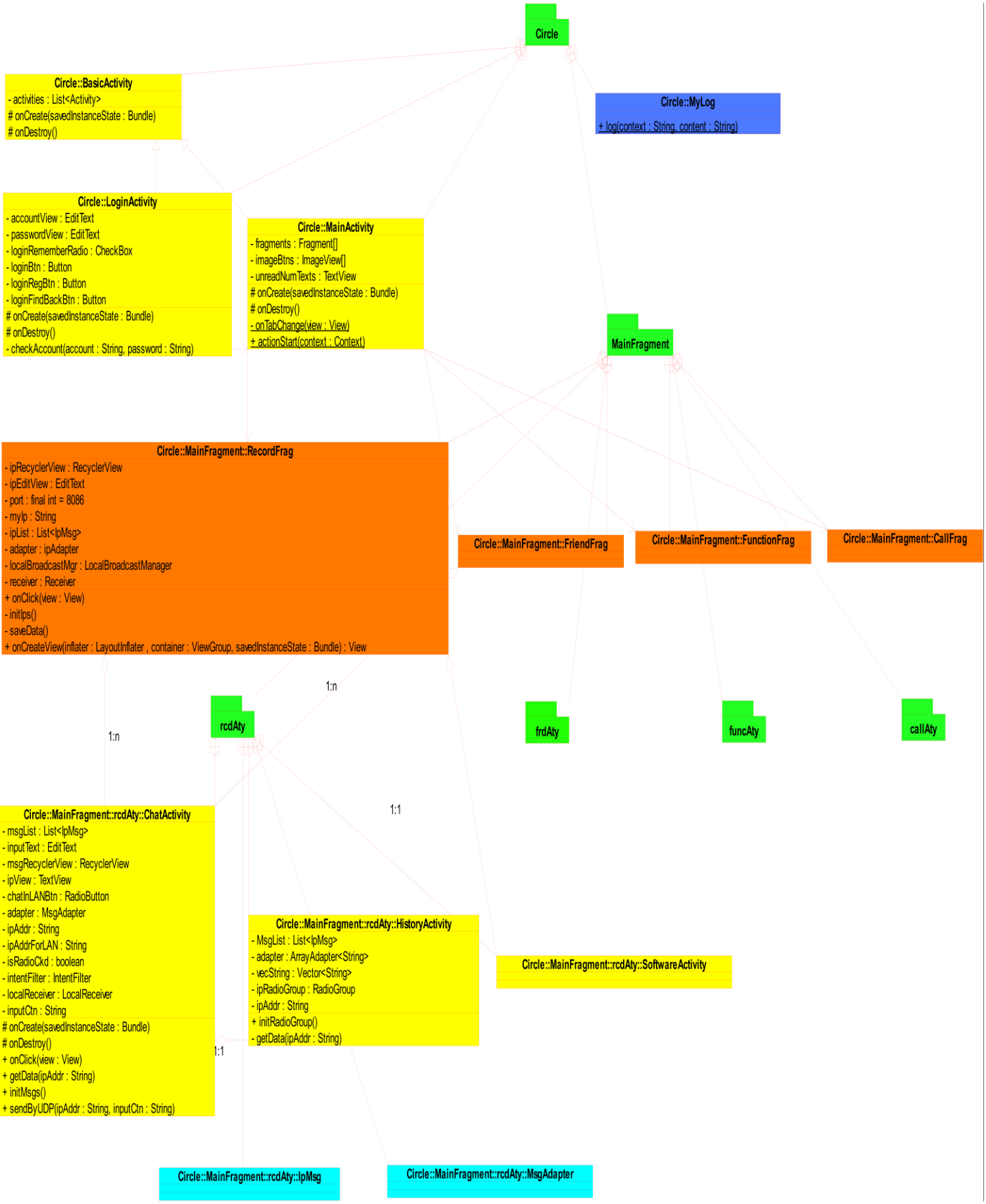
Circle
Circle::BasicActivity
- activities : List<Activity>
onCreate(savedInstanceState : Bundle)
onDestroy()
Circle::MyLog
+ log(context : String, content : String)
Circle::LoginActivity
- accountView : EditText
- passwordView : EditText
- loginRememberRadio : CheckBox
- loginBtn : Button
- loginRegBtn : Button
- loginFindBackBtn : Button
onCreate(savedInstanceState : Bundle)
onDestroy()
- checkAccount(account : String, password : String)
Circle::MainActivity
- fragments : Fragment[]
- imageBtns : ImageView[]
- unreadNumTexts : TextView
onCreate(savedInstanceState : Bundle)
onDestroy()
- onTabChange(view : View)
+ actionStart(context : Context)
MainFragment
Circle::MainFragment::RecordFrag
- ipRecyclerView : RecyclerView
- ipEditView : EditText
- port : final int = 8086
- myIp : String
- ipList : List<IpMsg>
- adapter : ipAdapter
- localBroadcastMgr : LocalBroadcastManager
- receiver : Receiver
+ onClick(view : View)
- initIps()
- saveData()
+ onCreateView(inflater : LayoutInflater , container : ViewGroup, savedInstanceState : Bundle) : View
Circle::MainFragment::FriendFrag
Circle::MainFragment::FunctionFrag
Circle::MainFragment::CallFrag
1:n
1:n
rcdAty
frdAty
funcAty
callAty
Circle::MainFragment::rcdAty::ChatActivity
- msgList : List<IpMsg>
- inputText : EditText
- msgRecyclerView : RecyclerView
- ipView : TextView
- chatInLANBtn : RadioButton
- adapter : MsgAdapter
- ipAddr : String
- ipAddrForLAN : String
- isRadioCkd : boolean
- intentFilter : IntentFilter
- localReceiver : LocalReceiver
- inputCtn : String
onCreate(savedInstanceState : Bundle)
onDestroy()
+ onClick(view : View)
+ getData(ipAddr : String)
+ initMsgs()
+ sendByUDP(ipAddr : String, inputCtn : String)
1:1
1:1
Circle::MainFragment::rcdAty::HistoryActivity
- MsgList : List<IpMsg>
- adapter : ArrayAdapter<String>
- vecString : Vector<String>
- ipRadioGroup : RadioGroup
- ipAddr : String
+ initRadioGroup()
- getData(ipAddr : String)
Circle::MainFragment::rcdAty::SoftwareActivity
Circle::MainFragment::rcdAty::IpMsg
Circle::MainFragment::rcdAty::MsgAdapter

图 2 类框架

2.3 模块分析

Android 中主要存在四大组件：Activity、Broadcast Receiver、Content Provider、Service，本设计主要用到前两个组件，下面针对 Android 界面进行功能分析。

2.2.1 RecordFragment（历史记录碎片）

进入软件时，首先展示 Login 界面（如图 3），这是首个 Activity，登录成功后进入 MainActivity，在 MainActivity 中存在四个 Fragment（如图 4 选中部分），该设计主要内容在其中之一的 RecordFragment 中，即图中的“记录”，至于四大 Fragment 的交互和分析由于与本设计内容无关，故下面只针对 RecordFragment 进行介绍和设计。

图 3 Login 界面

RecordFragment 中展示的是历史信息（如图 4 所示），主要有五大控件：RECORD 按钮、EditText 文本框、CONTACT 按钮、ABOUT 按钮和中间未可视的 RecyclerView。

（1） Record 按钮：点击后进入 HistoryActivity 界面，提供查询聊天功能；

（2） EditText 文本框：可在该框中输入未列举出来的局域网用户 IP 地址，输入对应 IP 后点击 CONTACT 即可进入 ChatActivity 界面；

（3） ABOUT 按钮：点击后进入 SoftwareActivity 界面，提供软件介绍信息。

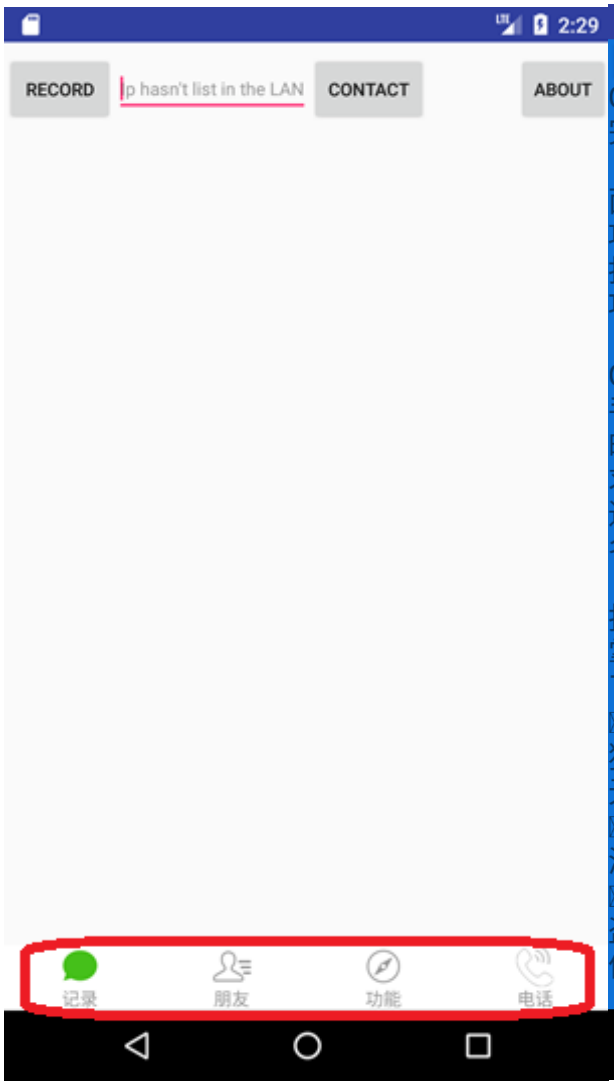

图 4 RecordFragment 碎片

图 5 SoftwareActivity 界面

2.2.2HistoryActivity（聊天查询——数据库）

在 RecordFragment 中用 Intent 启动 HistoryActivity 后，该活动开始初始化，除了一些 View 控件进行初始化外，还应扫描数据库，获得所记录的所有 IP 地址，接着初始化 RadioGroup，即初始化图 6 中左侧的所有 IP 对应的 RadioButton，此时控件初始化完毕（如图 6），可通过选择 RadioButton 查询与对应 IP 用户的聊天信息，此时显示聊天信息于图中靠右的 ListView 中（图中 Item 对应位置即为实际消息显示的位置）。另外，该界面除了提供查询的功能，还提供了删除数据的功能，两个删除按钮位于界面上方。

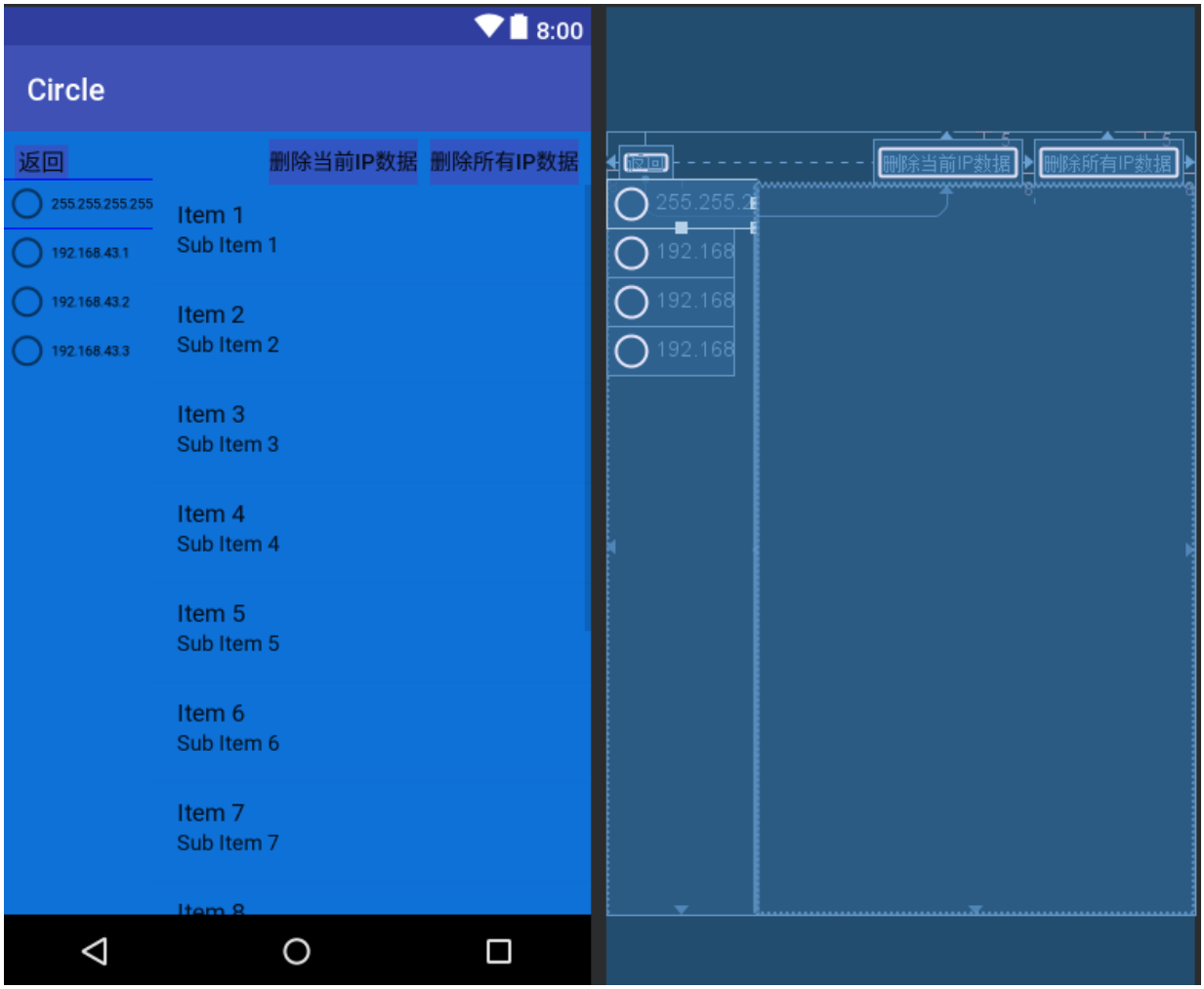

图 6　HistoryActivity 界面

2.2.3ChatActivity（聊天界面）

在 RecordFragment 中用 Intent 启动 ChatActivity 后，该活动进行初始化，初始化结束后，展示界面如图 7 所示，界面中主要有 6 个控件：返回键、IP 地址显示文本框 TextView、选择群聊按钮 ChatInLANBtn、输入框 EditText、发送按钮 SendButton。

（1） 点击返回键可返回启动该活动的上一个活动（图中左上角）；

（2） IP 地址显示文本框：显示当前聊天的用户 IP（图中正上方）；

（3） 选择群聊按钮：选中后发送的所有消息都是以广播的形式在局域网内传播（图中右上角）；

（4） 输入框：发送消息的输入位置（图中左下角）；

（5） 发送按钮：点击后将输入框中的消息取出、发送、删除（图中右下角）。

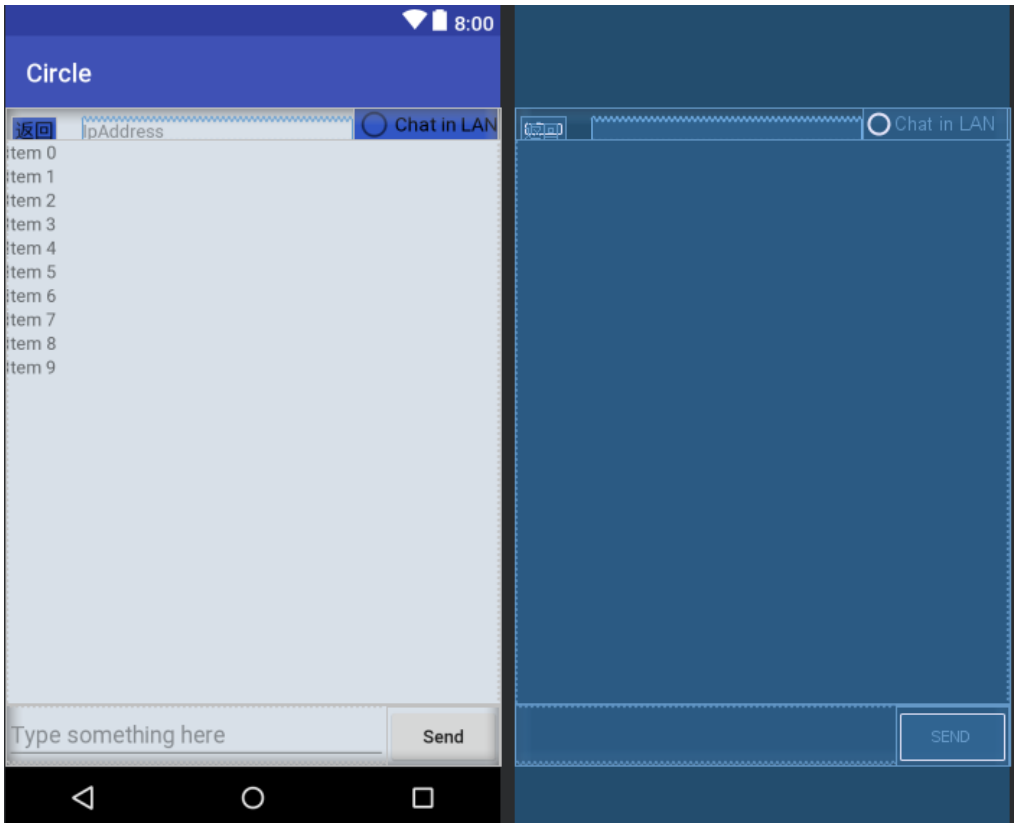

图 7　ChatActivity 界面

三、 程序模块的实现（代码分析）

下面只分析 5 个普通活动类以及 RecordFragment 碎片的代码文件，其他代码文件代码较为简单，此处便不再赘述。

3.1　UI 界面设计

实际开发中，UI 界面设计是首要步骤，设计完成后，再根据各界面对应的 Activity 对各控件进行逻辑操作。UI 界面图可参考 2.2 模块分析。以下是 6 大布局文件，分别是登录布局、主活动布局、历史碎片布局、历史活动布局、聊天活动布局、关于软件的活动布局。

3.1.1 login_activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LoginActivity"
```

```
android:orientation="vertical">

<Space
    android:layout_width="match_parent"
    android:layout_height="80dp" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <Space
        android:layout_width="50dp"
        android:layout_height="50dp"/>

    <TextView
        android:id="@+id/login_account_text_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="账户："
        android:textSize="20sp"/>

    <EditText
        android:id="@+id/login_account_edit_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:textSize="20sp"/>

    <Space
        android:layout_width="50dp"
        android:layout_height="50dp"/>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <Space
        android:layout_width="50dp"
        android:layout_height="50dp"/>
```

静态账户提示文本框

账户输入文本框

```
    <TextView
        android:id="@+id/login_password_text_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="密码："
        android:textSize="20sp"/>

    <EditText
        android:id="@+id/login_password_edit_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPassword"
        android:textSize="20sp"/>

    <Space
        android:layout_width="50dp"
        android:layout_height="50dp"/>

</LinearLayout>

<CheckBox
    android:id="@+id/login_remember_radio"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="50dp"
    android:text="记住密码" />

<Button
    android:id="@+id/login_btn"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="50dp"
    android:layout_gravity="center"
    android:text="登录"
    android:textSize="20sp" />

<Button
    android:id="@+id/login_register_btn"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginTop="5dp"
    android:text="注册"
    android:textSize="20sp"/>
```

静态密码提示文本框

密码输入文本框

记住密码的选择按钮

登录按钮

注册按钮

```
    <Button
        android:id="@+id/login_find_back_btn"
        android:layout_width="150dp"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="5dp"
        android:text="找回密码"
        android:textSize="20sp"/>

</LinearLayout>
```

找回密码按钮

3.1.2 main_activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <FrameLayout
        android:id="@+id/main_frameLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_above="@+id/foundation_title_bottom">
    </FrameLayout>

    <LinearLayout
        android:id="@+id/foundation_title_bottom"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#ffffff"
        android:orientation="horizontal"
        android:layout_alignParentBottom="true"
        android:gravity="bottom">

        <RelativeLayout
            android:id="@+id/title_rc_btn"
            android:onClick="onTabChange"
            android:layout_width="0dp"
```

4 大碎片内容展示位置

4 大碎片选择按钮布局

RecordFrag 碎片

```
    android:layout_height="wrap_content"
    android:layout_weight="1">
    <ImageView
        android:id="@+id/title_rc_img"
        android:layout_width="28dp"
        android:layout_height="28dp"
        android:focusable="false"
        android:layout_centerHorizontal="true"
        android:background="#ffffff"
        android:scaleType="centerInside"
        android:src="@drawable/main_title_record" />

    <TextView
        android:id="@+id/title_rc_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:text="@string/title_record_name"
        android:textColor="#9A9A9A"
        android:textSize="12sp"
        android:layout_below="@+id/title_rc_img"/>

    <TextView
        android:id="@+id/title_rc_unread_num"
        android:layout_width="12dp"
        android:layout_height="12dp"
        android:background="@drawable/aii"
        android:textColor="@android:color/white"
        android:textSize="12sp"
        android:gravity="center"
        android:visibility="gone"
        android:layout_toStartOf="@+id/title_rc_img" />
</RelativeLayout>

<RelativeLayout
    android:id="@+id/title_frd_btn"
    android:onClick="onTabChange"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1">
    <ImageView
        android:id="@+id/title_frd_img"
        android:layout_width="28dp"
        android:layout_height="28dp"
        android:layout_centerHorizontal="true"
```

RecordFrag 图片

显示文本“记录”

未读消息数量显示

FriendFrag 碎片

FriendFrag 图片

```
            android:background="#ffffff"
            android:focusable="false"
            android:scaleType="centerInside"
            android:src="@drawable/main_title_friend" />

        <TextView
            android:id="@+id/title_frd_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerHorizontal="true"
            android:text="@string/title_friend_name"
            android:textColor="#9A9A9A"
            android:textSize="12sp"
            android:layout_below="@+id/title_frd_img"/>

        <TextView
            android:id="@+id/title_frd_unread_num"
            android:layout_width="12dp"
            android:layout_height="12dp"
            android:background="@drawable/aii"
            android:visibility="gone"
            android:textColor="@android:color/white"
            android:textSize="12sp"
            android:gravity="center"
            android:layout_toStartOf="@+id/title_frd_img"/>
    </RelativeLayout>

    <RelativeLayout
        android:id="@+id/title_func_btn"
        android:onClick="onTabChange"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1">
        <ImageView
            android:id="@+id/title_func_img"
            android:layout_width="28dp"
            android:layout_height="28dp"
            android:layout_centerHorizontal="true"
            android:background="#ffffff"
            android:focusable="false"
            android:scaleType="centerInside"
            android:src="@drawable/main_title_function" />

        <TextView
            android:id="@+id/title_func_text"
```

显示文本“朋友”

未读消息数量显示

FunctionFrag 碎片

FunctionFrag 图片

显示文本“功能”

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/title_function_name"
        android:textColor="#9A9A9A"
        android:textSize="12sp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/title_func_img"/>

    <TextView
        android:id="@+id/title_func_unread_num"
        android:layout_width="12dp"
        android:layout_height="12dp"
        android:background="@drawable/aii"
        android:visibility="gone"
        android:textColor="@android:color/white"
        android:gravity="center"
        android:textSize="12sp"
        android:layout_toStartOf="@+id/title_func_img"/>
</RelativeLayout>

<RelativeLayout
    android:id="@+id/title_call_btn"
    android:onClick="onTabChange"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1">
    <ImageView
        android:id="@+id/title_call_img"
        android:layout_width="28dp"
        android:layout_height="28dp"
        android:layout_centerHorizontal="true"
        android:background="#ffffff"
        android:focusable="false"
        android:scaleType="centerInside"
        android:src="@drawable/main_title_call" />

    <TextView
        android:id="@+id/title_call_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/title_call_name"
        android:textColor="#9A9A9A"
        android:textSize="12sp"
        android:layout_centerHorizontal="true"
        android:layout_below="@+id/title_call_img"/>
```

未读消息数量显示

CallFrag 碎片

CallFrag 图片

显示文本“电话”

```
        </RelativeLayout>
    </LinearLayout>

</RelativeLayout>
```

3.1.3 main_fragment_record.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <Button
        android:id="@+id/rc_start_his_activity"
        android:layout_width="74dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="1dp"
        android:layout_marginTop="8dp"
        android:text="Record"
        android:textSize="12sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/rc_ip_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="Ip hasn't list in the LAN"
        android:textSize="12sp"
        app:layout_constraintBottom_toBottomOf="@+id/rc_start_his_activity"
        app:layout_constraintStart_toEndOf="@+id/rc_start_his_activity"
        app:layout_constraintTop_toTopOf="@+id/rc_start_his_activity" />

    <Button
        android:id="@+id/rc_contact_btn"
        android:layout_width="80dp"
        android:layout_height="wrap_content"
        android:text="Contact"
        android:textSize="12sp"
        app:layout_constraintBottom_toBottomOf="@+id/rc_ip_view"
        app:layout_constraintStart_toEndOf="@+id/rc_ip_view"
```

进入记录查询按钮

IP 输入框

联系按钮

```
        app:layout_constraintTop_toTopOf="@+id/rc_ip_view" />

    <Button
        android:id="@+id/rc_start_soft_activity"
        android:layout_width="63dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="1dp"
        android:text="About"
        android:textSize="12sp"
        app:layout_constraintBottom_toBottomOf="@+id/rc_contact_btn"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="@+id/rc_contact_btn" />

    <android.support.v7.widget.RecyclerView
        android:id="@+id/ip_recycler_view"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_weight="1"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/rc_start_his_activity" />

</android.support.constraint.ConstraintLayout>
```

关于软件的活动按钮

局域网 IP 展示框

3.1.4 mainfrag_rcdaty_his.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#0e71d7"
    android:orientation="horizontal"
    tools:context=".mainFragment.rcdAty.HistoryActivity">


    <Button
```

```
    android:id="@+id/his_call_back_btn"
    android:layout_width="35dp"
    android:layout_height="22dp"
    android:layout_marginStart="8dp"
    android:background="#3156c4"
    android:text="返回"
    android:textSize="15sp"
    app:layout_constraintBottom_toBottomOf="@+id/delete_current_data_btn"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="@+id/delete_current_data_btn" />

<Button
    android:id="@+id/delete_current_data_btn"
    android:layout_width="wrap_content"
    android:layout_height="30dp"
    android:layout_marginEnd="8dp"
    android:layout_marginTop="5dp"
    android:background="#3156c4"
    android:text="删除当前 IP 数据"
    app:layout_constraintEnd_toStartOf="@+id/delete_all_data_btn"
    app:layout_constraintTop_toTopOf="parent" />

<Button
    android:id="@+id/delete_all_data_btn"
    android:layout_width="wrap_content"
    android:layout_height="30dp"
    android:layout_marginEnd="8dp"
    android:layout_marginTop="5dp"
    android:background="#3156c4"
    android:text="删除所有 IP 数据"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<RadioGroup
    android:id="@+id/his_ip_group"
    android:layout_width="wrap_content"
    android:layout_height="0dp"
    android:layout_gravity="center_vertical"
    android:orientation="vertical"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/his_call_back_btn"
    app:layout_constraintBottom_toBottomOf="parent">

    <RadioButton
        android:id="@+id/radio_for_size"
```

返回按钮

删除当前 IP 数据按钮

删除所有 IP 数据按钮

IP 选择按钮组

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="255.255.255.255"
            android:textSize="9sp"/>
    </RadioGroup>

    <ListView
        android:id="@+id/uav_info_list_view"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@+id/his_ip_group"
        app:layout_constraintTop_toBottomOf="@+id/delete_current_data_btn">

    </ListView>

</android.support.constraint.ConstraintLayout>
```

聊天记录显示框

3.1.5 mainfrag_rcdaty_chat.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context=".mainFragment.RecordFrag"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#d8e0e8">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">
        <Button
            android:id="@+id/chat_call_back_btn"
            android:layout_width="35dp"
            android:layout_height="20dp"
            android:layout_marginStart="5dp"
            android:layout_marginTop="5dp"
            android:background="#3156c4"
            android:text="返回"
            android:textSize="15sp" />
        <TextView
```

返回按钮

```
        android:id="@+id/chat_ip_view"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="wrap_content"
        android:layout_marginStart="20dp"
        android:layout_marginTop="5dp"
        android:hint="IpAddress"/>

    <RadioButton
        android:id="@+id/chat_LAN_radio"
        android:layout_width="wrap_content"
        android:layout_height="25dp"
        android:layout_marginEnd="2dp"
        android:background="#3156c4"
        android:text="Chat in LAN"
        android:textAllCaps="false"
        android:textSize="15sp" />

</LinearLayout>



<android.support.v7.widget.RecyclerView
    android:id="@+id/msg_recycler_view"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"/>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <EditText
        android:id="@+id/input_text"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:hint="Type something here"
        android:maxLines="2" />
    <Button
        android:id="@+id/send_btn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Send"
        android:textAllCaps="false"/>
```

当前的聊天用户 IP

选择群聊按钮

聊天消息显示框

消息输入框

消息发送按钮

```
    </LinearLayout>

</LinearLayout>
```

3.1.6 mainfrag_rcdaty_soft.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="#0e71d7">

    <Button
        android:id="@+id/soft_call_back_btn"
        android:layout_width="35dp"
        android:layout_height="20dp"
        android:layout_marginStart="5dp"
        android:layout_marginTop="5dp"
        android:background="#3156c4"
        android:text="返回"
        android:textSize="15sp" />

    <TextView
        android:id="@+id/software_info_text"
        android:textSize="20sp"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1">
    </TextView>

</LinearLayout>
```

返回按钮

信息展示框

3.2 LoginActivity.java

```
package com.enzosalvetore.circle;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.ActionBar;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
```

```
import android.widget.EditText;
import android.widget.Toast;

import com.enzosalvetore.circle.mainFragment.RecordFrag;

public class LoginActivity extends AppCompatActivity {

    private static final String TAG = "LoginActivity";
    private EditText accountView,passwordView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.login_activity);
        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null){
            actionBar.hide();
        }
        accountView = (EditText)findViewById(R.id.login_account_edit_view);
        passwordView = (EditText)findViewById(R.id.login_password_edit_view);
        ((Button)findViewById(R.id.login_btn)).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String account = accountView.getText().toString();
                String password = passwordView.getText().toString();
                if(checkAccount(account,password)){
                    Intent intent = new Intent(LoginActivity.this,
                            MainActivity.class);
                    startActivity(intent);
                }else {
                    Toast.makeText(LoginActivity.this,"账号与密码不匹配",
                            Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    @Override
    protected void onDestroy(){
        super.onDestroy();
    }

    private boolean checkAccount(final String account,final String password){
        if(account.equals("123456")&&password.equals("123456")){
```

LoginActivity 入口函数

登录按钮监听函数

启动 MainActivity

检查账号和密码函数

```
            return true;
        }else return false;
    }
}
```

3.3 MainActivity.java

```
package com.enzosalvetore.circle;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentTransaction;
import android.support.v7.app.ActionBar;
import android.view.View;
import android.view.WindowManager;
import android.widget.ImageView;
import android.widget.TextView;

import com.enzosalvetore.circle.mainFragment.CallFrag;
import com.enzosalvetore.circle.mainFragment.FriendFrag;
import com.enzosalvetore.circle.mainFragment.FunctionFrag;
import com.enzosalvetore.circle.mainFragment.RecordFrag;

public class MainActivity extends BasicActivity{

    private static final String TAG = "MainActivity";
    private static int currentTabIndex;
    private static Fragment[] fragments;
    private static ImageView[] imageButtons;
    private static TextView[] unreadNumTexts;
    @Override
    protected void onCreate(Bundle saveInstanceState){
        super.onCreate(saveInstanceState);
        setContentView(R.layout.main_activity);
        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null){
            actionBar.hide();
        }
        initTabView();

        //设置软键盘不自动弹出
      getWindow().setSoftInputMode(  WindowManager.LayoutParams.SOFT_INPUT_STATE_HIDDEN);
    }
```

MainActivity 入口函数

```
@Override
protected void onDestroy(){
    super.onDestroy();
}

private void initTabView() {
    RecordFrag rcFrag = new RecordFrag();
    FriendFrag frdFrag = new FriendFrag();
    FunctionFrag funcFrag = new FunctionFrag();
    CallFrag callFrag = new CallFrag();
    fragments = new Fragment[] {rcFrag,frdFrag,funcFrag,callFrag};
    //三个 title_bottom 小红点（未读消息数量）
    unreadNumTexts = new TextView[3];
    unreadNumTexts[0] = (TextView) findViewById(R.id.title_rc_unread_num);
    unreadNumTexts[1] = (TextView) findViewById(R.id.title_frd_unread_num);
    unreadNumTexts[2] = (TextView) findViewById(R.id.title_func_unread_num);
    //获取 title_bottom 四个 img
    imageButtons = new ImageView[4];
    imageButtons[0] = (ImageView) findViewById(R.id.title_rc_img);
    imageButtons[1] = (ImageView) findViewById(R.id.title_frd_img);
    imageButtons[2] = (ImageView) findViewById(R.id.title_func_img);
    imageButtons[3] = (ImageView) findViewById(R.id.title_call_img);
    imageButtons[0].setSelected(true);
    // 添加显示 fragment
    getSupportFragmentManager().beginTransaction()
            .add(R.id.main_frameLayout, rcFrag)
            .add(R.id.main_frameLayout, frdFrag)
            .add(R.id.main_frameLayout, funcFrag)
            .add(R.id.main_frameLayout, callFrag)
            .hide(frdFrag).hide(funcFrag)
            .hide(callFrag).show(rcFrag).commit();
    currentTabIndex = 0;
}

public void onTabChange(View v){
    int index = Integer.MAX_VALUE;
    switch (v.getId()){
        case R.id.title_rc_btn:
            if(currentTabIndex != 0){
                index = 0;
            }
            break;
        case R.id.title_frd_btn:
            if(currentTabIndex != 1){
                index = 1;
```

初始化四大碎片

主活动的其他控件初始化

四大碎片切换函数

```
                }
                break;
            case R.id.title_func_btn:
                if(currentTabIndex != 2){
                    index = 2;
                }
                break;
            case R.id.title_call_btn:
                if(currentTabIndex != 3){
                    index = 3;
                }
                break;
        }
        if (index != Integer.MAX_VALUE){
            if(!fragments[index].isAdded()){
                getSupportFragmentManager().beginTransaction().add
                        (R.id.main_frameLayout, fragments[currentTabIndex]);
            }
            FragmentTransaction trx = getSupportFragmentManager().beginTransaction();
            if(index != currentTabIndex){
                trx.hide(fragments[currentTabIndex]);
                trx.show(fragments[index]).commit();
                imageButtons[currentTabIndex].setSelected(false);
                imageButtons[index].setSelected(true);
            }
            currentTabIndex = index;
        }
    }
}
```

3.4 RecordFrag.java

```
package com.enzosalvetore.circle.mainFragment;


import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v4.content.LocalBroadcastManager;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
```

```
import android.view.ViewGroup;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.enzosalvetore.circle.R;
import com.enzosalvetore.circle.mainFragment.rcdAty.ChatActivity;
import com.enzosalvetore.circle.mainFragment.rcdAty.HistoryActivity;
import com.enzosalvetore.circle.mainFragment.rcdAty.IpMsg;
import com.enzosalvetore.circle.mainFragment.rcdAty.SoftwareActivity;
import com.enzosalvetore.circle.MyLog;

import org.litepal.LitePal;
import org.litepal.crud.DataSupport;

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.Inet4Address;
import java.net.InetAddress;
import java.net.NetworkInterface;
import java.net.SocketException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Enumeration;
import java.util.List;

public class RecordFrag extends Fragment implements View.OnClickListener{

    //1.布件
    private static RecyclerView ipRecyclerView;
    private static EditText ipEditView;
    //2.变量
    public static final int port = 8086;
    private static String myIp;
    //3.容器
    private List<IpMsg> ipList = new ArrayList<>();
    private IpAdapter adapter;
    //4.其他
    public static LocalBroadcastManager localBroadcastManager;
    public static SimpleDateFormat formatter =
            new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    private static final String TAG = "RecordFrag";

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
```

```
                        Bundle savedInstanceState){
    View view = inflater.inflate(R.layout.main_fragment_record,
            container,false);
    MyLog.log(TAG,"in RecordFrag onCreateView");
    localBroadcastManager = LocalBroadcastManager.getInstance(getContext());
    new RecordFrag.Receiver().execute(port);
    initIps();
    ipRecyclerView = (RecyclerView)view.findViewById(R.id.ip_recycler_view);
    LinearLayoutManager layoutManager = new LinearLayoutManager(getContext());
    ipRecyclerView.setLayoutManager(layoutManager);
    adapter = new IpAdapter(ipList);
    ipRecyclerView.setAdapter(adapter);

    ipEditView = (EditText)view.findViewById(R.id.rc_ip_view);
    view.findViewById(R.id.rc_contact_btn).setOnClickListener(this);
    view.findViewById(R.id.rc_start_his_activity).setOnClickListener(this);
    view.findViewById(R.id.rc_start_soft_activity).setOnClickListener(this);
    LitePal.getDatabase();

    new Thread(new Runnable() {
        @Override
        public void run() {
            ChatActivity.sendByUDP("255.255.255.255","我上线了");
        }
    }).start();
    return view;
}

@Override
public void onDestroy(){
    new Thread(new Runnable() {
        @Override
        public void run() {
            ChatActivity.sendByUDP("255.255.255.255","我下线了");
        }
    }).start();
    super.onDestroy();
}

@Override
public void onClick(View v){
    switch (v.getId()){
        case R.id.rc_start_his_activity:
            MyLog.log(TAG,"click the start_history_activity");
            startActivity(new Intent(getContext(),
```

RecordFrag 入口函数

进入主活动后会马上执行该函数，通知局域网内所有在线用户：本人已经上线

销毁活动会马上执行该函数，通知局域网内所有在线用户：本人已经下线

按钮点击后调用该函数

启动 HistoryActivity

```
                    HistoryActivity.class));
            break;
        case R.id.rc_start_soft_activity:
            MyLog.log(TAG,"click the start_soft_activity");
            startActivity(new Intent(getContext(),
                    SoftwareActivity.class));
            break;
        case R.id.rc_contact_btn:
            MyLog.log(TAG,"click the rc_contact_btn");
            Intent intent = new Intent(v.getContext(), ChatActivity.class);
            String ipAddr = ipEditView.getText().toString();
            if ("".equals(ipAddr)) Toast.makeText(getContext(),
                    "发送的 ip 尚未填写",Toast.LENGTH_SHORT).show();
            else {
                intent.putExtra("ipAddress",ipAddr);
                startActivity(intent);
            }
    }
}

/******初始化 myIp 和 listView*****/
private void initIps(){
    try {
        for(Enumeration<NetworkInterface>en=NetworkInterface.getNetworkInterfaces();
             en.hasMoreElements();) {
            NetworkInterface intf = en.nextElement();
            for (Enumeration<InetAddress> enumIpAddr = intf.getInetAddresses();
                 enumIpAddr.hasMoreElements();) {
                InetAddress inetAddress = enumIpAddr.nextElement();
                if (!inetAddress.isLoopbackAddress() && (inetAddress instanceof
Inet4Address)) {
                    myIp = inetAddress.getHostAddress();
                }
            }
        }
    }
    catch (SocketException ex){
        ex.printStackTrace();
    }
    List<IpMsg>MsgList = DataSupport.select("iPAddress").
            order("iPAddress").find(IpMsg.class);
    if (MsgList.size()!= 0){
        String ipInList = MsgList.get(0).getIpAddress();
        String ipCandidate;
        for (IpMsg ipMsg:MsgList){
```

启动关于软件的活动

启动与输入 IP 的用户进行私聊的活动

获取本机 IP

局域网内的用户 IP 列表

```
                ipCandidate = ipMsg.getIpAddress();
                if (!ipInList.equals(ipCandidate)){
                    ipList.add(ipMsg);
                    ipInList = ipCandidate;
                }
            }
        }else{
            IpMsg ipMsg = new IpMsg("255.255.255");
            MsgList.add(ipMsg);
        }
    }

    /******用户展示 listView*****/
    static class ViewHolder extends RecyclerView.ViewHolder{
        TextView ipAddress;
        View btn;
        public ViewHolder(View view){
            super(view);
            btn = view;
            ipAddress = (TextView)view.findViewById(R.id.ip_addr_item);
        }
    }

    public class IpAdapter extends RecyclerView.Adapter<RecordFrag.ViewHolder> {

        private List<IpMsg> mIpList;

        public IpAdapter(List<IpMsg> ipList) {
            mIpList = ipList;
        }

        @Override
        public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
            View view = LayoutInflater.from(parent.getContext()).
                    inflate(R.layout.mainfrag_rcdaty_item_ip, parent, false);
            final ViewHolder holder = new ViewHolder(view);
            holder.btn.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    String                              ipAddr                              =
mIpList.get(holder.getAdapterPosition()).getIpAddress();
                    Intent intent = new Intent(v.getContext(), ChatActivity.class);
                    intent.putExtra("ipAddress", ipAddr);
                    startActivity(intent);
                }
```

```
        });
        return holder;
    }

    @Override
    public void onBindViewHolder(ViewHolder holder, int position) {
        IpMsg ipMsg = mIpList.get(position);
        holder.ipAddress.setText(ipMsg.getIpAddress());
    }

    @Override
    public int getItemCount() {
        return mIpList.size();
    }
}

/*****数据库保存*******/
public static void saveData(String ipAddr,String msgCtn,int type){
    //保存数据
    MyLog.log(TAG,"in saveData");
    IpMsg ipMsgData = new IpMsg(msgCtn,type);
    ipMsgData.setIpAddress(ipAddr);
    ipMsgData.setTime(formatter.format(new java.sql.Date
            (System.currentTimeMillis())));
    MyLog.log(TAG,"保存 ipMsgData 数据： "+"Time:"+ipMsgData.getTime()+
            ",ip:"+ipMsgData.getIpAddress()+ ":content:"+ipMsgData.getContent()
            +",Type:"+ipMsgData.getType());
    ipMsgData.save();
}

/*****监听端口 ***********/
private class Receiver extends AsyncTask<Integer,String,Boolean> {

    @Override
    protected Boolean doInBackground(Integer...receive){
        try {
            MyLog.log(TAG,"in doInBackground:监听端口:"+receive[0]);
            //1.创建一个 2048 字节的数组，用于接收数据
            byte[]buf = new byte[2048];
            //2.定义一个 DatagramSocket 对象，监听端口为 receive[0]
            DatagramSocket ds = new DatagramSocket(receive[0]);
            //3.定义一个 DatagramPacket 对象，用于接收数据
            DatagramPacket dp = new DatagramPacket(buf,buf.length);
            while(true) {
                //4.等待接受数据，没有数据则阻塞
```

```
                ds.receive(dp);
                String recvIp = dp.getAddress().getHostAddress();
                MyLog.log(TAG,"myIp:"+myIp);
                MyLog.log(TAG,"recvIp:"+recvIp);
                if (!recvIp.equals(myIp)){
                    String str = new String(dp.getData(),0,dp.getLength());
                    MyLog.log(TAG,"in doInBackground:recContent: "+str);
                    publishProgress(dp.getAddress().getHostAddress(),str);
                }
            }
        }catch(Exception e) {
            e.printStackTrace();
        }
        return false;
    }

    @Override
    public void onProgressUpdate(String...str){
        MyLog.log(TAG,"in onProgressUpdate");

        //判断是否为界面已有 ip
        boolean isNew = true;
        for (IpMsg ipMsg:ipList){
            MyLog.log(TAG,ipMsg.getIpAddress());
            if (ipMsg.getIpAddress().equals(str[0])){
                isNew = false;
                break;
            }
        }
        if (isNew){
            //新来一个用户更新用户展示 ip 界面
            MyLog.log(TAG,"isNew");
            IpMsg ip = new IpMsg(str[0]);
            ipList.add(ip);
        }
        adapter.notifyItemInserted(ipList.size());
        ipRecyclerView.scrollToPosition(ipList.size()-1);
        Intent intent = new Intent("com.enzosalvetore.circle.LOCAL_BROADCAST");
        //无论是否为新，都保存数据进数据库,并通知 ChatActivity
        String chatAtyIpAddr = "";
        if (ChatActivity.ipAddr!=null){
            chatAtyIpAddr = ChatActivity.ipAddr;
        }
        if (!chatAtyIpAddr.equals("")){
            //用户已打开聊天活动
```

```
                if (!chatAtyIpAddr.equals("255.255.255.255")){
                    //此时为私聊，且接收到的消息不会是来自自己，publish 之前已经过滤
                    MyLog.log(TAG,"私聊,收到："+str[1]);
                    //接收到的消息为已打开聊天活动的消息，则通知当前活动
                    if (chatAtyIpAddr.equals(str[0]))intent.putExtra("recv_msg",str[1]);
                    saveData(str[0],str[1], IpMsg.TYPE_RECEIVE);
                }else {
                    //此时为群聊，将发送者 ip 保存到 content 中
                    MyLog.log(TAG,"群聊");
                    intent.putExtra("recv_msg",str[0]+":"+str[1]);
                    saveData("255.255.255.255",str[0]+":"+str[1],IpMsg.TYPE_RECEIVE);
                }
            }else {
                //此时用户还没点开聊天活动，因此保存到私聊记录中
                MyLog.log(TAG,"私聊_用户没有点开群聊");
                saveData(str[0],str[1],IpMsg.TYPE_RECEIVE);
            }
            localBroadcastManager.sendBroadcast(intent);
            MyLog.log(TAG,"out onProgressUpdate");
        }
    }
}
```

3.5 HistoryActivity.java

```
package com.enzosalvetore.circle.mainFragment.rcdAty;
import android.os.Bundle;
import android.support.v7.app.ActionBar;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Toast;

import com.enzosalvetore.circle.R;
import com.enzosalvetore.circle.BasicActivity;
import com.enzosalvetore.circle.MyLog;

import org.litepal.crud.DataSupport;

import java.util.ArrayList;
import java.util.List;
```

```
import java.util.Vector;

/**
 * Created by Enzo on 2018/5/4.
 */

public class HistoryActivity extends BasicActivity {

    private static List<IpMsg> MsgList;
    private static ListView listView;
    private static ArrayAdapter<String> adapter;
    private static Vector<String> vecString;
    private static RadioGroup ipRadioGroup;
    private static String ipAddress;
    private static final String TAG = "HistoryActivity";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.mainfrag_rcdaty_his);
        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null) {
            actionBar.hide();
        }
        ipRadioGroup = (RadioGroup) findViewById(R.id.his_ip_group);
        ((RadioButton)findViewById(R.id.radio_for_size)).setVisibility(View.GONE);
        listView = (ListView) findViewById(R.id.uav_info_list_view);
        MsgList = new ArrayList<>();
        vecString = new Vector<>();
        adapter = new ArrayAdapter<String>(HistoryActivity.this,
                android.R.layout.simple_list_item_1, vecString);
        listView.setAdapter(adapter);

        //初始化 RadioGroup
        initRadioG();
        //返回键
        ((Button)findViewById(R.id.his_call_back_btn)).setOnClickListener
                (new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        onBackPressed();
                    }
                });
        //删除当前 ip 数据键
        ((Button)findViewById(R.id.delete_current_data_btn)).setOnClickListener
```

HistoryActivity 入口函数

```
            (new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    int id = ipRadioGroup.getCheckedRadioButtonId();
                    if (id == -1) {
                        MyLog.log(TAG,"尚未选择 Ip");
                        Toast.makeText(HistoryActivity.this,"请选择 Ip",
                                Toast.LENGTH_SHORT).show();
                    }else {
                        final String ipAddr = ipAddress;
                        if (!ipAddr.equals("")){
                            DataSupport.deleteAll
                                    (IpMsg.class,"iPAddress = ?", ipAddr);
                            vecString.clear();
                            adapter.notifyDataSetChanged();
                            ipAddress="";
                        }else {
                            Toast.makeText(HistoryActivity.this,"已删除",
                                    Toast.LENGTH_SHORT).show();
                        }
                    }
                }
            });
    //删除所有 ip 数据键
    ((Button)findViewById(R.id.delete_all_data_btn)).setOnClickListener
            (new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    DataSupport.deleteAll(IpMsg.class);
                    vecString.clear();
                    adapter.notifyDataSetChanged();
                    ipRadioGroup.clearAnimation();
                    ipAddress = "";
                    Toast.makeText(HistoryActivity.this,"已删除",
                            Toast.LENGTH_SHORT).show();
                }
            });
}

private void initRadioG(){
    MsgList = DataSupport.select("ipAddress")
            .order("ipAddress")
            .find(IpMsg.class);
    if (MsgList.isEmpty()){
        MyLog.log(TAG,"The list is empty when initialize the RadioGroup");
```

初始化所有 RadioButton

```
    }else {
        String ipInVec = MsgList.get(0).getIpAddress();
        String ipCandidate;
        Vector<String>ipVec = new Vector<>();
        ipVec.add(ipInVec);
        for (IpMsg ipMsg:MsgList){
            MyLog.log(TAG,ipMsg.getIpAddress());
            ipCandidate = ipMsg.getIpAddress();
            if (!ipInVec.equals(ipCandidate)){
                ipVec.add(ipCandidate);
                ipInVec = ipCandidate;
            }
        }

        for (final String ipAddr : ipVec){
            final RadioButton radioButton = new RadioButton(this);
            final String ipAddress = ipAddr;
            radioButton.setText(ipAddress);
            radioButton.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    HistoryActivity.ipAddress = ipAddress;
                    getData(ipAddress);
                }
            });
            ipRadioGroup.addView(radioButton);
        }
    }
}

private void getData(String ipAddress) {
    MyLog.log(TAG, "in getData,ipAddress:");
    if (!ipAddress.isEmpty()) {
        MyLog.log(TAG, "getData,ipAddress:" + ipAddress);
        //查询数据
        MsgList.clear();
        MsgList = DataSupport.where("iPAddress = ?",ipAddress)
                .select("time","content","type")
                .order("time")
                .find(IpMsg.class);
        //更新 ListView
        vecString.clear();
        if (!MsgList.isEmpty()){
            for (IpMsg msg : MsgList) {
                String ctn;
```

获取数据库内容

```
                    if (msg.getType()==IpMsg.TYPE_RECEIVE)
                        ctn = msg.getTime()+",Recv:"+""+msg.getContent();
                    else
                        ctn = msg.getTime()+",Send:"+""+msg.getContent();
                    vecString.add(ctn);
                }
            }else {
                MyLog.log(TAG, "数据库为空");
                Toast.makeText(HistoryActivity.this,
                        "数据库为空", Toast.LENGTH_SHORT).show();
            }
            adapter.notifyDataSetChanged();
        } else {
            MyLog.log(TAG, "ip 地址为空");
            Toast.makeText(HistoryActivity.this,
                    "ip 地址为空", Toast.LENGTH_SHORT).show();
        }
        MyLog.log(TAG, "out getData");
    }
}
```

3.6 ChatActivity.java

```
package com.enzosalvetore.circle.mainFragment.rcdAty;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.support.v7.app.ActionBar;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.TextView;
import android.widget.Toast;

import com.enzosalvetore.circle.R;
import com.enzosalvetore.circle.BasicActivity;
import com.enzosalvetore.circle.mainFragment.RecordFrag;
import com.enzosalvetore.circle.MyLog;
```

```
import org.litepal.crud.DataSupport;

import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;
import java.util.List;

public class ChatActivity extends BasicActivity {
    public List<IpMsg> msgList;
    private static EditText inputText;
    private static RecyclerView msgRecyclerView;
    private static TextView ipView;
    private static RadioButton chatInLANrBtn;
    public static MsgAdapter adapter;
    public static String ipAddr;
    private static String ipAddrForLAN;
    private static boolean isRadioCkd;
    private static IntentFilter intentFilter;
    private static LocalReceiver localReceiver;
    public String inputCtn;
    private static final String TAG = "ChatActivity";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.mainfrag_rcdaty_chat);
        ActionBar actionBar = getSupportActionBar();
        if (actionBar != null){
            actionBar.hide();
        }
        /***初始化***/
        ipView = (TextView)findViewById(R.id.chat_ip_view);
        inputText = (EditText)findViewById(R.id.input_text);
        chatInLANrBtn = (RadioButton)findViewById(R.id.chat_LAN_radio);
        msgRecyclerView = (RecyclerView)findViewById(R.id.msg_recycler_view);

        isRadioCkd = false;
        msgList = new ArrayList<>();
        initMsgs();      //查询数据库，初始化 msgList
        LinearLayoutManager layoutManager = new LinearLayoutManager(this);
        msgRecyclerView.setLayoutManager(layoutManager);
        adapter = new MsgAdapter(msgList);
        msgRecyclerView.setAdapter(adapter);
```

ChatActivity 入口函数

```
        msgRecyclerView.scrollToPosition(msgList.size()-1); //定位到记录的最后一行
        intentFilter = new IntentFilter();
        localReceiver = new LocalReceiver();
        intentFilter.addAction("com.enzosalvetore.circle.LOCAL_BROADCAST");
        RecordFrag.localBroadcastManager.registerReceiver
                (localReceiver,intentFilter);

        ((Button)findViewById(R.id.chat_call_back_btn)).setOnClickListener
                (new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        onBackPressed();
                    }
                });

        chatInLANrBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (isRadioCkd){
                    MyLog.log(TAG,"点击 Radio 前已选中");
                    chatInLANrBtn.setChecked(false);
                    ipAddr = ipAddrForLAN;  //还原原通信 ip
                }else {
                    MyLog.log(TAG,"点击 Radio 前没有选中");
                    ipAddrForLAN = ipAddr;   //保留当前通信 ip
                    chatInLANrBtn.setChecked(true);
                    ipAddr = "255.255.255.255";
                }
                isRadioCkd = !isRadioCkd;
            }
        });

        ((Button)findViewById(R.id.send_btn)).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                final String content = inputText.getText().toString();
                if(!"".equals(content)){
                    final IpMsg msg = new IpMsg(content,IpMsg.TYPE_SENT);
                    msgList.add(msg);
                    //有新消息时刷新 RecyclerView 的显示
                    adapter.notifyItemInserted(msgList.size()-1);
                    //将 RecyclerView 定位到最后一行
                    msgRecyclerView.scrollToPosition(msgList.size()-1);
                    inputCtn = inputText.getText().toString();
```

返回键

群聊选择键

消息发送按钮

```
                new Thread(new Runnable() {
                    @Override
                    public void run() {
                        sendByUDP(ipAddr,inputCtn);
                        RecordFrag.saveData(ipView.getText().toString(),
                                content,IpMsg.TYPE_SENT);
                    }
                }).start();
                inputText.setText("");
            }
        }
    });
}

@Override
protected void onDestroy(){
    ipAddr = "";
    RecordFrag.localBroadcastManager.unregisterReceiver(localReceiver);
    super.onDestroy();
}

private void getData(String ipAddress) {
    MyLog.log(TAG, "in getData");
    if (!ipAddress.isEmpty()) {
        MyLog.log(TAG, "getData,ipAddress:" + ipAddress);
        msgList.clear();
        msgList = DataSupport.where("iPAddress = ?",ipAddress)
                .select("time","content","type")
                .order("time")
                .find(IpMsg.class);
        if (msgList.isEmpty()){
            MyLog.log(TAG, "数据库为空");
            Toast.makeText(ChatActivity.this,
                    "数据库为空", Toast.LENGTH_SHORT).show();
        }else {
            MyLog.log(TAG,"查询数据库结果：");
            for (IpMsg ipMsg:msgList){
                MyLog.log(TAG,ipMsg.getTime()+"ip:"+
                        ipMsg.getIpAddress()+":"+ipMsg.getContent());
            }
        }
    } else {
        MyLog.log(TAG, "ip 地址为空");
        Toast.makeText(ChatActivity.this,
                "ip 地址为空", Toast.LENGTH_SHORT).show();
```

获取指定 IP 聊天记录

```
        }
        MyLog.log(TAG, "out getData");
    }

    private void initMsgs(){
        Intent intent = getIntent();
        ipAddr = intent.getStringExtra("ipAddress");
        if ("".equals(ipAddr)) onBackPressed();
        ipView.setText(ipAddr);
        getData(ipAddr);   //初始化 List
    }
    public static void sendByUDP(String ipAddr,String inputCtn){
        if (!inputCtn.equals("")){
            try{
                DatagramSocket ds = new DatagramSocket();
                byte buf[] = inputCtn.getBytes();
                int size = buf.length;
                MyLog.log(TAG,ipAddr+","+RecordFrag.port);
                MyLog.log(TAG,"Send>>content:"+inputCtn+"size:"+size);
                DatagramPacket dp = new DatagramPacket(buf,size,
                        InetAddress.getByName(ipAddr),RecordFrag.port);
                ds.send(dp);
                MyLog.log(TAG,"已发送");
                ds.close();
            }catch (IOException e){
                e.printStackTrace();
            }
        }
    }

    class LocalReceiver extends BroadcastReceiver{
        @Override
        public void onReceive(Context context,Intent intent){
            final String ctn = intent.getStringExtra("recv_msg");
            if (!ctn.equals("")){
                final IpMsg ipMsg = new IpMsg(ctn,IpMsg.TYPE_RECEIVE);
                MyLog.log(TAG,"收到广播信息:"+ipMsg.getContent());
                msgList.add(ipMsg);
                adapter.notifyItemInserted(msgList.size()-1);
                msgRecyclerView.scrollToPosition(msgList.size()-1);
            }
        }
    }
}
```

初始化聊天界面消息内容

采用 UDP 发送消息

本地广播接收器，用于接收来自 RecordFrag 发送来的消息。

四、 成果展示

定义：三部手机，左侧手机为用户 1，中间手机为用户 2，右侧手机为用户 3（如图 9）。
用户 2 点击登录按钮后会向局域网内的所有 IP 用户发送“我上线了”的消息，此时其他用户收到后会更新界面，比如：用户 3 收到后更新界面如图 10 所示，用户 3 查看消息时会发现收到来自用户 2 的“我上线了”的消息（如图 11）。

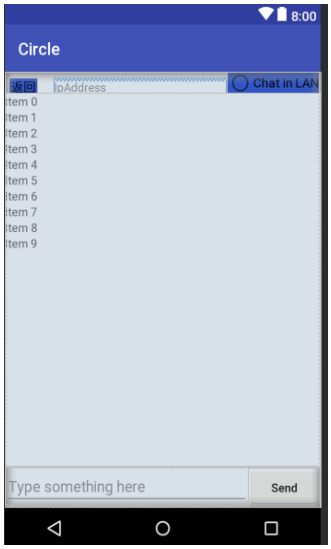

图 8

图 9 登录界面

图 10

图 11

图 12

图 13

图 14

图 15

群聊功能

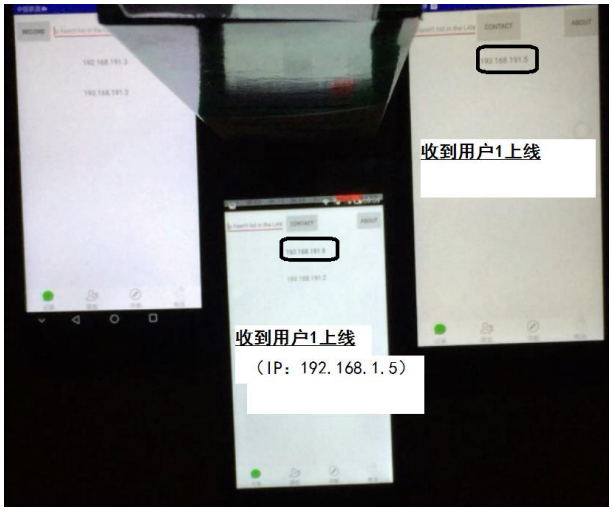

图 16 用户 1 上线

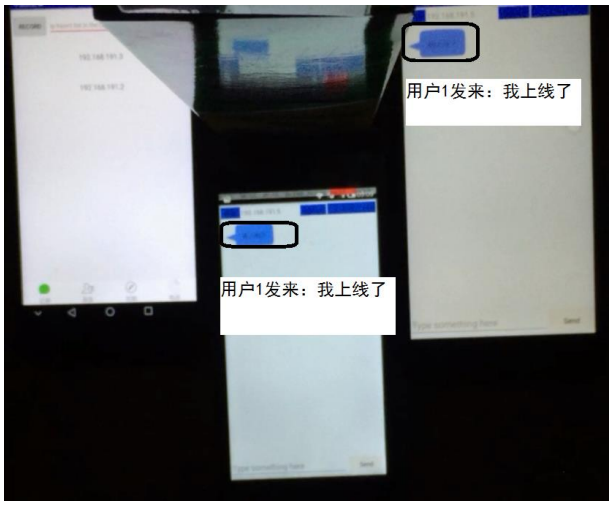

图 17 其他用户收到上线通知

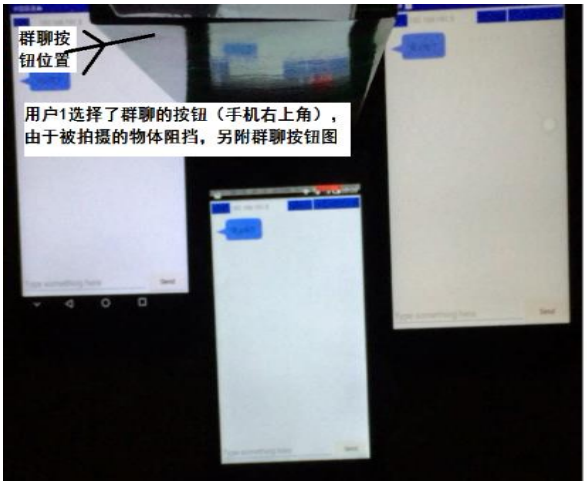

图 18 用户 1 选择群聊按钮

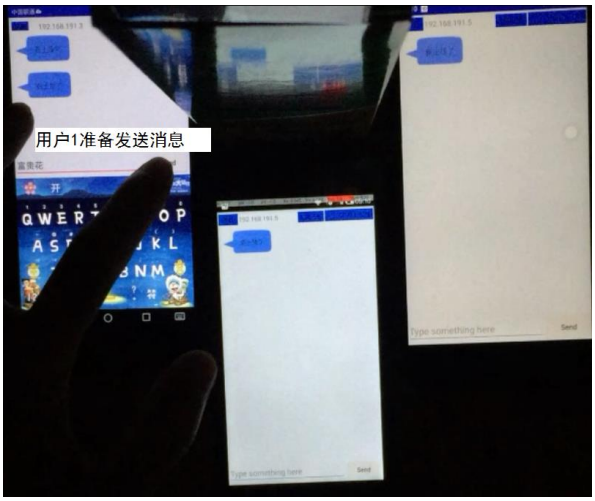

图 19 用户 1 发送消息

图 20 用户 2、3 同时收到消息

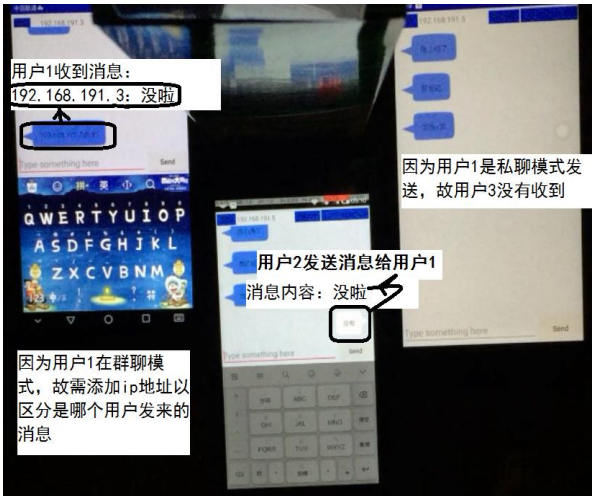

图 21 用户 2 发送私聊消息给用户 1

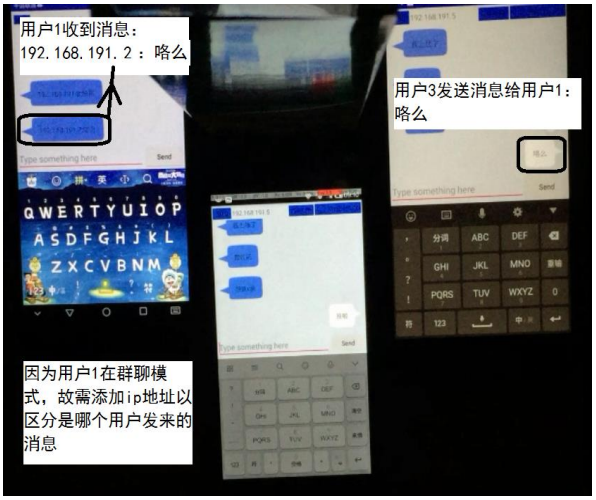

图 22 用户 3 发送私聊消息给用户 1

此外还有聊天数据库查看、聊天内容删除等附加功能，此处便不再赘述，具体可参考视频文件。

五、 进一步研究

本设计实现功能较为简单，在进行深入思考后，笔者在此提出几个研究方向：

缺陷：

1. 本设计通信过程限于局域网， 无法在 Internet 上进行通信。
2. 本设计通信方法采用 UDP 进行通信，当连入 Internet 后，由于网络状态并不稳定，采用该协议无法提供可靠传输。
3. 群聊功能仅限于同一局域网，无法在 Internet 上真正实现群聊。

针对以上缺陷提出的研究方向：

1. 公有 IP 无法在互联网上进行通信，应在云服务器上架构服务端，在客户端以公有 IP 访问服务端的私有 IP，即可实现消息转发，从而间接实现“P2P”通信（两客户端通信）。为防止用户过多，更进一步的研究方向：在服务端实现虚拟机，在虚拟机上为用户提供服务，从而实现服务器的便捷扩展，支持更多的用户容量，实现负载均衡。
2. 可采用 TCP 提供 Internet 上的可靠通信，当软件用户量庞大时，可考虑使用 TCP 协议的扩展功能，调整参数，实现更高质量的通信，若还无法满足需求，可考虑借鉴 TCP 协议，自定义新的协议规定，使协议更加切合实际网络，进一步提高通信质量。
3. 可借鉴 WeChat，实现 Internet 上的群聊，但此处笔者有另一创新的想法：实现金字塔传信，金字塔关系在实际中比较常见，比如公司的层次结构、学校人员的层次结构、军队的层次结构，金子塔传信指的是：最高层向最底层人员发送消息时，一键可达，

而非通过中层的转发，由于篇幅限制，该传信机制的优缺点以及必要性，笔者便不再在此处接着讨论，有意者可联系笔者，笔者愿分享这一想法。

六、 设计心得

本课程设计提供给笔者实现 idea 的机会，使笔者对软件开发有了初步的认识、加深了笔者对 Android 开发的理解、进一步规范了笔者的编程习惯、使笔者深刻认识到软件架构的重要性。

希望本设计能给后来者提供一条通往 Android 开发的路。

最后，很感谢徐兴教授对本次课设的动员，收益良多。

附件 1 Application 的配置（.app/build.gradle）

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 27
    defaultConfig {
        applicationId "com.enzosalvetore.circle"
        minSdkVersion 17
        targetSdkVersion 27
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
        externalNativeBuild {
            cmake {
                cppFlags "-std=c++11 -frtti -fexceptions"
            }
        }
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles   getDefaultProguardFile('proguard-android.txt'),   'proguard-
rules.pro'
        }
    }
    externalNativeBuild {
        cmake {
            path "CMakeLists.txt"
        }
```

```
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:27.1.1'
    implementation 'com.android.support.constraint:constraint-layout:1.1.0'
    implementation 'com.android.support:recyclerview-v7:27.1.1'
    implementation 'org.litepal.android:core:1.6.1'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
}
```

附件 2　项目目录结构

```
C:.
|   AndroidManifest.xml
|
├──assets
|       litepal.xml
|
├──cpp
|       native-lib.cpp
|
├──java
|   └──com
|       └──enzosalvetore
|           └──circle
|               |   BasicActivity.java
|               |   LoginActivity.java
|               |   MainActivity.java
|               |   MyLog.java
|               |
|               └──mainFragment
|                   |   CallFrag.java
|                   |   FriendFrag.java
|                   |   FunctionFrag.java
|                   |   RecordFrag.java
|                   |
|                   ├──callAty
|                   ├──frdAty
```

```
|                   ├──funcAty
|                   └──rcdAty
|                          ChatActivity.java
|                          HistoryActivity.java
|                          IpMsg.java
|                          MsgAdapter.java
|                          SoftwareActivity.java
|
└──res
    ├──drawable
    |       ic_launcher_background.xml
    |       main_title_call.xml
    |       main_title_friend.xml
    |       main_title_function.xml
    |       main_title_record.xml
    |
    ├──drawable-hdpi
    |       aii.png
    |       foundation_title_call_normal.png
    |       foundation_title_call_pressed.png
    |       foundation_title_frd_normal.png
    |       foundation_title_frd_pressed.png
    |       foundation_title_func_normal.png
    |       foundation_title_func_pressed.png
    |       foundation_title_rc_normal.png
    |       foundation_title_rc_pressed.png
    |
    ├──drawable-ldpi
    |       message_left.9.png
    |       message_right.9.png
    |
    ├──drawable-v24
    |       ic_launcher_foreground.xml
    |
    ├──layout
    |       foundation_activity_login.xml
    |       main_activity.xml
    |       main_fragment_call.xml
    |       main_fragment_friend.xml
    |       main_fragment_function.xml
    |       main_fragment_record.xml
    |       other_rcdaty_chat.xml
    |       other_rcdaty_his.xml
    |       other_rcdaty_item_ip.xml
    |       other_rcdaty_item_msg.xml
```

```
|       other_rcdaty_soft.xml
|
├──mipmap-anydpi-v26
|       ic_launcher.xml
|       ic_launcher_round.xml
|
├──mipmap-hdpi
|       ic_launcher.png
|       ic_launcher_round.png
|
├──mipmap-mdpi
|       ic_launcher.png
|       ic_launcher_round.png
|
├──mipmap-xhdpi
|       ic_launcher.png
|       ic_launcher_round.png
|
├──mipmap-xxhdpi
|       ic_launcher.png
|       ic_launcher_round.png
|
├──mipmap-xxxhdpi
|       ic_launcher.png
|       ic_launcher_round.png
|
└──values
        colors.xml
        strings.xml
        styles.xml
```

附件 3　AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.enzosalvetore.circle">

    <!-- 允许应用程序改变网络状态 -->
    <uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />

    <!-- 允许应用程序改变 WIFI 连接状态 -->
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
```

```
<!-- 允许应用程序访问有关的网络信息 -->
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<!-- 允许应用程序访问 WIFI 网卡的网络信息 -->
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />

<!-- 允许应用程序完全使用网络 -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<application
    android:name="org.litepal.LitePalApplication"
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">

    <activity android:name=".LoginActivity"
        android:screenOrientation="portrait"
        android:launchMode="singleTask">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity android:name=".MainActivity"
        android:screenOrientation="portrait"
        android:launchMode="singleTask">
    </activity>

    <activity android:name=".mainFragment.rcdAty.ChatActivity"
        android:screenOrientation="portrait">

    </activity>
    <activity android:name=".mainFragment.rcdAty.HistoryActivity"
        android:screenOrientation="portrait">

    </activity>
    <activity android:name=".mainFragment.rcdAty.SoftwareActivity"
        android:screenOrientation="portrait">

    </activity>
```

```
    </application>

</manifest>
```

参考文献

[1].郭霖.第一行代码 Android 第 2 版.中国工信出版集团.人民邮电出版社.2016.12.