4 QQ项目实现

## 4.1安卓平台系统实现

### 4.1.1 功能描述

整个项目功能完善，包含登录注册、好友列表、好友信息、搜索好友、添加删除好友、与好友聊天、好友动态等几个功能模块。每个模块组成整个系统功能，功能完善，具有真正QQ的功能。

### 4.1.2 界面展示

注册界面主要来注册登录用户，是用来实现后续好友、聊天、搜索好友、添加删除好友的基本操作单位。



图4-1 注册截图

输入数据注册成功：



图4-2 注册成功截图

登录界面主要是由用户来操作的，所以要简单明了易操作。利用数据库对用户信息进行核对，核对正确后利用界面跳转到QQ主界面。用户的账号密码是提前自己在注册界面注册并保存在服务器数据库中的，每个用户都有自己单独的唯一的一个账号密码用来登录。

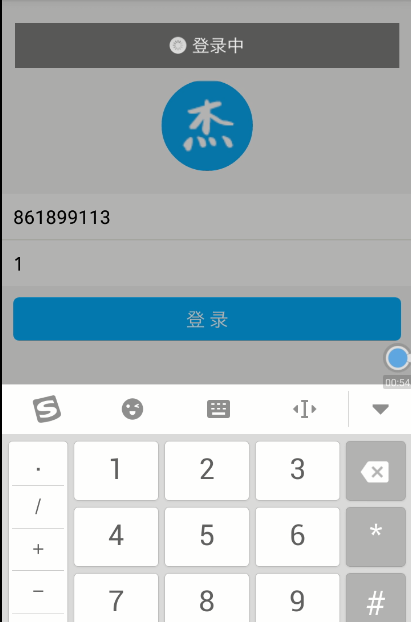


图4-3 登录截图

登陆成功之后进入主界面：



图4-图4-4 主界面

添加好友：

搜索好友：在指定文本框输入想要搜索的qq账号

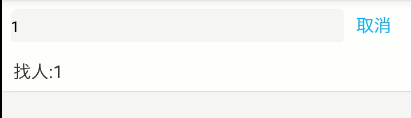


图4-5 输入qq账号

在输入账号后,会出现找人：1，意思为找qq号有1或者昵称有1的qq用户



图4-6 查找qq用户

再次输入可以再次筛选指定的qq用户：



添加好友：点击qq用户列表进入qq用户详细信息页面可以添加好友



图4-7 添加好友

点击完添加好友后，出现提示：

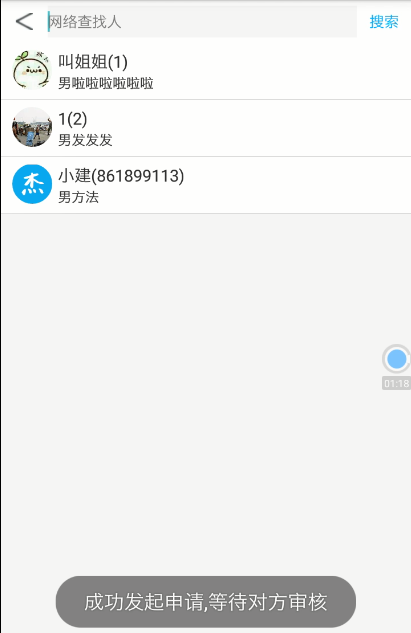


图4-8 成功发起好友申请

登录被添加好友的账号，主界面上会有1条好友信息提示：



图4-9 好友验证显示

点击新朋友图标，进入所有好友申请列表：

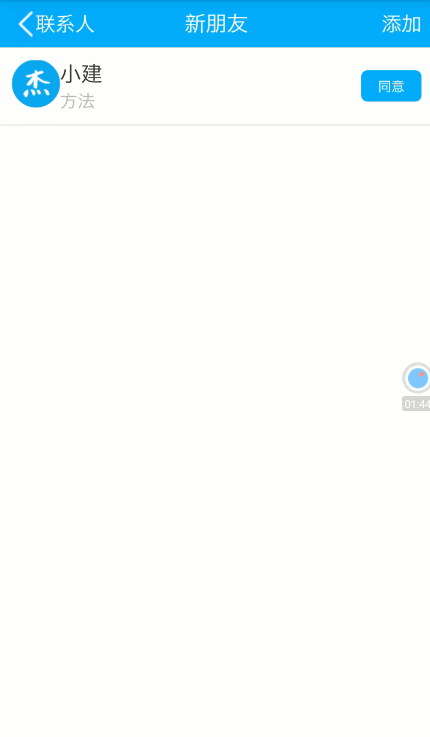


图4-10 好友申请列表

点开申请人信息，可以选择同意或者拒绝：

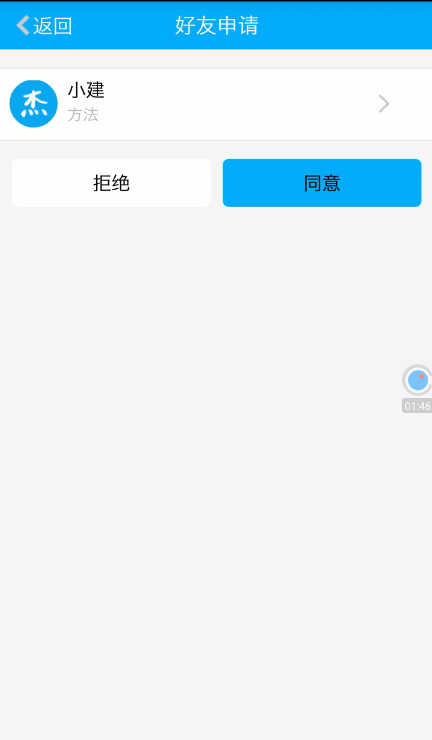


图4-11 同意或者拒绝

点击同意，在主界面的好友列表中就有了：



图4-12 成功添加好友

聊天功能：聊天是比较重要的一个功能，点开好友头像进入聊天界面，输入聊天消息，聊天消息将会保存在数据库中，等好友上线会收到，好友回复信息时自己也会收到。

点开好友头像进入聊天界面并发送消息：

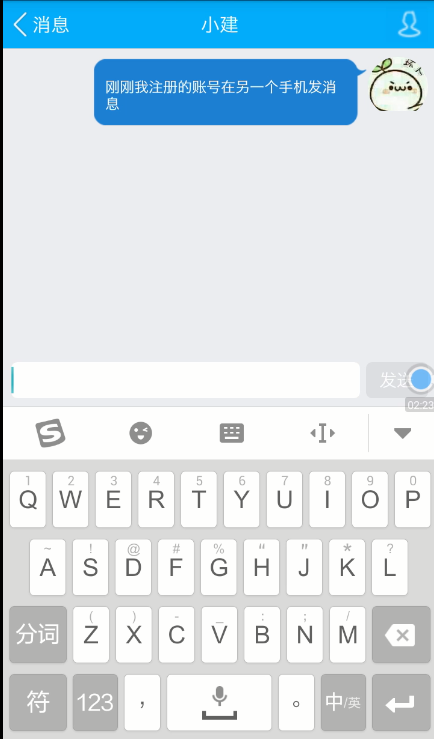


图4-13 聊天界面

等待好友回信息并显示：



图4-14聊天

好友动态功能：这功能需要实现能发表附带图片的说说，能查看好友发表的动态，并最新的动态显示在前面。

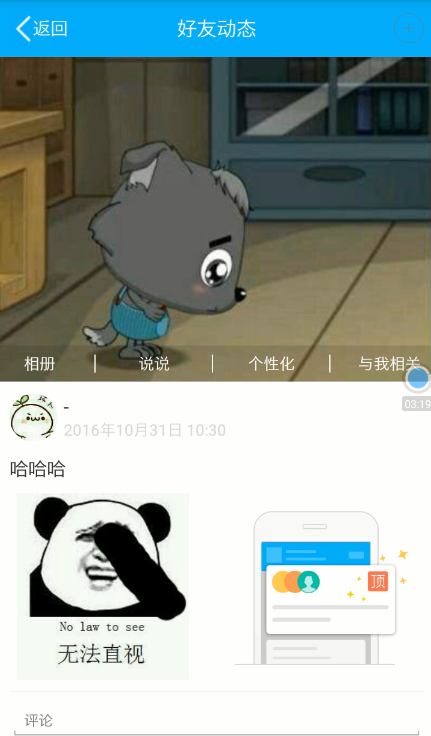


图4-15 好友动态界面

写说说：

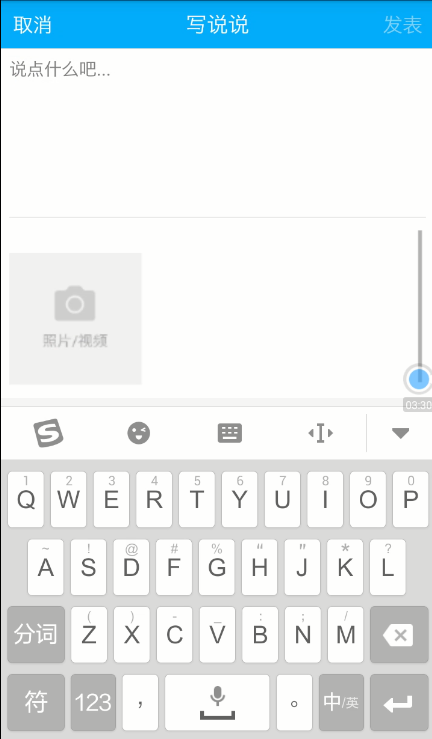


图4-16 写说说

输入文字和选择图片：

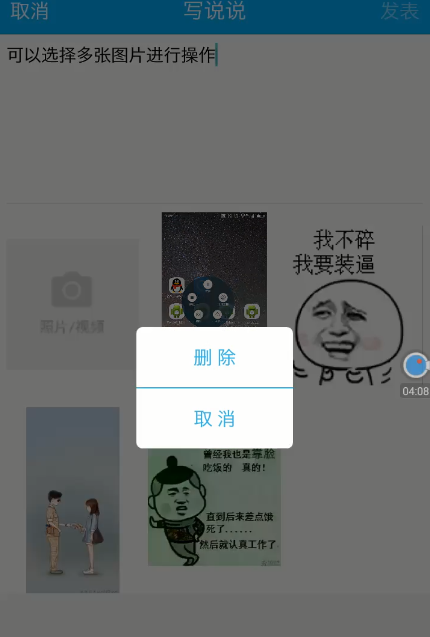


图4-17 删除说说选中的图片

选择的图片并且能删除，删除后发表：



图4-18 发表的说说

登录另一个账号查看是否有这个好友的动态并且发表最新动态：

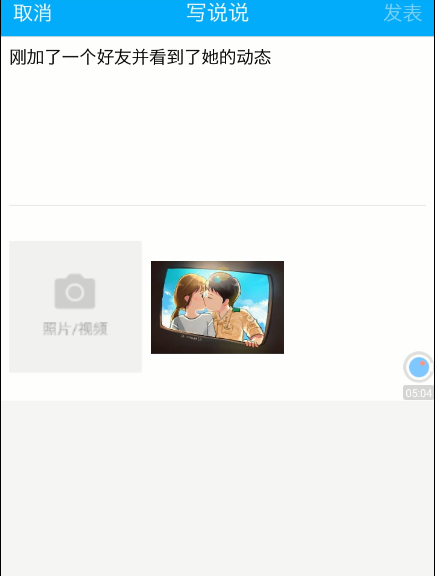


图4-19 查看好友说说



图4-20 自己也发了说说图

好友管理功能：能对自己的好友操作，进行分组，分成家人、好友、同学、朋友组，并且能删除好友

点击好友头像进入详细信息：



图4-21 好友信息

点击移动分组：



图4-22 移动分组

选择要分在家人这组项并点击完成：



图4-22 移动分组成功

再次点击好友头像进入详细信息：



图4-23 进行删除操作

点击删除好友：



图4-24 删除好友成功

### 4.1.3 功能概括和关键代码

一：添加好友 主要在NewFriendActivity.java文件中

在android\_qq\_hy表中，重要的是myqq\_id，与hyqq\_id，myqq\_id是发起添加好友的用户，hyqq\_id是被添加好友的用户。假设：小明向小花发送好友请求。

（1）在HyUserZiliaoActivity中小明发送申请，被添加者是小花，直接往数据库好友表里添加一行好友数据，其中myqq\_id是小明发起添加好友的人，hyqq\_id是小花被添加好友的人，但是状态为0，代表小花未同意。

**public** **void** addRen(AndroidQqUser hyuser){

AsyncHttpClient client = **new** AsyncHttpClient();// 开启网络请求对象类

String uri =URI\_IP.***URI***+"Android\_Service/QQ/becomeQqhy";//设置好请求的网络路径

RequestParams params = **new** RequestParams();// 要请求时附带的参数

// 具体数据信息

params.put("qqhy.myqqId",qquser.getQqId()+"");// 我的id

params.put("qqhy.myqqZhanghao",qquser.getQqZhanghao());

params.put("qqhy.myqqName",qquser.getQqName());

params.put("qqhy.myqqTouxiang",qquser.getQqTouxiang());

params.put("qqhy.myqqMark",qquser.getQqMark());

params.put("qqhy.myqqStatu",qquser.getQqStatu()+"");//在线还是下线

params.put("qqhy.myqqFengzu", 0+"");//默认好友

params.put("qqhy.hyqqId",hyuser.getQqId()+"");// 被添加好友的id

params.put("qqhy.hyqqZhanghao",hyuser.getQqZhanghao());

params.put("qqhy.hyqqName",hyuser.getQqName());

params.put("qqhy.hyqqTouxiang",hyuser.getQqTouxiang());

params.put("qqhy.hyqqMark",hyuser.getQqMark());

params.put("qqhy.hyqqStatu",hyuser.getQqStatu()+"");

params.put("qqhy.hyqqFengzu", 0+"");//默认分组在好友，等下移动

params.put("qqhy.hyStatu",0+"");

client.post(uri,params,**new** JsonHttpResponseHandler(){

@Override

**public** **void** onStart() {

}

@Override

**public** **void** onFailure(Throwable arg0, String arg1) {

//登录失败-网络问题

handler.sendEmptyMessage(2);

}

@Override

**public** **void** onSuccess(**int** arg0, JSONObject arg1) {

//如果Jsonobject获取的值不正确（jsonview）是不会继续下去的

**int** result=arg1.optInt("result");

**if**(result==1){// 当为1 代表在数据库中插入成功

handler.sendEmptyMessage(3);

}**else**{

handler.sendEmptyMessage(4);

}

}

@Override

**public** **void** onFinish() {

}

});

}

（2）在服务器端，具体操作数据库，往好友表添加一行数据的action

**public** String becomeQqhy(){//状态为0

**try**{

PrintWriter out = response.getWriter();

**boolean** bln=ss.addQqhy(qqhy);

**if**(bln){

result=**new** ResultFail(1);

}**else**{

result=**new** ResultFail(0);

}

out.print(JSON.*toJSONString*(result));

out.close();

}**catch**(Exception e){

}

**return** **null**;

}

（3）小花登录后点击同意好友，则将这行数据的状态改为1，代表同意。

**private** **int** hystatu;

**public** String UpdHaoYou(){

System.***out***.println(hyId);

System.***out***.println(hystatu);

**try**{

PrintWriter out = response.getWriter();

AndroidQqHy qqhys = ss.getQqhy(hyId);// 先通过主键查到是哪行数据

qqhys.setHyStatu((**long**)hystatu);// 查到后设置状态为传入的状态，同意还是拒绝

**boolean** bln=ss.updHaoYouStatu(qqhys);// 修改

System.***out***.println(bln);

**if**(bln){//成功失败

result = **new** ResultFail(1);

}**else**{

result = **new** ResultFail(0);

}

out.print(JSON.*toJSONString*(result));

out.close();

}**catch**(Exception e){

}

**return** **null**;

}

注意：小明与小花的好友关系在数据库只存在一条数据，重要概念是

当小明登录账号时， hyqq\_id是小花的id，说明小明的好友有小花。

当小花登录账号时，myqq\_id是小明的id，hyqq\_id是小花的id，这就要转换成，小花的好友有小明。

二：聊天 主要在ChartActivity.java文件中

在android\_qq\_message表中，qq\_id是发送消息的当前登录账号，m\_jsid是被发送消息的好友账号，与好友android\_qq\_hy表一样，发送者与接受者在不同账号中可以转换。

（1）在ChartActivity中小明对小花发送消息”小花你好”，发送者是小明，也就是在好友表中为myqq\_id的人，接受者是小花，也就是在好友表中为hyqq\_id的人。聊天界面上显示，右边是小明的头像和 ”小花你好”

**public** **void** sendMsgTodateOfHy(String str){

sendmsg=**new** AndroidQqMessage(); // 这个消息对象是为了发送消息后，直接在消息列表中，而不用在数据库中查找出来

//日期

Date date=**new** Date();//

AsyncHttpClient client = **new** AsyncHttpClient();// 发送网络请求对象

String uri =URI\_IP.***URI***+"Android\_Service/QQ/sendMessage";// 地址

RequestParams params = **new** RequestParams();// 参数组成

// qqname是当前聊天的对象就是好友的昵称，与qqhy是我与好友在数据库中存储的这一条数据

// 当好友的昵称在qqhy表中是被添加的好友，那么发送者是我，接受者是好友

**if**(qqname.equals(qqhy.getHyqqName())){

params.put("msg.qqId", qqhy.getMyqqId()+"");// 设置发送者id

params.put("msg.qqZhanghao", qqhy.getMyqqZhanghao());// 设置发送者qq账号

params.put("msg.qqName", qqhy.getMyqqName());// 设置发送者昵称

params.put("msg.qqTouxiang", qqhy.getMyqqTouxiang());// 设置发送者头像

params.put("msg.MJsid", qqhy.getHyqqId()+"");// 设置接受者id

params.put("msg.MZhanghao", qqhy.getHyqqZhanghao());// 设置接受者账号

params.put("msg.MName", qqhy.getHyqqName());// 设置接受者昵称

params.put("msg.MTouxiang", qqhy.getHyqqTouxiang());// 设置接受者头像

sendmsg.setQqId(qqhy.getMyqqId());

sendmsg.setQqZhanghao(qqhy.getMyqqZhanghao());

sendmsg.setQqName(qqhy.getMyqqName());

sendmsg.setQqTouxiang(qqhy.getMyqqTouxiang());

sendmsg.setMJsid(qqhy.getHyqqId());

sendmsg.setMZhanghao(qqhy.getHyqqZhanghao());

sendmsg.setMName(qqhy.getHyqqName());

sendmsg.setMTouxiang(qqhy.getHyqqTouxiang());

}**else**{// 当好友的昵称在qqhy表中是发起添加的我，那么发送者是好友，接受者是我

params.put("msg.qqId", qqhy.getHyqqId()+"");

params.put("msg.qqZhanghao", qqhy.getHyqqZhanghao());

params.put("msg.qqName", qqhy.getHyqqName());

params.put("msg.qqTouxiang", qqhy.getHyqqTouxiang());

params.put("msg.MJsid", qqhy.getMyqqId()+"");

params.put("msg.MZhanghao", qqhy.getMyqqZhanghao());

params.put("msg.MName", qqhy.getMyqqName());

params.put("msg.MTouxiang", qqhy.getMyqqTouxiang());

sendmsg.setQqId(qqhy.getHyqqId());

sendmsg.setQqZhanghao(qqhy.getHyqqZhanghao());

sendmsg.setQqName(qqhy.getHyqqName());

sendmsg.setQqTouxiang(qqhy.getHyqqTouxiang());

sendmsg.setMJsid(qqhy.getMyqqId());

sendmsg.setMZhanghao(qqhy.getMyqqZhanghao());

sendmsg.setMName(qqhy.getMyqqName());

sendmsg.setMTouxiang(qqhy.getMyqqTouxiang());

}

params.put("msg.MMessage", str);// 设置聊天消息

params.put("msg.MStatu", 0+"");

sendmsg.setMMessage(str);// 设置聊天消息

sendmsg.setMDate(date.getTime());

sendmsg.setMStatu((**long**)0);

client.post(uri,params,**new** JsonHttpResponseHandler(){

@Override

**public** **void** onStart() {

}

@Override

**public** **void** onFailure(Throwable arg0, String arg1) {

//登录失败-网络问题

handler.sendEmptyMessage(2);

}

@Override

**public** **void** onSuccess(**int** arg0, JSONObject arg1) {

**int** result=arg1.optInt("result");

**if**(result==1){// 发送是否成功

handler.sendEmptyMessage(1);

}**else**{

handler.sendEmptyMessage(0);

}

}

@Override

**public** **void** onFinish() {

}

});

}

（2）在服务器端真正的添加一行消息记录进入数据表中

**public** String sendMessage(){

**try**{

PrintWriter out = response.getWriter();

Date dd=**new** Date();

msg.setMDate(dd);

**boolean** bln = ss.addMsg(msg);

System.***out***.println(bln);

//返回成功等待图片取消 添加到list中

**if**(bln){

result = **new** ResultFail(1);//成功

}**else**{

result = **new** ResultFail(0);//失败 就弹出失败的哪个图片

}

out.print(JSON.*toJSONString*(result));

out.close();

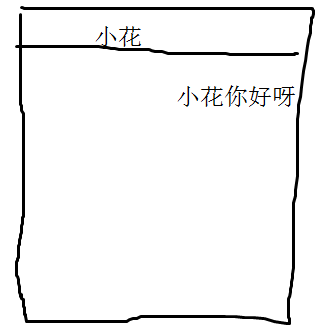
}**catch**(Exception e){

e.printStackTrace();

}

**return** **null**;

}



(3) 当在小花的手机上接受信息时候，查找有关于发送者是小明，接受者是小花或者发送者是小花，接受者是小明的信息行。

//在聊天界面时 获取的消息列表

**public** List getChartList(Long qqId,Long hyId){

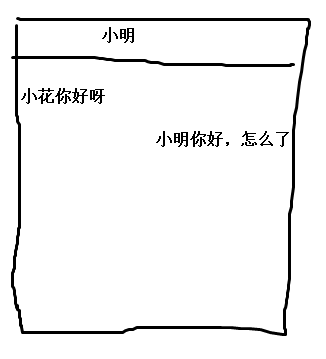
**return** listbysql("select m\_id from android\_qq\_message where (qq\_id="+qqId+" and m\_jsid="+hyId+") or (qq\_id="+hyId+" and m\_jsid="+qqId+") order by m\_date");

}

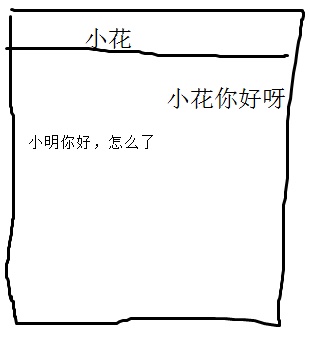
(4) 当在小花的手机上聊天界面中的消息显示，发送者是小明，接受者是小花，但是发送者不是当前登录者小花，于是这条信息”小花你好呀”会被放在左边。



(5)当小花回复信息给小明”小明你好，怎么了”，发送者是小花，接受者是小明，发送者是当前登录者小花，于是这条信息”小明你好，怎么了”会放在右边



(6)在小明手机里，接受到小花发来的信息”小明你好，怎么了”，这时候发送者是小花，接受者是小明，但是发送者不是当前登录者小明，于是这条信息会放在左边



聊天页面的显式代码在ChartBasadpter.java文件

（7）实时获取对方发来的信息是通过Handler来发送延迟信息查询数据库实现的。

handler.sendEmptyMessageDelayed(6, 12000);//一直查询

**private** Handler handler = **new** Handler(){

@Override

**public** **void** handleMessage(Message msg) {

**int** what = msg.what;

**switch** (what) {

**case** 0://发送消息失败

**break**;

**case** 1://发送成功添加到list

chartlist.add(sendmsg);

//刷新

notifylist();

//如果是第一条还要显示在消息列表中 和 我是发送者则显示 用广播和全局的msglist 用sendmsg和msglist比较

//checkifExit(sendmsg);

**break**;

**case** 2://网络

Toast.*makeText*(ChartActivity.**this**, "网络连接失败", 0).show();

**break**;

**case** 3:

//显示在聊天列表

notifylist();

**break**;

**case** 4:

Toast.*makeText*(ChartActivity.**this**, "网络连接失败", 0).show();

**break**;

**case** 5://无消息

handler.sendEmptyMessageDelayed(6, 12000);//一直查询

//new chartlist

chartlist = **new** ArrayList<AndroidQqMessage>();

**break**;

**case** 6:

getMsgOfdate();

**break**;

**default**:

**break**;

}

}

};

三：好友动态功能 主要文件DongTaiActivity.java

好友动态功能能查看好友发表的说说，最主要的就是查询自己好友的动态和根据时间来逆序排序，还有上传的图片能查看到。

(1)小明点击动态图标弹出新页面，里面有好友的发的说说。需要在后台查询出说说数据。并且要加上小明自己发的说说。

**public** String dynamic(){

**try**{

System.***out***.println(CurrPage);

System.***out***.println(PageCount);

PrintWriter out = response.getWriter();

//根据qq\_id查询好友 封装为Arraylist

List hylist = ss.selhy(qqId);//有3个字段，主要获取myqq\_id和hyqq\_id

//根据myqq\_id或者hyqq\_id查找动态 根据qqId筛选好友的

qqhyidlist = **new** ArrayList();//QQ好友id列表

**for**(**int** i=0;i<hylist.size();i++){

Map hymap = (Map)hylist.get(i);

Long hyqq\_id = Long.*parseLong*(hymap.get("hyqq\_id")+"");

Long myqq\_id = Long.*parseLong*(hymap.get("myqq\_id")+"");

**if**(hyqq\_id==qqId){//如果自己为好友

System.***out***.println("进来吧hyqq\_id");

qqhyidlist.add(myqq\_id);

}**else** **if**(myqq\_id==qqId){

System.***out***.println("进来吧myqq\_id");

qqhyidlist.add(hyqq\_id);

}

}

qqhyidlist.add(qqId);//加上我

List dynamiclist = ss.seldynamic(qqhyidlist,(CurrPage-1)\*PageCount,PageCount);//QQ好友动态 是开始第几个和结束的

List dynamiclists = **new** ArrayList<AndroidQqDynamic>(dynamiclist.size());//转换为dynamic对象list

**for**(**int** j=0;j<dynamiclist.size();j++){

Map map = (Map)dynamiclist.get(j);

Long dyId=Long.*parseLong*(map.get("dy\_id")+"");

AndroidQqDynamic dy = ss.getdynamic(dyId);

dynamiclists.add(dy);

}

qqdtlist = **new** AndroidQqDynamicList(1,dynamiclists);

out.print(JSON.*toJSONString*(qqdtlist));

out.close();

}**catch** (Exception e) {

}

**return** **null**;

}

(2)小明发说说，添加图片和输入文字点击发送后，图片文件将会保存在服务器中，而android\_qq\_dynamic中的dy\_photos字段保存的是图片文件保存在服务器的路径+小明qq号+文件名，多个图片的文件用逗号。发说说文字与图片上传到服务器是分开来的，图片有自己指定的action，表数据也有专门的action去执行。

a.上传文件到服务器指定位置

安卓端代码：

**public** **void** sendPhotos(){

**for**(**int** i=0;i<file.length;i++){// 选择的图片文件数组

**if**(file[i]!=**null**){

AsyncHttpClient client = **new** AsyncHttpClient();// 网络请求

String uri=URI\_IP.***URI***+"Android\_Service/QQ/saveDtPhotos";// 地址

RequestParams params = **new** RequestParams();// 组成参数

**try** {

params.put("file", file[i]);// 图片文件可以直接传入

} **catch** (Exception e) {

// **TODO** 自动生成的 catch 块

e.printStackTrace();

}

params.put("dynamicnamephoto",dynamicnamephoto[i]);//dynamicnamephoto是选择的图片名称

client.post(uri,params,**new** JsonHttpResponseHandler(){

@Override

**public** **void** onStart() {

}

@Override

**public** **void** onFailure(Throwable arg0, String arg1) {

//登录失败-网络问题

handler.sendEmptyMessage(2);

}

@Override

**public** **void** onSuccess(**int** arg0, JSONObject arg1) {

//如果Jsonobject获取的值不正确（jsonview）是不会继续下去的

**int** result=arg1.optInt("result");

**if**(result==1){

handler.sendEmptyMessage(3);

}**else**{

handler.sendEmptyMessage(4);

}

}

@Override

**public** **void** onFinish() {

}

});

}**else**{

i=file.length;

}

}

handler.sendEmptyMessage(3);

}

服务器代码：

**public** String saveDtPhotos(){

**try** {

PrintWriter out = response.getWriter();

System.***out***.println(file[0].getName());

// 使用文件输出流输出图片

FileOutputStream fos=**new** FileOutputStream(**this**.savePath+dynamicnamephoto[0]);

// 先将上传的文件转换为文件输入流

FileInputStream fil=**new** FileInputStream(**this**.file[0]);

// 每次读入 写出的大小。

// io流，就是先从文件输入流中读入1024字节，再用文件输出流输出1024字节到(this.savePath+dynamicnamephoto[0])这个文件夹里

**byte**[] b=**new** **byte**[1024];

**int** real=fil.read(b);

**int** ii=0;

**while**(real>0){

fos.write(b, 0, real);

real=fil.read(b);//每次读入 写出 文件对象就会少去这部分，所以这样1kb1kb写出 知道 读入不了了

ii++;

}

System.***out***.println(ii);

fos.close();fil.close();

result = **new** ResultFail(1);

out.print(JSON.*toJSONString*(result));

} **catch** (Exception e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**return** **null**;

}

B:上传说说一行表数据到action保存

安卓端代码：

**public** **void** add(){

Date date = **new** Date();

AsyncHttpClient client = **new** AsyncHttpClient();

String uri=URI\_IP.***URI***+"Android\_Service/QQ/adddynamic";

RequestParams params = **new** RequestParams();

params.put("qqdynamic.qqId", qquser.getQqId()+"");

params.put("qqdynamic.qqZhanghao", qquser.getQqZhanghao());

params.put("qqdynamic.qqName", qquser.getQqName());

params.put("qqdynamic.qqTouxiang", qquser.getQqTouxiang());

params.put("qqdynamic.dyDate",date.~~toLocaleString~~());

params.put("qqdynamic.dyContent",adddt\_rl2\_et1.getText().toString());

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

Object obj = dynamicnamephoto[i];

**if**(obj!=**null**){

dyPhotos+=dynamicnamephoto[i]+",";//数据库字符串

}**else**{

i=dynamicnamephoto.length;

}

}

params.put("qqdynamic.dyPhotos", dyPhotos+"");//多张图片用，号隔开 取也是。。。 这个文件名需要重新定义在组成这个

client.post(uri,params,**new** JsonHttpResponseHandler(){

@Override

**public** **void** onStart() {

}

@Override

**public** **void** onFailure(Throwable arg0, String arg1) {

//登录失败-网络问题

handler.sendEmptyMessage(2);

}

@Override

**public** **void** onSuccess(**int** arg0, JSONObject arg1) {

//如果Jsonobject获取的值不正确（jsonview）是不会继续下去的

**int** result=arg1.optInt("result");

**if**(result==1){

handler.sendEmptyMessage(1);

}**else**{

handler.sendEmptyMessage(0);

}

}

@Override

**public** **void** onFinish() {

}

});

}

服务端代码：

//上传的文件名称

**private** String[] dynamicnamephoto;

**public** String adddynamic(){

//返回json

**try** {

PrintWriter out = response.getWriter();

**boolean** bln= ss.addQqDt(qqdynamic);

System.***out***.println(bln);

//上传成功并重新获取刷新页面

**if**(bln){

result = **new** ResultFail(1);

}**else**{

result = **new** ResultFail(0);

}

out.print(JSON.*toJSONString*(result));

out.close();

} **catch** (IOException e) {

// **TODO** 自动生成的 catch 块

e.printStackTrace();

}

**return** **null**;

}

### **4.1.4** 服务器端的实现

服务器端功能

主控计算机处理各无线终端发送过来的数据信息并进行处理、加工（必要的情况下还进行信息回馈）以及发送，从而进行控制、结算、统计、结账打印以及管理等一系列职能操作。

服务端主要是数据库操作，针对无线终端各模块而设置响应相应功能的服务器小程序。主要的Servlet（如图显示）。

服务器端

响应头像图片的action

响应登录注册的action

响应好友列表的action

响应说说的action

响应发送消息的action

响应说说图片的action

响应添加好友的action

响应删除好友的action

响应添加好友的action

响应查询消息的action

图4-17 服务器功能图