**User Interface and Practice Team Project**

**LKN Messenger**

20122457 이호준

20142332 김형근  
20142340 노원연

**(Java File)  
MainActivity**

package com.example.khk.lknmessenger;

import android.app.Activity;

import android.app.PendingIntent;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.Toast;

import java.io.IOException;

public class MainActivity extends Activity {

PendingIntent pendingIntent;

Context mContext;

Button StartButton;

String sendMsg="Send",recvMsg="Receive";

@Override

protected void onCreate(final Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mContext = this.getApplicationContext();

StartButton = (Button)findViewById(R.id.StartButton);

StartButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent();

intent.setClass(mContext, LoginActivity.class);

pendingIntent = PendingIntent.getActivity(mContext,0,intent,0);

try {

SocketManager.getSocket();

Toast.makeText(getBaseContext(),"Execute On-line Mode",Toast.LENGTH\_LONG).show();

while( SocketManager.isConnected == false );

} catch (IOException e) {

e.printStackTrace();

}

try

{

pendingIntent.send(mContext,0,intent);

finish();

}

catch(PendingIntent.CanceledException e)

{

System.out.println("Sending contentIntent failed: ");

}

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.menu\_main, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

//noinspection SimplifiableIfStatement

if (id == R.id.action\_settings) {

return true;

}

return super.onOptionsItemSelected(item);

}

}

1. **AddFriendAck**package com.example.khk.lknmessenger;

public class AddFriendAck {

private int answer;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

answer = Packet.SUCCESS;

}

public void setAnswerFail() {

answer = Packet.FAIL;

}

}

1. **AddFrinedReq**

package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class AddFriendReq {

String FriendName;

public String getFriendName() {

return FriendName;

}

public void setFriendName(String name) {

FriendName = name;

}

}

**Database**package com.example.khk.lknmessenger;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class Database {

public static final String memberData = "Member\_Data";

public static final String messBoard = "Mess\_Board";

public static final String roomList = "Room\_Id";

public static final String friendList = "Friend\_List";

Connection con;

Statement st;

ResultSet rs;

PreparedStatement pstmt;

public String query;

public Database(){

Connection con = null;

Statement st = null;

ResultSet rs = null;

pstmt = null;

}

public boolean connect(){

try{

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/androidDB","androidDB", "androidDB");

st = con.createStatement();

}catch(SQLException e){

System.out.println("SQLException: " + e.getMessage());

System.out.println("SQLState: " + e.getSQLState());

return false;

}

return true;

}

public Statement getStatement(){

return st;

}

public Connection getConnection(){

return con;

}

public void setPreparedStatement(){

try{

pstmt = con.prepareStatement(query);

//psmt.setString(1,test);

//psmt.clearParameters(); that can recycle psmt;

}catch(SQLException e){

printError(e, query);

}

}

public void excuteStatement(){

try{

getStatement().execute(query);}

catch(Exception e)

{

e.printStackTrace();

}

}

public ResultSet excuteStatementReturnRs(){

try{

rs= getStatement().executeQuery(query);}

catch(Exception e)

{

e.printStackTrace();

}

finally{

return rs;

}

}

public PreparedStatement getPreparedStatement(){

return pstmt;

}

public void printError(SQLException e, String query){

System.out.println("SQLException: " + e.getMessage());

System.out.println("SQLState: " + e.getSQLState());

System.out.println("Query: " + query);

}

public void clear(){

String query = "";

try{

query = "delete from login\_data";

st.executeQuery(query);

query = "alter table login\_data auto\_increment = 1";

st.executeQuery(query);

}catch(SQLException e){

printError(e, query);

}

}

}

**EnterroomAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-08.

\*/

public class EnterroomAck {

private int answer;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

}

**EnterroomReq**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-08.

\*/

public class EnterroomReq {

private int roomId;

public int getRoomid() {

return roomId;

}

public void setRoomid(int arg) {

this.roomId = arg;

}

}

**GiveMemAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-08.

\*/

public class GiveMemAck {

private int memNum;

private String memberName;

private String memberId;

public int getmemNum()

{

return memNum;

}

public void setmemberNum(int arg)

{

memNum = arg;

}

public String getmemberName()

{

return memberName;

}

public void setmemberName(String arg)

{

memberName = arg;

}

public String getmemberId()

{

return memberId;

}

public void setmemberId(String arg)

{

memberId = arg;

}

}

**GiveMemReq**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class InviRoomAck {

private int answer;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

}

**InviRoomAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class InviRoomAck {

private int answer;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

}

**InviRoomReq**

**JoinAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-06.

\*/

public class JoinAck {

private int answer;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

}

**JoinActivity**package com.example.khk.lknmessenger;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.PendingIntent;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import java.io.IOException;

/\*\*

\* Created by KHK on 2015-11-24.

\*/

public class JoinActivity extends Activity {

PendingIntent pendingIntent;

Context mContext;

EditText InputName, InputID, InputPW, InputRepeat;

String UserName, UserID, UserPW, UserRepeat;

String sendMsg,recvMsg;

Button SignUp, Cancel;

Packet packet;

private JoinReq joinReq;

private JoinAck joinAck;

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.joinwindow);

mContext = this.getApplicationContext();

InputName = (EditText) findViewById(R.id.NameEdit);

InputID = (EditText) findViewById(R.id.IDEdit);

InputPW = (EditText) findViewById(R.id.PWEdit);

InputRepeat = (EditText) findViewById(R.id.RepeatEdit);

SignUp = (Button) findViewById(R.id.SignupButton);

Cancel = (Button) findViewById(R.id.CancelButton);

SignUp.setOnClickListener(OnClickListener);

Cancel.setOnClickListener(OnClickListener);

joinReq = new JoinReq();

joinAck = new JoinAck();

}

private View.OnClickListener OnClickListener = new View.OnClickListener() {

@Override

public void onClick(View view) {

int id = view.getId();

Intent intent;

switch (id) {

case R.id.SignupButton:

if (!((InputName.getText().toString() != null) && (!InputName.getText().toString().equals("")) &&

(InputID.getText().toString() != null) && (!InputID.getText().toString().equals("")) &&

(InputPW.getText().toString() != null) && (!InputPW.getText().toString().equals("")) &&

(InputRepeat.getText().toString() != null) && (!InputRepeat.getText().toString().equals("")))) {

AlertDialog.Builder dialog = new AlertDialog.Builder( JoinActivity.this );

dialog.setMessage("You have void space somewhere");

dialog.setPositiveButton("ok", null);

dialog.show();

// if text edits have no data, create an alert dialog

}

else

{

UserName = InputName.getText().toString();

UserID = InputID.getText().toString();

UserPW = InputPW.getText().toString();

UserRepeat = InputRepeat.getText().toString();

joinReq.setName(UserName);

joinReq.setId(UserID);

joinReq.setPassword(UserPW);

Log.e("UserName", UserName);

Log.e("UserID", UserID);

Log.e("UserPW",UserPW);

sendMsg = PacketCodec.encodeJoinReq(joinReq);

try {

Log.e("2sendMsg", "Link Start!!"); //Toast.makeText(getBaseContext(),"Link Start!!",Toast.LENGTH\_SHORT).show();

SocketManager.sendMsg(sendMsg);

Log.e("3sendMsg", sendMsg);

recvMsg = SocketManager.receiveMsg();

Log.e("recvMsg", recvMsg);

packet = PacketCodec.decodeHeader(recvMsg);

} catch (IOException e) {

e.printStackTrace();

}

joinAck = PacketCodec.decodeJoinAck(packet.getData());

// analyzing the ACK sent from server

if( joinAck.getAnswer() == Packet.FAIL ) {

AlertDialog.Builder dialog = new AlertDialog.Builder( JoinActivity.this );

dialog.setMessage("It's already have same ID");

dialog.setPositiveButton("ok", null);

dialog.show();

} // if the ACK means login fail, create the alert dialog

else if( joinAck.getAnswer() == Packet.SUCCESS ) {

Toast.makeText(getBaseContext(), "Sign up Clear!", Toast.LENGTH\_SHORT).show();

intent = new Intent( JoinActivity.this, LoginActivity.class );

startActivity(intent);

finish();

}

}

break;

case R.id.CancelButton:

intent = new Intent(JoinActivity.this, LoginActivity.class);

startActivity(intent);

finish();

break;

default:

}

}

};

}

**JoinReq**package com.example.khk.lknmessenger;

public class JoinReq {

private String name;

private String id;

private String password;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

**LobbyAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class LobbyAck {

private int answer;

private int RoomNum;

private String RoomName;

private int FriendNum;

private String FriendName;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

public int getRoomNum() {

return RoomNum;

}

public void setRoomNum(int num) {

RoomNum = num;

}

public String getRoomName() {

return RoomName;

}

public void setRoomName(String name) {

RoomName = name;

}

public int getFriendNum() {

return FriendNum;

}

public void setFriendNum(int num) {

FriendNum = num;

}

public String getFriendName() {

return FriendName;

}

public void setFriendName(String name) {

FriendName = name;

}

}

**LobbyActivity**package com.example.khk.lknmessenger;

import android.app.ActionBar;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.Dialog;

import android.app.DialogFragment;

import android.app.FragmentManager;

import android.app.TabActivity;

import android.content.DialogInterface;

import android.content.Intent;

import android.os.Bundle;

import android.support.v4.app.FragmentActivity;

import android.util.Log;

import android.view.Gravity;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.EditText;

import android.widget.LinearLayout;

import android.widget.ListView;

import android.widget.RelativeLayout;

import android.widget.TabHost;

import android.widget.TextView;

import android.widget.Toast;

import java.io.IOException;

import java.util.ArrayList;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class LobbyActivity extends TabActivity {

private String UserID;

private String sendMsg,recvMsg;

private TabHost mTabHost;

private TextView talklist;

private EditText friendlist;

private Button enterRoom;

private ArrayAdapter<String> arrAdapter1,arrAdapter2;

private LobbyReq lobbyReq;

private LobbyAck lobbyAck;

private MakeRoomReq makeRoomReq;

private MakeRoomAck makeRoomAck;

private EnterroomReq enterRoomReq;

private EnterroomAck enterRoomAck;

private Packet packet;

private Intent intent;

private String Roomname;

FragmentManager fm = getFragmentManager();

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.lobbywindow);

intent = getIntent();

UserID = intent.getExtras().getString("UserID");

final ArrayList arr = new ArrayList();

arr.add("Room1");

arr.add("Room2");

arr.add("Room3");

mTabHost = getTabHost();

//arrAdapter1 = new ArrayAdapter<String>(getApplicationContext(), android.R.layout.simple\_list\_item\_1);

arrAdapter2 = new ArrayAdapter<String>(getApplicationContext(), android.R.layout.simple\_list\_item\_2,arr);

// talklist.setAdapter(arrAdapter2);

enterRoom = (Button)findViewById(R.id.MakeRoom);

talklist = (TextView)findViewById(R.id.Talklist);

friendlist = (EditText)findViewById(R.id.Friendlist);

mTabHost = getTabHost();

// 탭 1, 2, 3 을 추가하면서 태그를 달아주고, 제목(또는 아이콘)을 설정한다.

mTabHost.addTab(mTabHost.newTabSpec("tab1").setContent(R.id.Friendlist).setIndicator("Friendlist"));

mTabHost.addTab(mTabHost.newTabSpec("tab2").setContent(R.id.Talklist).setIndicator("Talklist"));

enterRoom.setOnClickListener(OnClickListener);

//talklist.setOnItemClickListener(setOnItemClickListener);

//friendlist.setAdapter(arrAdapter1);

lobbyReq = new LobbyReq();

lobbyAck = new LobbyAck();

makeRoomReq = new MakeRoomReq();

makeRoomAck = new MakeRoomAck();

enterRoomReq = new EnterroomReq();

enterRoomAck = new EnterroomAck();

}

private View.OnClickListener OnClickListener = new View.OnClickListener() {

@Override

public void onClick(View v) {

EditDialog();

}

};

public void EditDialog() {

final EditText alpha;

AlertDialog.Builder editDialog;

editDialog = new AlertDialog.Builder(LobbyActivity.this);

LayoutInflater inflater = LobbyActivity.this.getLayoutInflater();

View layout = inflater.inflate(R.layout.alertmakeroom, null);

alpha = (EditText) layout.findViewById(R.id.name);

editDialog.setView(layout)

.setPositiveButton(android.R.string.ok, new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

Roomname = alpha.getText().toString();

makeRoomReq.setRoomName(Roomname);

Log.e("RoomName", Roomname);

sendMsg = PacketCodec.encodeMakeRoomReq(makeRoomReq);

try {

Log.e("2sendMsg", "Link Start!!"); //Toast.makeText(getBaseContext(),"Link Start!!",Toast.LENGTH\_SHORT).show();

SocketManager.sendMsg(sendMsg);

Log.e("3sendMsg", sendMsg);

recvMsg = SocketManager.receiveMsg();

Log.e("recvMsg", recvMsg);

packet = PacketCodec.decodeHeader(recvMsg);

} catch (IOException e) {

e.printStackTrace();

}

makeRoomAck = PacketCodec.decodeMakeRoomAck(packet.getData());

// analyzing the ACK sent from server

if( makeRoomAck.getAnswer() == Packet.FAIL ) {

Toast.makeText(getBaseContext(), "Failed to make Room", Toast.LENGTH\_SHORT).show();

} // if the ACK means login fail, create the alert dialog

else if( makeRoomAck.getAnswer() == Packet.SUCCESS ) {

Toast.makeText(getBaseContext(), "Make New Room", Toast.LENGTH\_SHORT).show();

enterRoomReq.setRoomid(1);

sendMsg = PacketCodec.encodeEnterroomReq(enterRoomReq);

Log.e("sendMsg", sendMsg);

try {

Log.e("2sendMsg", "Link Start!!"); //Toast.makeText(getBaseContext(),"Link Start!!",Toast.LENGTH\_SHORT).show();

SocketManager.sendMsg(sendMsg);

Log.e("3sendMsg", sendMsg);

recvMsg = SocketManager.receiveMsg();

Log.e("recvMsg", recvMsg);

packet = PacketCodec.decodeHeader(recvMsg);

} catch (IOException e) {

e.printStackTrace();

}

enterRoomAck = PacketCodec.decodeEnterroomAck(packet.getData());

intent = new Intent( LobbyActivity.this, MessageActivity.class );

intent.putExtra("Answer",enterRoomAck.getAnswer());

intent.putExtra("Roomname",Roomname);

intent.putExtra("RoomId",1);

startActivity(intent);

}

}

});

editDialog.show();

}

}

**LobbyReq**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class LobbyReq {

private String id;

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

}

**LoginAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-03.

\*/

public class LoginAck {

private int answer;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

}

**LoginActivity**package com.example.khk.lknmessenger;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.PendingIntent;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import java.io.BufferedReader;

import java.io.IOException;

/\*\*

\* Created by KHK on 2015-11-24.

\*/

public class LoginActivity extends Activity{

PendingIntent pendingIntent;

Context mContext;

String UserID,UserPW;

String sendMsg;

String recvMsg;

EditText IDText,PWText;

Button Login,Join;

Packet packet;

private LoginReq loginReq;

private LoginAck loginAck;

public void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.loginwindow);

mContext = this.getApplicationContext();

IDText = (EditText)findViewById(R.id.IDText);

PWText = (EditText)findViewById(R.id.PWText);

Login = (Button)findViewById(R.id.LoginButton);

Join = (Button)findViewById(R.id.JoinButton);

Login.setOnClickListener(OnClickListener);

Join.setOnClickListener(OnClickListener);

loginReq = new LoginReq();

loginAck = new LoginAck();

}

private View.OnClickListener OnClickListener = new View.OnClickListener() {

public void onClick( View view ) {

int id = view.getId();

Intent intent;

switch(id)

{

case R.id.LoginButton: // access(log in)

if (!((IDText.getText().toString() != null) && (!IDText.getText().toString().equals("")) &&

(PWText.getText().toString() != null) && (!PWText.getText().toString().equals("")))) {

AlertDialog.Builder dialog = new AlertDialog.Builder( LoginActivity.this );

dialog.setMessage("Invalid ID or password.");

dialog.setPositiveButton("ok", null);

dialog.show();

// if text edits have no data, create an alert dialog

}

else

{

UserID = IDText.getText().toString();

UserPW = PWText.getText().toString();

loginReq.setId(UserID);

loginReq.setPassword(UserPW);

sendMsg = PacketCodec.encodeLoginReq(loginReq);

Log.e("sendMsg", sendMsg);

try {

Log.e("2sendMsg", "Link Start!!"); //Toast.makeText(getBaseContext(),"Link Start!!",Toast.LENGTH\_SHORT).show();

SocketManager.sendMsg(sendMsg);

Log.e("3sendMsg", sendMsg);

recvMsg = SocketManager.receiveMsg();

Log.e("recvMsg", recvMsg);

packet = PacketCodec.decodeHeader(recvMsg);

} catch (IOException e) {

e.printStackTrace();

}

loginAck = PacketCodec.decodeLoginAck(packet.getData());

// analyzing the ACK sent from server

if( loginAck.getAnswer() == Packet.FAIL ) {

AlertDialog.Builder dialog = new AlertDialog.Builder( LoginActivity.this );

dialog.setMessage("Invalid address or password.");

dialog.setPositiveButton("ok", null);

dialog.show();

} // if the ACK means login fail, create the alert dialog

else if( loginAck.getAnswer() == Packet.SUCCESS ) {

Toast.makeText(getBaseContext(),"Login Successful!",Toast.LENGTH\_SHORT).show();

intent = new Intent( LoginActivity.this, LobbyActivity.class );

intent.putExtra("UserID",UserID);

startActivity(intent);

}

}

break;

case R.id.JoinButton:

intent = new Intent( LoginActivity.this, JoinActivity.class );

startActivity( intent );

break;

default:

}

}

};

}

**LoginReq**

package com.example.khk.lknmessenger;

public class LoginReq {

private String id;

private String password;

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

}

**MakeRoomReq**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class MakeRoomReq {

String RoomName;

public String getRoomName() {

return RoomName;

}

public void setRoomName(String name) {

RoomName = name;

}

}

**MessageActivity  
package com.example.khk.lknmessenger;**

**import android.app.Activity;**

**import android.app.AlertDialog;**

**import android.content.Intent;**

**import android.os.Bundle;**

**import android.util.Log;**

**import android.view.MotionEvent;**

**import android.view.View;**

**import android.widget.ArrayAdapter;**

**import android.widget.Button;**

**import android.widget.EditText;**

**import android.widget.ListView;**

**import android.widget.ScrollView;**

**import android.widget.Toast;**

**import java.io.IOException;**

**/\*\***

**\* Created by KHK on 2015-12-06.**

**\*/**

**public class MessageActivity extends Activity {**

**private String Roomname;**

**private int ID;**

**private Button Send;**

**private EditText editText;**

**private ScrollView scrollView;**

**private ListView listView;**

**private String message;**

**private String sendMsg,recvMsg;**

**private MssReq mssReq;**

**private MssAck mssAck;**

**private Packet packet;**

**private ArrayAdapter<String> arrAdapter ;**

**private Intent intent;**

**public void onCreate(Bundle savedInstanceState)**

**{**

**super.onCreate(savedInstanceState);**

**setContentView(R.layout.messagewindow);**

**intent = getIntent();**

**Roomname = intent.getExtras().getString("Roomname");**

**ID = intent.getExtras().getInt("RoomId");**

**scrollView = (ScrollView)findViewById(R.id.scrollView);**

**arrAdapter = new ArrayAdapter<String>(getApplicationContext(), R.layout.mylist);**

**listView = (ListView)findViewById(R.id.Message);**

**listView.setAdapter( arrAdapter ) ;**

**listView.setOnTouchListener(new View.OnTouchListener() {**

**@Override**

**public boolean onTouch(View v, MotionEvent event) {**

**scrollView.requestDisallowInterceptTouchEvent(true);**

**return false;**

**}**

**});**

**Send = (Button)findViewById(R.id.Send);**

**editText = (EditText)findViewById(R.id.EditText);**

**/\*f(message.equals(""))**

**Send.setEnabled(false);**

**else**

**Send.setEnabled(true);\*/**

**Send.setOnClickListener(OnClickListener);**

**mssReq = new MssReq();**

**mssAck = new MssAck();**

**}**

**private View.OnClickListener OnClickListener = new View.OnClickListener() {**

**@Override**

**public void onClick(View v) {**

**message = editText.getText().toString();**

**mssReq.setMessage(message);**

**sendMsg = PacketCodec.encodeMssReq(mssReq);**

**Log.e("sendMsg", sendMsg);**

**try {**

**Log.e("2sendMsg", "Link Start!!"); //Toast.makeText(getBaseContext(),"Link Start!!",Toast.LENGTH\_SHORT).show();**

**SocketManager.sendMsg(sendMsg);**

**Log.e("3sendMsg", sendMsg);**

**recvMsg = SocketManager.receiveMsg();**

**Log.e("recvMsg", recvMsg);**

**packet = PacketCodec.decodeHeader(recvMsg);**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**mssAck = PacketCodec.decodeMssAck(packet.getData());**

**// analyzing the ACK sent from server**

**if( mssAck.getAnswer() == Packet.FAIL ) {**

**AlertDialog.Builder dialog = new AlertDialog.Builder( MessageActivity.this );**

**dialog.setMessage("Please Input Correct Sentence");**

**dialog.setPositiveButton("ok", null);**

**dialog.show();**

**} // if the ACK means login fail, create the alert dialog**

**else if( mssAck.getAnswer() == Packet.SUCCESS ) {**

**arrAdapter.add(message);**

**message = "";**

**editText.setText("");**

**String[] arg;**

**MssAndArrtimeAndUser[] mau;**

**arg = PacketCodec.nextDecode(mssAck.getListnum(), mssAck.getlist());**

**mau = PacketCodec.nextTinydecode(arg.length,arg);**

**for(int i = 0; i<mau.length;i++) {**

**Log.e("Text", mau[i].getMss());**

**arrAdapter.add(mau[i].getMss());**

**}**

**}**

**}**

**};**

**}**

**MssAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class MssAck {

private int answer;

private String arrTime;

private int listNum;

private String list;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

public String getArrtime() {

return arrTime;

}

public void setArrtime(String arg) {

this.arrTime = arg;

}

public int getListnum() {

return listNum;

}

public void setListNum(int arg) {

this.listNum = arg;

}

public String getlist() {

return list;

}

public void setlist(String arg) {

this.list = arg;

}

}

**MssAndArrtimeAndUser**package com.example.khk.lknmessenger;

/\*\*

\* Created by 호준 on 2015-12-08.

\*/

public class MssAndArrtimeAndUser {

private String Mss;

private String Arrtime;

private String Username;

public String getMss() {

return Mss;

}

public void setMss(String name) {

Mss = name;

}

public String getArrtime() {

return Arrtime;

}

public void setArrtime(String arrTime) {

Arrtime = arrTime;

}

public String getUsername() {

return Username;

}

public void setName(String name) {

Username = name;

}

public String getAll(){

String all =null;

return all;

}

}

**MssReq**package com.example.khk.lknmessenger;

public class MssReq {

private String message;

public String getMessage() {

return message;

}

public void setMessage(String message) {

this.message = message;

}

}

**Packet**package com.example.khk.lknmessenger;

public class Packet {

public static final String JOIN\_REQ = "JOIN\_REQ";

public static final String JOIN\_ACK = "JOIN\_ACK";

public static final String LOG\_REQ = "LOG\_REQ";

public static final String LOG\_ACK = "LOG\_ACK";

public static final String MSS\_REQ = "MSS\_REQ";

public static final String MSS\_ACK = "MSS\_ACK";

public static final String INVIROOM\_REQ = "INVIROOM\_REQ";

public static final String INVIROOM\_ACK = "INVIROOM\_ACK";

public static final String MAKEROOM\_REQ = "MAKEROOM\_REQ";

public static final String MAKEROOM\_ACK = "MAKEROOM\_ACK";

public static final String ADDFRIEND\_REQ = "ADDFRIEND\_REQ";

public static final String ADDFRIEND\_ACK = "ADDFRIEND\_ACK";

public static final String LOBBY\_REQ = "LOBBY\_REQ";

public static final String LOBBY\_ACK = "LOBBY\_ACK";

public static final String ROOM\_REQ = "ROOM\_REQ";

public static final String ROOM\_ACK = "ROOM\_ACK";

public static final String GIVEMEM\_REQ = "GIVEMEM\_REQ";

public static final String GIVEMEM\_ACK = "GIVEMEM\_ACK";

public static final String ENTERROOM\_REQ = "ENTERROOM\_REQ";

public static final String ENTERROOM\_ACK = "ENTERROOM\_ACK";

public static final String FIELD\_DELIM = "|";

public static final String PK\_DELIM = "?";

public static final String SMALLDELIM = ":";

public static final String TINYDELIM = "'";

public static final int SUCCESS = 0x01;

public static final int FAIL = 0x00;

private String type;

private String data;

public Packet(String type, String data){

this.type = type;

this.data = data;

}

public String getType() {

return type;

}

public void setType(String type) {

this.type = type;

}

public String getData() {

return data;

}

public void setData(String data) {

this.data = data;

}

}

**PacketCodec**package com.example.khk.lknmessenger;

import java.io.BufferedReader;

import java.io.IOException;

import java.util.Scanner;

public class PacketCodec {

public static String readBuffReader(BufferedReader in) throws IOException{

char charBuf[] = new char[1];

String src = "";

while(in.read(charBuf, 0, 1) != -1)

{

if(charBuf[0] == '\n')

break;

src += charBuf[0];

}

//System.out.println("readBufferReader step two...while state END" + charBuf[0]);

if(src.equals(""))

return null;

return src;

}

public static Packet decodeHeader(String src) throws IOException{

String type, data;

System.out.println("Decode : START...");

Scanner s = new Scanner(src).useDelimiter("\\"+Packet.FIELD\_DELIM);

type = s.next();

s.skip(Packet.FIELD\_DELIM);

s.useDelimiter("\\"+Packet.PK\_DELIM);

data = s.next();

System.out.println("Decode : END...");

return new Packet(type, data);

}

// About join request

// Dncode join request packet data

public static String encodeJoinReq(JoinReq pk\_data){

String data = Packet.JOIN\_REQ

+ Packet.FIELD\_DELIM + pk\_data.getName()

+ Packet.FIELD\_DELIM + pk\_data.getId()

+ Packet.FIELD\_DELIM + pk\_data.getPassword()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static JoinReq decodeJoinReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

JoinReq dst = new JoinReq();

dst.setName(s.next());

s.skip(Packet.FIELD\_DELIM);

dst.setId(s.next());

s.skip(Packet.FIELD\_DELIM);

dst.setPassword(s.next());

return dst;

}

public static String encodeJoinAck(JoinAck pk\_data){

String data = Packet.LOG\_ACK + Packet.FIELD\_DELIM

+ Integer.toString(pk\_data.getAnswer()) + Packet.FIELD\_DELIM;

/\*if (pk\_data.getAnswer() == Packet.SUCCESS){

}\*/

data += Packet.PK\_DELIM;

return data;

}

public static JoinAck decodeJoinAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

JoinAck dst = new JoinAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

return dst;

}

public static String encodeLoginReq(LoginReq pk\_data){

String data = Packet.LOG\_REQ

+ Packet.FIELD\_DELIM + pk\_data.getId()

+ Packet.FIELD\_DELIM + pk\_data.getPassword()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static LoginReq decodeLoginReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

LoginReq dst = new LoginReq();

dst.setId(s.next());

s.skip(Packet.FIELD\_DELIM);

dst.setPassword(s.next());

return dst;

}

public static String encodeLoginAck(LoginAck pk\_data){

String data = Packet.LOG\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM;

/\*if (pk\_data.getAnswer() == Packet.SUCCESS){

}\*/

data += Packet.PK\_DELIM;

return data;

}

public static LoginAck decodeLoginAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

LoginAck dst = new LoginAck();

if(s.nextInt()==Packet.SUCCESS)

dst.setAnswerOk();

else

dst.setAnswerFail();

return dst;

}

public static String encodeMssReq(MssReq pk\_data){

String data = Packet.MSS\_REQ

+ Packet.FIELD\_DELIM + pk\_data.getMessage()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static MssReq decodeMssReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

MssReq dst = new MssReq();

dst.setMessage(s.next());

return dst;

}

public static String encodeMssAck(MssAck pk\_data){

String data = Packet.MSS\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM + pk\_data.getArrtime()

+ Packet.FIELD\_DELIM + pk\_data.getlist()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static MssAck decodeMssAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

MssAck dst = new MssAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

s.skip(Packet.FIELD\_DELIM);

dst.setArrtime(s.next());

s.skip(Packet.FIELD\_DELIM);

dst.setlist(s.next());

return dst;

}

public static String encodeInviRoomReq(InviRoomReq pk\_data){

String data = Packet.INVIROOM\_REQ

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static InviRoomReq decodeInviRoomReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

InviRoomReq dst = new InviRoomReq();

return dst;

}

public static String encodeInviRoomAck(InviRoomAck pk\_data ){

String data = Packet.INVIROOM\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static InviRoomAck decodeInviRoomAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

InviRoomAck dst = new InviRoomAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

return dst;

}

public static String encodeMakeRoomReq(MakeRoomReq pk\_data){

String data = Packet.MAKEROOM\_REQ

+ Packet.FIELD\_DELIM + pk\_data.getRoomName()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static MakeRoomReq decodeMakeRoomReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

MakeRoomReq dst = new MakeRoomReq();

dst.setRoomName(s.next());

return dst;

}

public static String encodeMakeRoomAck(MakeRoomAck pk\_data ){

String data = Packet.MAKEROOM\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static MakeRoomAck decodeMakeRoomAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

MakeRoomAck dst = new MakeRoomAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

return dst;

}

public static String encodeAddFriendReq(AddFriendReq pk\_data){

String data = Packet.ADDFRIEND\_REQ

+ Packet.FIELD\_DELIM + pk\_data.getFriendName()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static AddFriendReq decodeAddFriendReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

AddFriendReq dst = new AddFriendReq();

dst.setFriendName(s.next());

return dst;

}

public static String encodeAddFrinedAck(AddFriendAck pk\_data ){

String data = Packet.ADDFRIEND\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static AddFriendAck decodeAddFriendAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

AddFriendAck dst = new AddFriendAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

return dst;

}

public static String encodeLobbyReq(LobbyReq pk\_data){

String data = Packet.LOBBY\_REQ

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static LobbyReq decodeLobbyReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

LobbyReq dst = new LobbyReq();

return dst;

}

public static String encodeLobbyAck(LobbyAck pk\_data ){

String data = Packet.LOBBY\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getRoomNum())

+ Packet.FIELD\_DELIM + pk\_data.getRoomName()

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getFriendNum())

+ Packet.FIELD\_DELIM + pk\_data.getFriendName()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static LobbyAck decodeLobbyAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

LobbyAck dst = new LobbyAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

s.skip(Packet.FIELD\_DELIM);

dst.setRoomNum(s.nextInt());

s.skip(Packet.FIELD\_DELIM);

dst.setRoomName(s.next());

s.skip(Packet.FIELD\_DELIM);

dst.setFriendNum(s.nextInt());

s.skip(Packet.FIELD\_DELIM);

dst.setFriendName(s.next());

return dst;

}

public static String encodeRoomReq(RoomReq pk\_data){

String data = Packet.ROOM\_REQ

+ Packet.FIELD\_DELIM + pk\_data.getRoomname()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static RoomReq decodeRoomReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

RoomReq dst = new RoomReq();

dst.setRoomname(s.next());

return dst;

}

public static String encodeRoomAck(RoomAck pk\_data ){

String data = Packet.ROOM\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getMauNum())

+ Packet.FIELD\_DELIM + pk\_data.getMau()

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getMemberNum())

+ Packet.FIELD\_DELIM + pk\_data.getMember()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static RoomAck decodeRoomAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

RoomAck dst = new RoomAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

s.skip(Packet.FIELD\_DELIM);

dst.setMauNum(s.nextInt());

s.skip(Packet.FIELD\_DELIM);

dst.setMau(s.next());

s.skip(Packet.FIELD\_DELIM);

dst.setMemberNum(s.nextInt());

s.skip(Packet.FIELD\_DELIM);

dst.setMember(s.next());

return dst;

}

public static String encodeGiveMemReq(GiveMemReq pk\_data){

String data = Packet.GIVEMEM\_REQ

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static GiveMemReq decodeGiveMemReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

GiveMemReq dst = new GiveMemReq();

return dst;

}

public static String encodeGiveMemAck(GiveMemAck pk\_data ){

String data = Packet.GIVEMEM\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getmemNum())

+ Packet.FIELD\_DELIM + pk\_data.getmemberName()

+ Packet.FIELD\_DELIM + pk\_data.getmemberId()

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static String preEncode(int num, String[] mem)

{

String send = "";

send += Integer.toString(num);

for(int i = 0; i<num; i++)

{ send+=Packet.SMALLDELIM;

send+=mem[i];

}

return send;

}

public static GiveMemAck decodeGiveMemAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

GiveMemAck dst = new GiveMemAck();

dst.setmemberNum(s.nextInt());

s.skip(Packet.FIELD\_DELIM);

dst.setmemberName(s.next());

s.skip(Packet.FIELD\_DELIM);

dst.setmemberId(s.next());

return dst;

}

public static String[] nextDecode(int num, String mem)

{

String[] member = new String[num];

Scanner s = new Scanner(mem).useDelimiter(Packet.SMALLDELIM);

for(int i =0 ;i<num;i++){

member[i] = s.next();

s.skip(Packet.SMALLDELIM);

}

return member;

}

public static String encodeEnterroomReq(EnterroomReq pk\_data){

String data = Packet.ENTERROOM\_REQ

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getRoomid())

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static EnterroomReq decodeEnterroomReq(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

EnterroomReq dst = new EnterroomReq();

dst.setRoomid(s.nextInt());

return dst;

}

public static String encodeEnterroomAck(EnterroomAck pk\_data ){

String data = Packet.ENTERROOM\_ACK

+ Packet.FIELD\_DELIM + Integer.toString(pk\_data.getAnswer())

+ Packet.FIELD\_DELIM

+ Packet.PK\_DELIM;

return data;

}

public static EnterroomAck decodeEnterroomAck(String pk\_data){

Scanner s = new Scanner(pk\_data).useDelimiter("\\"+Packet.FIELD\_DELIM);

EnterroomAck dst = new EnterroomAck();

if(Packet.SUCCESS==s.nextInt())

dst.setAnswerOk();

else

dst.setAnswerFail();

return dst;

}

public static MssAndArrtimeAndUser[] nextTinydecode(int num,String[] arg)

{

MssAndArrtimeAndUser[] mau = new MssAndArrtimeAndUser[num];

for(int i =0 ;i<num;i++){

Scanner s = new Scanner(arg[i]).useDelimiter(Packet.TINYDELIM);

for(int j =0 ;j<3;j++){

mau[j].setName(s.next());

s.skip(Packet.TINYDELIM);

mau[j].setMss(s.next());

s.skip(Packet.TINYDELIM);

mau[j].setArrtime(s.next());

s.skip(Packet.TINYDELIM);

}

}

return mau;

}

}

**RoomAck**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class RoomAck {

private int answer;

private int mauNum ;

private String mau;

private int memberNum;

private String member;

public int getAnswer() {

return answer;

}

public void setAnswerOk() {

this.answer = Packet.SUCCESS;

}

public void setAnswerFail()

{

this.answer = Packet.FAIL;

}

public int getMauNum() {

return mauNum;

}

public void setMauNum(int num) {

mauNum = num;

}

public String getMau() {

return mau;

}

public void setMau(String mau) {

this.mau = mau ;

}

public int getMemberNum() {

return memberNum;

}

public void setMemberNum(int num) {

memberNum = num;

}

public String getMember() {

return member;

}

public void setMember(String member) {

this.member = member;

}

}

**RoomReq**package com.example.khk.lknmessenger;

/\*\*

\* Created by KHK on 2015-12-07.

\*/

public class RoomReq {

String Roomname;

public String getRoomname()

{

return Roomname;

}

public void setRoomname(String arg)

{

Roomname = arg;

}

}

**SocketManager**package com.example.khk.lknmessenger;

import android.app.Activity;

import android.app.ProgressDialog;

import android.os.Handler;

import android.os.Message;

import android.util.Log;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.InetAddress;

import java.net.InetSocketAddress;

import java.net.Socket;

/\*\*

\* Created by KHK on 2015-12-05.

\*/

public class SocketManager{

public final static String HOST = "115.23.216.223";

public final static int PORT = 9193;

//public static Thread worker;

public static Socket socket;

public static PrintWriter socket\_out;

public static BufferedReader socket\_in;

//public static Thread worker;

// variables about socket

public static String result;

// String variable received from the server

public static boolean isConnected = false;

public static boolean isGetEnd = false;

// boolean variables for determining whether

// the socket is connected or not and the

// received string is full or not

public static LoginActivity loginActivity;

public static ProgressDialog dialog;

public static void getSocket() throws IOException {

// method for initializing socket and

// connecting to the server

Thread worker = new Thread() {

/\* Since the Android 4.4(Kitkat), the android system

\* doesn't allow thread tasks which requires lots of time,

\* including socket thread, in main thread. So we have to

\* create other threads for executing these thread tasks. \*/

public void run() {

//super.run();

if(socket == null)

socket = new Socket();

if (!socket.isConnected())

try {

socket.connect( new InetSocketAddress(InetAddress.getByName("sayaten.net"), PORT) );

Log.e("Socket: ", "Socket OK");

socket\_in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

Log.e("Socket in: ", "Socket in OK");

socket\_out = new PrintWriter(socket.getOutputStream(), true);

Log.e("Socket out: ", "Socket out OK");

isConnected = true;

//throw new IOException();

// allocating variables about socket

} catch (IOException e) {

//e.printStackTrace();

}

}

};

worker.start();

Log.e("Socket Last: ", "REQ");

}

public static void closeSocket() throws IOException

{

// method for closing socket

if( socket != null ) {

Log.e("Close", "Close Socket");

socket.close();

}

}

public static void sendMsg( String message ) throws IOException {

// method for sending a packet to the server

if(!socket.isConnected())

{

Log.e("Socket: ","No");

}

socket\_out.println( message );

}

public static String receiveMsg() throws IOException {

// method for receiving a packet from the server

Thread receiver = new Thread() {

/\* Unlike sending to the server, receiving a data from the

\* server needs thread tasking. So we have to create another

\* thread for receiving from the server. \*/

public void run() {

try {

result = PacketCodec.readBuffReader(socket\_in);

isGetEnd = true;

} catch (IOException e) {

e.printStackTrace();

}

}

};

isGetEnd = false;

receiver.start();

while( !isGetEnd );

return result;

}

public static Handler handler = new Handler() {

public void handleMessage(Message msg) {

dialog.dismiss();

}

};

}

**TcpServer**package com.example.khk.lknmessenger;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

public class TcpServer {

private static Thread thread = null;

private static ServerSocket serverSocket = null;

private static Socket clientSocket = null;

private static final int PORT = 9193; // can use 9193 9194 9195

private static ThreadTcp threadServer = null;

public static void main(String[] args) throws IOException{

System.out.println("Server is operating.....");

while(true)

{

serverSocket = new ServerSocket(PORT);

serverSocket.setReuseAddress(true);

while(true) {

clientSocket = serverSocket.accept();

if(clientSocket != null){

threadServer = new ThreadTcp(clientSocket, true);

thread = new Thread(threadServer);

thread.start();

clientSocket = null;

}

}

}

}

}

**ThreadTcp**

package com.example.khk.lknmessenger;

import android.util.Log;

import android.widget.Toast;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.ServerSocket;

import java.net.Socket;

import java.sql.ResultSet;

public class ThreadTcp implements Runnable{

//private Base64Codec bs64 = new Base64Codec();//

private Socket clientSocket = null;//

private PrintWriter out = null;

private BufferedReader in = null;

private boolean isContinous = false;

private int user\_id = 0;

private Database db;

String inputData;

Packet rcvPacket;

ResultSet rs;

public ThreadTcp(Socket clientSocket, boolean isContinous) throws IOException{

this.clientSocket = clientSocket;

this.isContinous = isContinous;

System.out.println("Client Connect");

db = new Database();

if(db.connect())

System.out.println("DB Connect");

else

System.out.println("DB Fail...");

}

public ThreadTcp(ServerSocket serverSocket, boolean isContinous) throws IOException{

clientSocket = serverSocket.accept();

this.isContinous = isContinous;

System.out.println("Client Connect");

if(db.connect())

System.out.println("DB Connect");

else

System.out.println("DB Fail...");

}

public void run(){

try{

System.out.println("ThreadTcp run step zero");

in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));//

out = new PrintWriter(clientSocket.getOutputStream(), true);

System.out.println("ThreadTcp run step one");

while(isContinous){

inputData = PacketCodec.readBuffReader(in);

if(inputData==null){

continue;

}

rcvPacket = PacketCodec.decodeHeader(inputData);//

System.out.println("ThreadTcp step four");

isContinous = handler(rcvPacket, out);//

rcvPacket=null;

}//

in.close();

out.close();

clientSocket.close();

}catch(Exception e){

e.printStackTrace();

}

}

public boolean handler(Packet src, PrintWriter out)throws IOException

{

String sendString;

switch(src.getType()){

case Packet.LOG\_REQ:

System.out.println("Log REQ recevied.");

LoginReq lo\_req = PacketCodec.decodeLoginReq(src.getData());

LoginAck lo\_ack = new LoginAck();

db.query = "select Id, Pw, Dbid from "+Database.memberData;//+" where \* Dbid";

rs = db.excuteStatementReturnRs();

try{

while(rs.next())

{

if(lo\_req.getId().equals(rs.getString("Id"))){

System.out.println("Login Ack : Id\_Success");

if(lo\_req.getPassword().equals(rs.getString("Pw"))){

lo\_ack.setAnswerOk();

user\_id=rs.getInt("Dbid");

System.out.println("Login Ack : Success");

break;

}

}

}

if(lo\_ack.getAnswer()!=Packet.SUCCESS){

lo\_ack.setAnswerFail();

System.out.println("Login Ack : Fail");

}

sendString = PacketCodec.encodeLoginAck(lo\_ack);

try{

out.println(sendString);

System.out.println("Login Ack dispatched.");

}

catch(Exception e){

e.printStackTrace();

}

}

catch(Exception e)

{

e.printStackTrace();

}

break;

case Packet.JOIN\_REQ:

System.out.println("Join REQ recevied");

JoinReq jo\_req = PacketCodec.decodeJoinReq(src.getData());

JoinAck jo\_ack = new JoinAck();

db.query = "select Id from "+Database.memberData;//+" where \* Dbid";

rs = db.excuteStatementReturnRs();

try{

while(rs.next())

{

if(jo\_req.getId().equals(rs.getString("Id"))){

jo\_ack.setAnswerFail();

System.out.println("Join Ack : Fail");

break;

}

}

if(jo\_ack.getAnswer()!=Packet.FAIL){

jo\_ack.setAnswerOk();

db.query = "insert into "+Database.memberData+" (Name,Id,Pw) values "

+"('"+jo\_req.getName()+"','"+jo\_req.getId()+"','"+jo\_req.getPassword()+"')";

db.excuteStatement();

System.out.println("Join Ack : Success");

}

sendString = PacketCodec.encodeJoinAck(jo\_ack);

try{

out.println(sendString);

System.out.println("JOin Ack dispatched.");

}

catch(Exception e){

e.printStackTrace();

}

}

catch(Exception e)

{

e.printStackTrace();

}

break;

case Packet.MSS\_REQ:

System.out.println("MSS REQ recevied");

MssReq mss\_req = PacketCodec.decodeMssReq(src.getData());

MssAck mss\_ack = new MssAck();

mss\_ack.setAnswerOk();

sendString = PacketCodec.encodeMssAck(mss\_ack);

out.println(sendString);

System.out.println("Mss Ack dispatched.");

break;

case Packet.GIVEMEM\_REQ:

System.out.println("GIVEMEM REQ recevied");

GiveMemReq give\_req = PacketCodec.decodeGiveMemReq(src.getData());

GiveMemAck give\_ack = new GiveMemAck();

db.query = "select Id, Name from "+Database.memberData;//+" where \* Dbid";

rs = db.excuteStatementReturnRs();

int count = 0;

String name="";

String Id="";

try{

while(rs.next())

{

count++;

Id+=rs.getString("Id");

Id+=Packet.SMALLDELIM;

name+=rs.getString("Name");

name+=Packet.SMALLDELIM;

}

give\_ack.setmemberNum(count);

give\_ack.setmemberName(name);

give\_ack.setmemberId(Id);

sendString = PacketCodec.encodeGiveMemAck(give\_ack);

try{

out.println(sendString);

System.out.println("JOin Ack dispatched.");

}

catch(Exception e){

e.printStackTrace();

}

}

catch(Exception e)

{

e.printStackTrace();

}

break;

}

return isContinous;

}

}

**(XML File)  
activity\_main.xml**<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:paddingLeft="@dimen/activity\_horizontal\_margin"

android:paddingRight="@dimen/activity\_horizontal\_margin"

android:paddingTop="@dimen/activity\_vertical\_margin"

android:paddingBottom="@dimen/activity\_vertical\_margin"

tools:context=".MainActivity"

android:background="#2cb26e">

<TextView

android:id="@+id/Logo"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:text="LKN Messenger"

android:textSize="60dp"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

style="@style/Base.Widget.AppCompat.Button.Borderless.Colored"

android:textColor="#ffffff" />

<Button

android:id="@+id/StartButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Start Messenger"

android:layout\_alignParentBottom="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="156dp"

android:background="#1e875f"

android:breakStrategy="balanced"

android:textColor="#ffffff" />

</RelativeLayout>

**alertmakeroom.xml**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<EditText

android:id="@+id/name"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="16dp"

android:layout\_marginLeft="4dp"

android:layout\_marginRight="4dp"

android:layout\_marginBottom="4dp"

android:hint = "Input Room Name"

android:background="#ffffff"

android:textColor="#000000" />

</LinearLayout>

**joinwindow.xml**<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#2cb26e">

<TextView

android:id="@+id/NameJoin"

android:text="Input Name"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_above="@+id/IDEdit"

android:layout\_alignParentStart="true"

android:textColor="#ffffff" />

<EditText

android:id="@+id/NameEdit"

android:layout\_width="300dp"

android:layout\_height="wrap\_content"

android:layout\_alignParentTop="true"

android:layout\_alignParentEnd="true"

android:layout\_marginTop="29dp"

android:layout\_alignStart="@+id/IDEdit"

android:textColor="#ffffff" />

<TextView

android:id="@+id/IDJoin"

android:text="Input ID"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_alignTop="@+id/IDEdit"

android:layout\_alignParentStart="true"

android:textColor="#ffffff" />

<EditText

android:id="@+id/IDEdit"

android:layout\_width="300dp"

android:layout\_height="wrap\_content"

android:layout\_below="@+id/NameEdit"

android:layout\_marginTop="39dp"

android:layout\_toEndOf="@+id/RepeatJoin"

android:textColor="#ffffff" />

<TextView

android:id="@+id/PWJoin"

android:text="Input Password"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_below="@+id/IDEdit"

android:layout\_alignParentStart="true"

android:layout\_marginTop="46dp"

android:textColor="#ffffff" />

<EditText

android:id="@+id/PWEdit"

android:layout\_width="300dp"

android:layout\_height="wrap\_content"

android:layout\_alignTop="@+id/PWJoin"

android:layout\_alignParentEnd="true"

android:layout\_toEndOf="@+id/RepeatJoin"

android:textColor="#ffffff" />

<TextView

android:id="@+id/RepeatJoin"

android:text="Repeat Password"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_below="@+id/IDEdit"

android:layout\_alignParentStart="true"

android:layout\_marginTop="126dp"

android:textColor="#ffffff" />

<EditText

android:id="@+id/RepeatEdit"

android:layout\_width="300dp"

android:layout\_height="wrap\_content"

android:layout\_alignTop="@+id/RepeatJoin"

android:layout\_alignParentEnd="true"

android:layout\_toEndOf="@+id/RepeatJoin"

android:textColor="#ffffff" />

<Button

android:id="@+id/SignupButton"

android:text="Sign Up"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@+id/RepeatEdit"

android:layout\_alignEnd="@+id/PWJoin"

android:layout\_marginTop="121dp"

android:textColor="#ffffff"

android:background="#1e875f" />

<Button

android:id="@+id/CancelButton"

android:text="Cancel"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textColor="#ffffff"

android:background="#1e875f"

android:layout\_alignTop="@+id/SignupButton"

android:layout\_alignStart="@+id/RepeatEdit" />

</RelativeLayout> **lobbywindow.xml**<?xml version="1.0" encoding="utf-8"?>

<TabHost xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@android:id/tabhost">

<LinearLayout android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:orientation="vertical"

android:weightSum="1">

<!-- Tab Widget -->

<TabWidget android:id="@android:id/tabs"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:background="#1e875f" />

<!-- Tab Content -->

<FrameLayout android:id="@android:id/tabcontent"

android:layout\_width="fill\_parent"

android:layout\_height="384dp"

android:layout\_weight="0.21">

<!-- Tab 1 -->

<EditText

android:id="@+id/Friendlist"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:hint = "Friend"/>

<!-- Tab 2 -->

<TextView

android:id="@+id/Talklist"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Talklist"/>

<!-- Tab 3 -->

</FrameLayout>

<Button

android:id="@+id/MakeRoom"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Create Room"

android:background="#1e875f"

android:textColor="#ffffff" />

</LinearLayout>

</TabHost> **loginwindow.xml**<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical" android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="#2cb26e">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:text="LKN Messenger"

android:id="@+id/Logo"

android:textSize="60dp"

android:layout\_alignParentTop="true"

android:layout\_centerHorizontal="true"

style="@style/Base.Widget.AppCompat.Button.Borderless.Colored"

android:textColor="#ffffff" />

<EditText

android:id="@+id/IDText"

android:hint="Input ID"

android:layout\_width="300dp"

android:layout\_height="wrap\_content"

android:layout\_above="@+id/PWText"

android:layout\_alignStart="@+id/PWText"

android:textColor="#ffffff" />

<EditText

android:id="@+id/PWText"

android:hint="Input Password"

android:layout\_width="300dp"

android:layout\_height="wrap\_content"

android:layout\_centerVertical="true"

android:layout\_centerHorizontal="true"

android:textColor="#ffffff" />

<Button

android:id="@+id/LoginButton"

android:text="Log in"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@+id/PWText"

android:layout\_alignStart="@+id/PWText"

android:layout\_marginStart="48dp"

android:layout\_marginTop="41dp"

android:background="#1e875f"

android:textColor="#ffffff" />

<Button

android:id="@+id/JoinButton"

android:text="Join"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:background="#1e875f"

android:textColor="#ffffff"

android:layout\_alignTop="@+id/LoginButton"

android:layout\_toEndOf="@+id/LoginButton"

android:layout\_marginStart="22dp" />

</RelativeLayout>

**messagewindow.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical" android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:weightSum="1">

<ScrollView

android:id="@+id/scrollView"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:fillViewport="true">

<ListView

android:id="@+id/Message"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

</ScrollView>

<Button

android:id="@+id/Send"

android:text="Send"

android:layout\_width="91dp"

android:layout\_height="64dp"

android:layout\_gravity="right"

android:layout\_alignTop="@+id/EditText"

android:layout\_alignParentEnd="true" />

<EditText

android:id="@+id/EditText"

android:layout\_width="236dp"

android:layout\_height="58dp"

android:layout\_alignParentBottom="true"

android:layout\_alignParentStart="true"

android:layout\_toStartOf="@+id/Send" />

</RelativeLayout>

**Mylist.xml**<?xml version="1.0" encoding="utf-8"?>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@android:id/text1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceSmall"

android:gravity="center\_vertical"

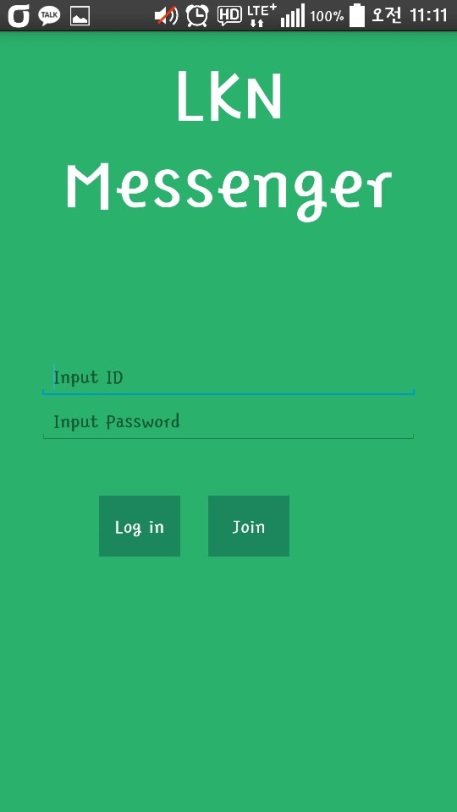
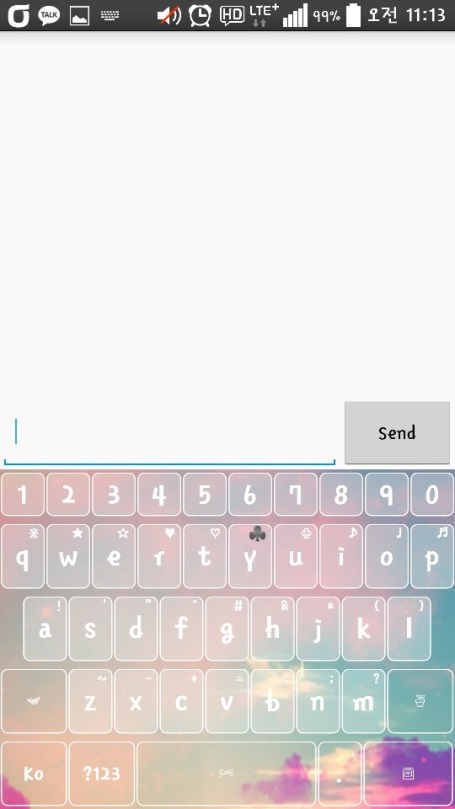
android:paddingStart="2dip"

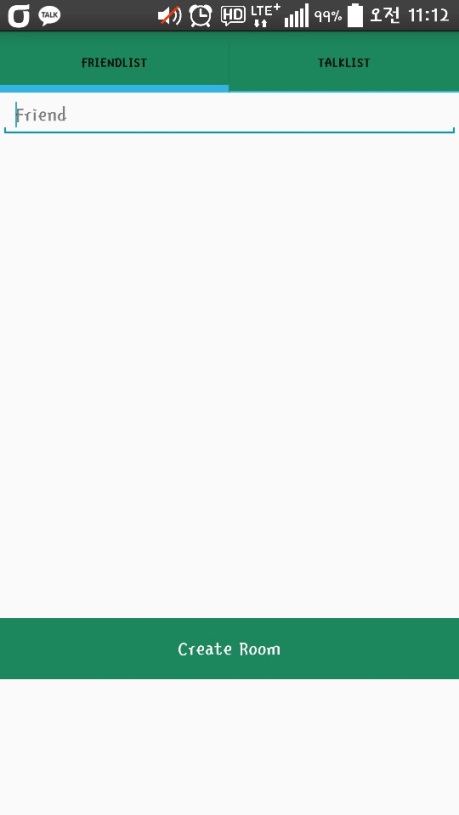
android:textSize="21sp"

android:textColor="#000000"

android:minHeight="34dp"

/>

**ScreenShot   
 **

****