Course Number: ECE655

Term: Fall 2016

Name: Zhang, Xiyue

UW ID: 20601564

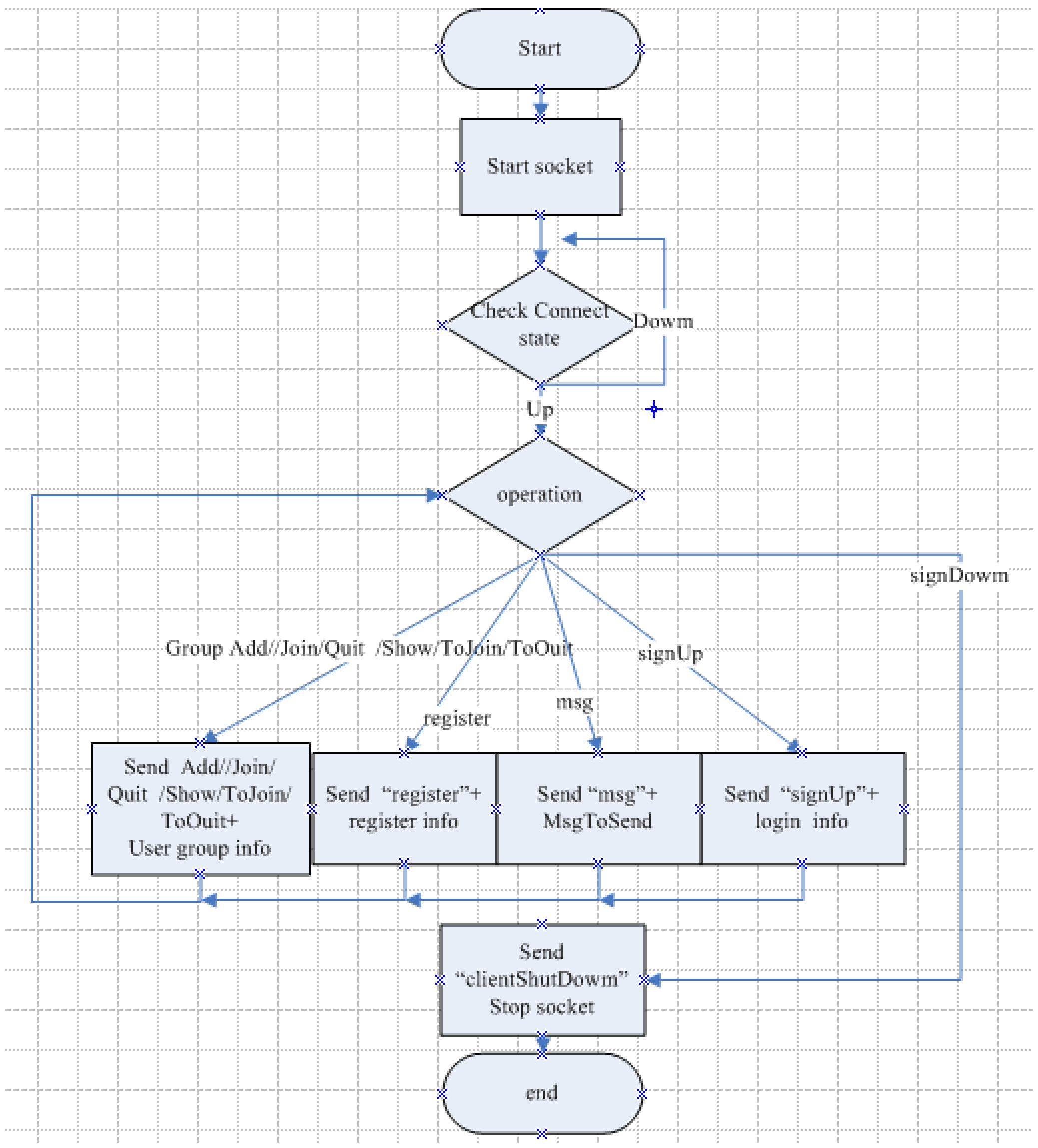
Email: x562zhan@uwaterloo.ca

Abstract

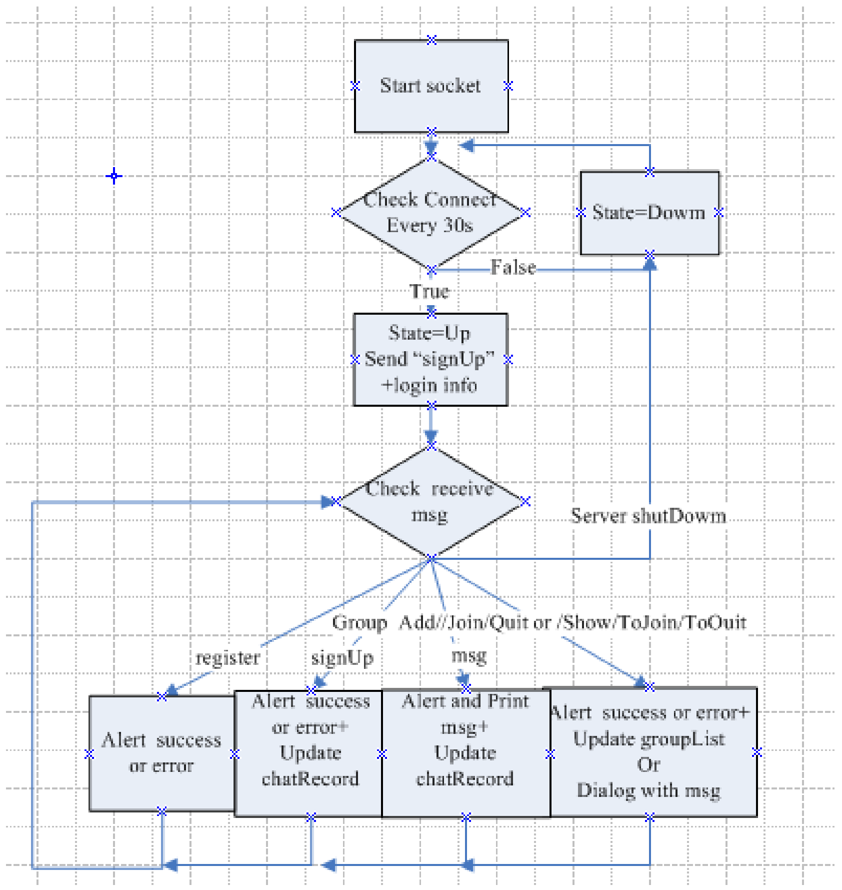
The project is instant messaging (IM) app. The whole system is composed of one server and several clients. The client can connect with server to send message and receive message. The server needs to deal with the request from client and sent the response to client. Also the server connects with database to manage chat message. The IM application is developed under Android SDK API 15(Android 4.0.3) and JDK 1.7.0.

Control Flow Graph

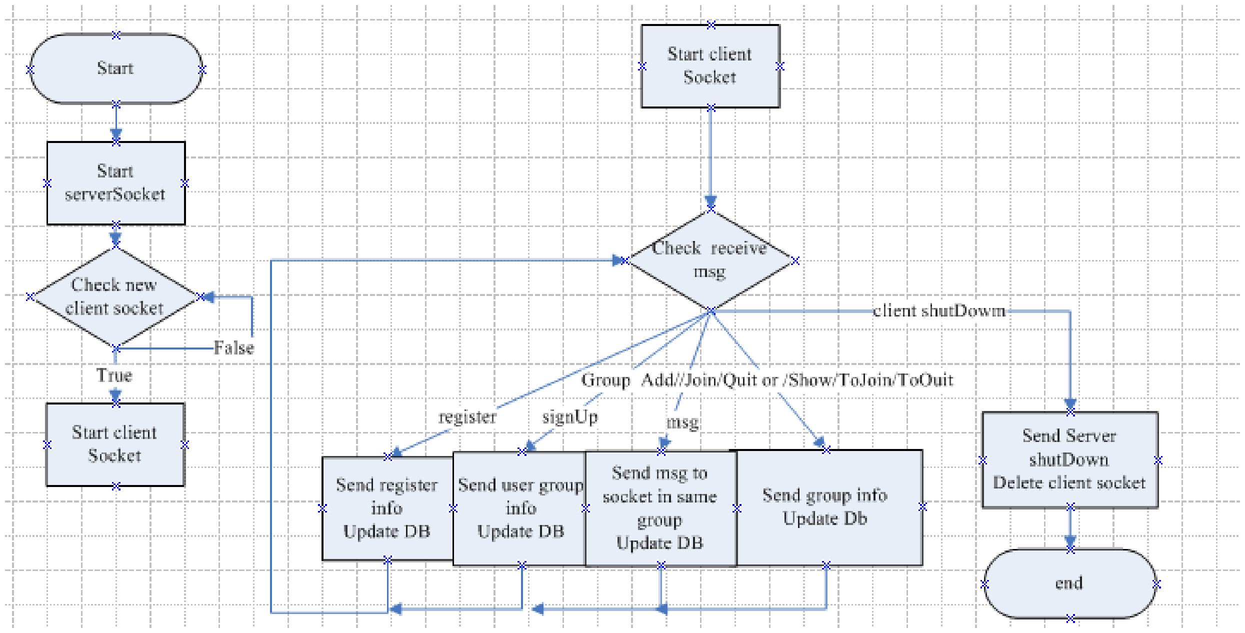
Client Sender


Start
Start socket
Check Connect state
Dowm
Up
operation
signDowm
Group Add//Join/Quit /Show/ToJoin/ToOuit
signUp
msg
register
Send Add//Join/ Quit /Show/ToJoin/ ToOuit+ User group info
Send "register"+ register info
Send "msg"+ MsgToSend
Send "signUp"+ login info
Send "clientShutDowm" Stop socket
end

Client Receiver


Start socket
Check Connect Every 30s
State=Dowm
False
True
State=Up Send "signUp" +login info
Check receive msg
Server shutDowm
Group Add//Join/Quit or /Show/ToJoin/ToOuit
register
signUp
msg
Alert success or error
Alert success or error+ Update chatRecord
Alert and Print msg+ Update chatRecord
Alert success or error+ Update groupList Or Dialog with msg

Server Socket Manager


Start
Start serverSocket
Check new client socket
False
True
Start client Socket
Start client Socket
Check receive msg
client shutDowm
Group Add//Join/Quit or /Show/ToJoin/ToOuit
register
signUp
msg
Send register info Update DB
Send user group info Update DB
Send msg to socket in same group Update DB
Send group info Update Db
Send Server shutDown Delete client socket
end

Database

The server needs to connect with database to manage message. Here are the tables as below.

User

id	INTEGER PRIMARY KEY AUTOINCREMENT
name	Varchar(255) NOT NULL UNIQUE
password	Varchar(255) NOT NULL

Group

id	INTEGER PRIMARY KEY AUTOINCREMENT
name	Varchar(255) NOT NULL UNIQUE
msg	longtext

UserGroup

id	INTEGER PRIMARY KEY AUTOINCREMENT
userId	Int(20) NOT NULL
groupId	Int(20) NOT NULL

Function

1 **Client-Server Connection**
The app connects with the server once every 30 seconds. If the client is successfully connected to the Server, it pops up message "connect to server success" and the System Status is 'Up'. If the connection is broken, it pops up message "connect to server error" and System Status is' Down', It tries to connect server until it successfully connect server.

2 **User Registration**
User can register a new account by username and password. If the user registers successfully, the app pops up message "register success", otherwise it pops up message "register error".

3 **User Login and Logout**
User can login from login window to chat windows. If the user login successfully, the app pops up message "login success", otherwise it pops up message "login error". If user has logged in in other device right now, it does not allow user to log in again. Besides User can logout from the chat windows by pressing back button to login window.

4 **Chat Group Manage**
After login successfully, user can enter the chat group window. There is a TextView to show UP and Down state. And there is a chat group ListView for user to add, join or quit group. User can join multiple groups. When user adds a new group, he joins this group automatically. When user wants to joins group, he clicks join button, chooses the group to join in the dialog. After joining successfully, it will add that new group to group list. When user wants to quit

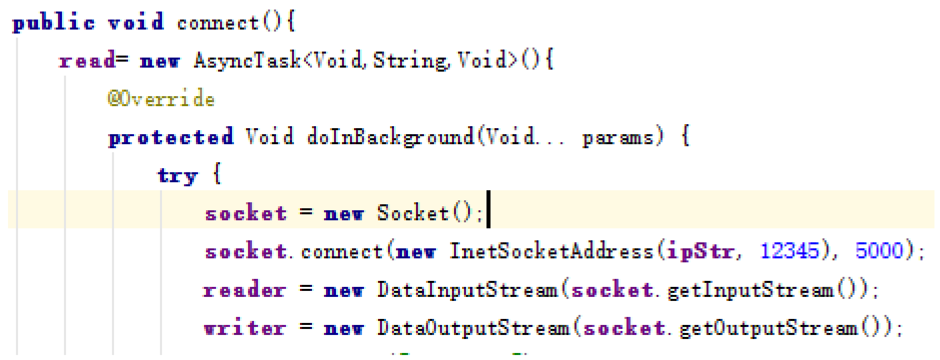
group, it is similar to join group operation. After quit successfully, it will delete that group in group list. In addition, the group message will be update in server database.

5 **Chat Message Manage**

In chat group window, user can choose the group in chat group list and app will show the chat window for that group. In chat window, user can send message to other users in that group by typing in message and click send button. The receiver window shows new message along with the sender's name by rolling down the ScrollView. When receiving the new message, the app will alert the user by showing message on the top of the screen. User can click the top message to chat window which received that new message. In addition, the chat message will be update in server database.

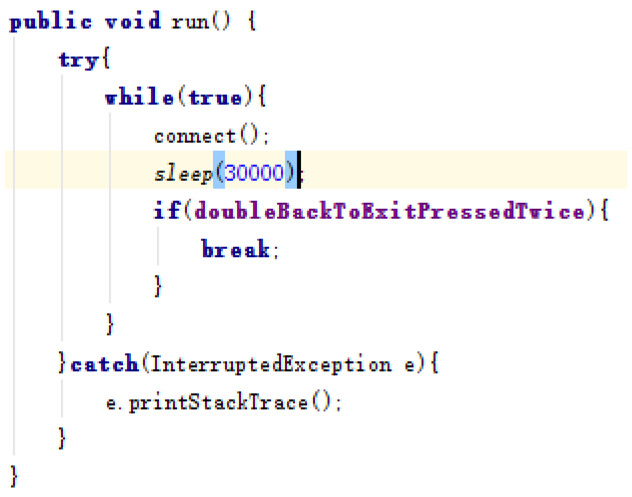
Implement

1. Client-Server Connection



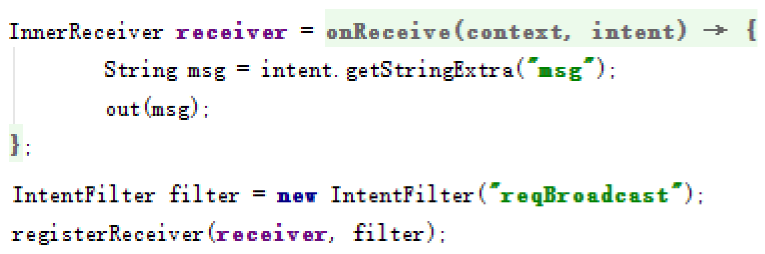
```
public void connect(){
    read= new AsyncTask<Void,String,Void>(){
        @Override
        protected Void doInBackground(Void... params) {
            try {
                socket = new Socket();
                socket.connect(new InetSocketAddress(ipStr, 12345), 5000);
                reader = new DataInputStream(socket.getInputStream());
                writer = new DataOutputStream(socket.getOutputStream());
```

As graph above, the app user socket to connect client and server.



```
public void run() {
    try{
        while(true){
            connect();
            sleep(30000);
            if(doubleBackToExitPressedTwice){
                break;
            }
        }
    }catch(InterruptedException e){
        e.printStackTrace();
    }
}
```

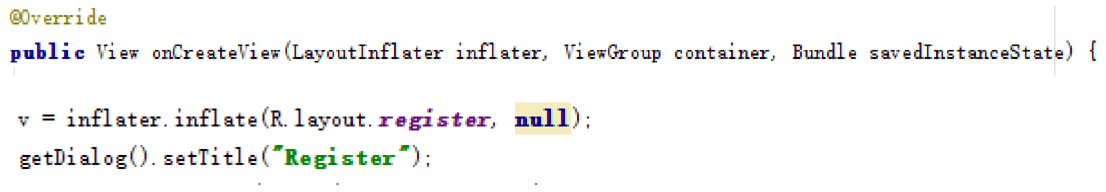
As graph above, it is the run function in a Tread class to start connection every 30s



```
InnerReceiver receiver = onReceive(context, intent) → {
        String msg = intent.getStringExtra("msg");
        out(msg);
};
IntentFilter filter = new IntentFilter("reqBroadcast");
registerReceiver(receiver, filter);
```

As graph above, InnerReceiver is used to send broadcast between activities

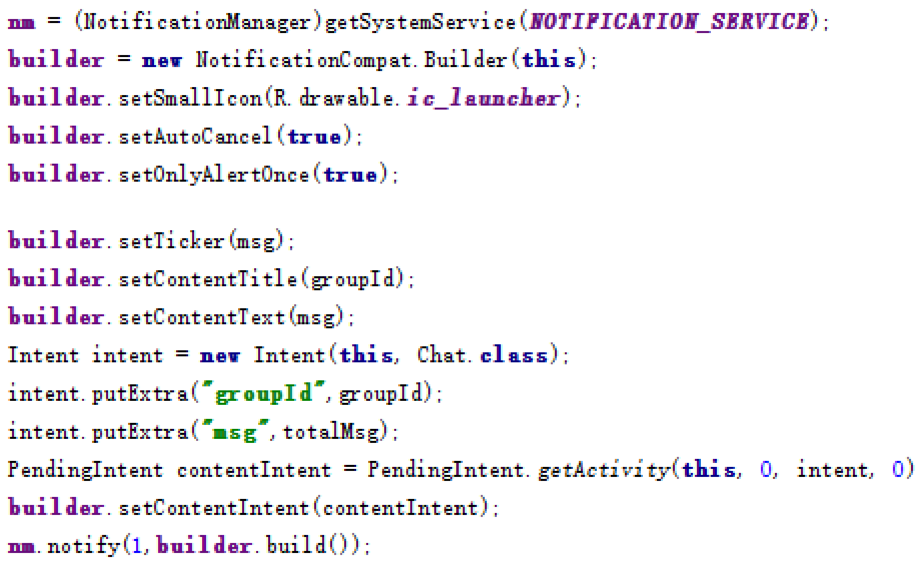
2. Show Dialog



```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

 v = inflater.inflate(R.layout.register, null);
 getDialog().setTitle("Register");
```

3. Alert received message



```
nm = (NotificationManager)getSystemService(NOTIFICATION_SERVICE);
builder = new NotificationCompat.Builder(this);
builder.setSmallIcon(R.drawable.ic_launcher);
builder.setAutoCancel(true);
builder.setOnlyAlertOnce(true);

builder.setTicker(msg);
builder.setContentTitle(groupId);
builder.setContentText(msg);
Intent intent = new Intent(this, Chat.class);
intent.putExtra("groupId",groupId);
intent.putExtra("msg",totalMsg);
PendingIntent contentIntent = PendingIntent.getActivity(this, 0, intent, 0)
builder.setContentIntent(contentIntent);
nm.notify(1,builder.build());
```

Future Function

As for alert message, I think it should be more user-friendly to tell user what's the new message. For example we can add alert color on the group list to tell which group received new message. When receiving message, user can hear the alert sound. In addiction there should be function to modify user or group's name. All of these functions should be implement for the future work.