KTV自助管理系统

关键代码分析说明书

编号：BDQN-LIB

版本：1.0

|  |  |  |  |
| --- | --- | --- | --- |
| **作者：** | 付鑫、况星、常国栋 | **日期：** | 2017-3-14 |
| **审批：** | 崔志文 | **日期：** | 2017-3-14 |

目 录

[1.前台端： 1](#_Toc485205355)

[1.1.1 动态添加包厢信息代码 1](#_Toc485205356)

[1.1.2 动态添加包厢信息分析 2](#_Toc485205357)

[2.前台端： 2](#_Toc485205358)

[2.1.1 网络连接代码 2](#_Toc485205359)

[2.1.2 网络连接分析 4](#_Toc485205360)

[2.2.1 数据和json转换代码 4](#_Toc485205361)

[2.2.2 数据和json转换分析 5](#_Toc485205362)

[3.手机端： 5](#_Toc485205363)

[3.1.1 网络连接代码 5](#_Toc485205364)

[3.1.2 网络连接分析 7](#_Toc485205365)

[3.2.1 数据与json之间转换代码 7](#_Toc485205366)

[3.3.2 数据与json之间转换分析 9](#_Toc485205367)

1.前台端：

1.1.1 动态添加包厢信息代码

/// <summary>

/// 动态添加包厢信息

/// </summary>

public void AddRomeBtn()

{

List<Room> roomList = roommanage.GetRoomList();

int weight = 200;//横坐标

int height = 0;//纵坐标

for(int i = 0,j = 0;i < roomList.Count;i++,j++)

{

Button btn = new Button();

if(j \* weight + 200 > this.Size.Width - 75)//如果横坐标超过屏幕大小，纵坐标加100，横坐标清空

{

height += 100;

j = 0;

}

if(roomList[i].R\_RoomState1 == 1)

{

btn.Text = roomList[i].R\_RoomId1 + " 使用中 " + roomList[i].R\_RoomType1;

btn.BackColor = Color.Red;

}

else if (roomList[i].R\_RoomState1 == 0)

{

btn.Text = roomList[i].R\_RoomId1 + " 空闲中 " + roomList[i].R\_RoomType1;

btn.BackColor = Color.Green;

}

btn.Name = "btnRoom" + roomList[i].R\_RoomId1;

btn.Size = new Size(200, 100);

btn.Location = new Point(j \* weight, height);

List<string> taglist = new List<string>();

taglist.Add(roomList[i].R\_RoomId1.ToString());

taglist.Add("");

btn.Tag = taglist;

btn.Click += new System.EventHandler(btn\_Click);

this.Controls.Add(btn);//将创建好的btn添加到窗体

}

}

1.1.2 动态添加包厢信息分析

根据包厢数量动态判断位置和信息，生成按钮控件。

2.前台端：

2.1.1 网络连接代码

/// <summary>

/// 网络连接工具类

/// </summary>

IPAddress ip;//ip地址

IPEndPoint point;//带端口号的ip地址

Socket socketWatch;//用于监听的socket

Socket socketSend;//用于发送的socket

Thread listenThread;//用于监听的线程

/// <summary>

/// 监听

/// </summary>

public void Listen()

{

//在服务器端创建一个负责监听IP地址和端口号的Socket

socketWatch = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.Tcp);

foreach (IPAddress \_IPAddress in Dns.GetHostEntry(Dns.GetHostName()).AddressList)

{

if (\_IPAddress.AddressFamily.ToString() == "InterNetwork")

{

ip = \_IPAddress;//获取本机IP地址

//Console.WriteLine(ip.ToString());

}

}

//ip = IPAddress.Parse("192.168.1.110");//创建ip地址对象

point = new IPEndPoint(ip, 8998);//创建包含端口号的ip地址对象

socketWatch.Bind(point);//绑定包含端口的ip到监听socket

socketWatch.Listen(10);//开始监听，最大监听10个设备

while (true)

{

socketSend = socketWatch.Accept();//等待客户端的连接，并创建一个用于通讯的socket

Console.WriteLine(socketSend.RemoteEndPoint + "设备已连接到本服务器");

listenThread = new Thread(new ParameterizedThreadStart(GetClientMessage));//创建负责监听的子线程

listenThread.IsBackground = true;//设置该子线程为后台线程

listenThread.Start(socketSend);//启动线程传入监听socket

}

}

/// <summary>

/// 获得客户端的消息

/// </summary>

/// <param name="obj"></param>

private void GetClientMessage(Object obj)

{

Socket socket = obj as Socket;//转换数据类型

while (true)

{

try

{

Byte[] buffer = new Byte[2048];//接收信息的字节数组

int count = socket.Receive(buffer);//判断客户端连接状态

if (count == 0)

{

Console.WriteLine(socket.RemoteEndPoint + "断开连接");

break;//客户端断开时跳出

}

else

{

String str = Encoding.UTF8.GetString(buffer, 0, count);//获得客户端发送的信息

Console.WriteLine(socket.RemoteEndPoint + "发送消息：" + str);

ReturnJson(str, socket);

}

}

catch (Exception)

{

}

}

}

2.1.2 网络连接分析

使用socket和安卓端进行通讯，发送和接收消息。

2.2.1 数据和json转换代码

/// <summary>

/// 获得Json字符串，将数据转换为json字符串以便发送

/// </summary>

/// <param name="obj"></param>

/// <returns></returns>

private byte[] GetJson(Object obj)

{

StringBuilder jsonStr = new StringBuilder();

if (obj is List<SongInfo>)

{

jsonStr.Clear();

jsonStr.Append("{");

int i = 0;

foreach(SongInfo song in obj as List<SongInfo>)

{

i++;

jsonStr.Append("\""+song.SGI\_SongId1 + "\":{");

jsonStr.Append("\"name\":\"" + song.SGI\_SongName1 + "\",");

jsonStr.Append("\"type\":\"" + song.SGI\_SongType1 + "\",");

jsonStr.Append("\"count\":\"" + song.SGI\_SongWordCount1 + "\",");

jsonStr.Append("\"spell\":\"" + song.SGI\_SongSpell1 + "\",");

jsonStr.Append("\"singer\":\"" + song.SGI\_SongSinger1 + "\",");

jsonStr.Append("\"click\":\"" + song.SGI\_SongClickRate1 + "\"");

jsonStr.Append("}");

if (i != (obj as List<SongInfo>).Count)

{

jsonStr.Append(",");

}

}

jsonStr.Append("}");

return Encoding.UTF8.GetBytes(jsonStr.ToString());

}

else if (obj is List<SingerInfo>)

{

jsonStr.Clear();

jsonStr.Append("{");

int i = 0;

foreach (SingerInfo singer in obj as List<SingerInfo>)

{

i++;

jsonStr.Append("\"" + singer.SRI\_SingeId1 + "\":{");

jsonStr.Append("\"name\":\"" + singer.SRI\_SingerName1 + "\",");

jsonStr.Append("\"type\":\"" + singer.SRI\_SingeType1 + "\"");

jsonStr.Append("}");

if (i != (obj as List<SingerInfo>).Count)

{

jsonStr.Append(",");

}

}

jsonStr.Append("}");

return Encoding.UTF8.GetBytes(jsonStr.ToString());

}

return null;

}

2.2.2 数据和json转换分析

把数据转换为json字符串发送到安卓端。

3.手机端：

3.1.1 网络连接代码

private PrintWriter writer; // 输出对象

DataInputStream dis;// 读取流

Socket socket;// 发送socket

byte[] Reader = new byte[618587];

/\*\*

\* 连接

\*

\* @param v

\*/

public boolean ConnTest(final String ip, final int post) {

new Thread(new Runnable() {// 创建连接线程

public void run() {

try {

// System.out.println("try connection!");

android.util.Log.e("SendMessage", "try connection");

socket = new Socket(ip, post);// 创建连接到服务器的socket

// System.out.println("connectioned!");

android.util.Log.e("SendMessage", "connectioned");

} catch (Exception e) {

return;

}

}

}).start();

return true;

}

/\*\*

\* 发送信息

\*

\* @param message

\* @return

\*/

public byte[] SendMessage(String message) {

try {

writer = new PrintWriter(socket.getOutputStream(), true);// 将printwriter绑定到socket

writer.println(message);

writer.flush();

dis = new DataInputStream(socket.getInputStream());// 绑定输入流到DataInputStream对象

Reader = new byte[618587];

new Thread(new Runnable() {

public void run() {

try {

while (true) {

dis.read(Reader);// 读取数据

final String str = new String(Reader);

if (str.trim() != "" && str.trim() != null) {// 如果获取到反馈

//System.out.println("读到数据" + str);

break;

}

}

} catch (IOException e) {

System.out.println(e);

}

}

}).start();

Thread.sleep(100);// 等待子线程读取到值

while (true) {// 不断刷新读取

final String str = new String(Reader);

// Thread.sleep(100);

if (str.trim() != "" && str.trim() != null) {// 如果获取到反馈

// System.out.println("返回数据");

//System.out.println(str);

android.util.Log.e("SendMessage", new String(Reader).trim());

return Reader;// 返回收到的数据

}

}

} catch (Exception e) {

System.out.println(e);

return "Exception".getBytes();

}

}

3.1.2 网络连接分析

通过socket和服务器进行数据传递，用于接受和发送数据。

3.2.1 数据与json之间转换代码

/\*\*

\* 从json字符串中获得歌手数据集合

\*

\*/

public List<Singer> GetSingerOnJson(byte[] buff){

List<Singer> singerList = new ArrayList<Singer>();

try {

String string = new String(buff);

JSONObject jsonObject = new JSONObject(string);

Iterator iterator = jsonObject.keys();

while(iterator.hasNext()){

String key = (String) iterator.next();

JSONObject singerJson = jsonObject.getJSONObject(key);

Singer singer = new Singer();

singer.setSRI\_SingeId(Integer.parseInt(key));

singer.setSRI\_SingerName(singerJson.getString("name"));

singer.setSRI\_SingeType(singerJson.getString("type"));

//singer.setSRI\_SingePhoto();歌手照片

singerList.add(singer);

}

return singerList;

} catch (JSONException e) {

System.out.println(e);

}

return singerList;

}

/\*\*

\* 从json字符串中获得歌曲数据集合

\*

\*/

public List<Song> GetSongOnJson(byte[] buff){

List<Song> songList = new ArrayList<Song>();

try {

String string = new String(buff);

JSONObject jsonObject = new JSONObject(string);

Iterator iterator = jsonObject.keys();

while(iterator.hasNext()){

String key = (String) iterator.next();

JSONObject songJson = jsonObject.getJSONObject(key);

Song song = new Song();

song.setSGI\_SongId(Integer.parseInt(key));

song.setSGI\_SongName(songJson.getString("name"));

song.setSRI\_SingerName(songJson.getString("singer"));

songList.add(song);

}

return songList;

} catch (JSONException e) {

return null;

}

}

3.3.2 数据与json之间转换分析

接受服务器传来的json数据并转换为集合。