# Socket程序设计文档

By shinepans

# 设计部分:

服务器端界面设计:

功能需求: 消息发送到指定的客户端

记录发送的消息到数据库

可以查询聊天历史,从数据库中调取

界面友好

**服务器端界面设计**

服务器端

IP Port

开始服务

发送

聊天记录查询

聊天记录

服务器端功能:

监听指定的IP 端口

显示聊天记录

记录数据到数据库

**客户端界面设计**

服务器端

IP Port

连接

发送

聊天记录

功能与服务器相同,但没有连天记录功能(安全性).

# 软件设计环境

Windows 8.1 x64 vs 2013 Sqlserver 2008 R2 Office

# 完成人员

潘尚

# 开发步骤

1. 了解socket原理,tcp原理
2. 学习socket类的使用
3. 开发服务器端与客户端
4. 建立数据库
5. 建立程序与数据库的连接
6. 经过不断调试改善程序
7. 完善数据库
8. 美化界面
9. 软件测试
10. 发布软件

在使用字符串的时候,集中规定消息发送方的名称,

比如有

系统消息 用户1 用户2 消息

以及程序所做的相应的事 比如 服务器监听 客户端连接 消息都会有记录,可以在数据库中查询

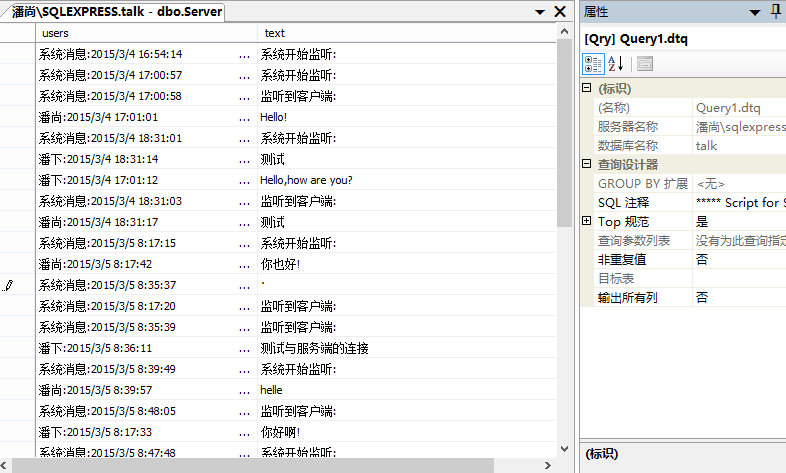
考虑到使用的方便,将分离系统消息 与用户消息

# 数据库:

建立了两列数据

一个用来存放用户与时间

另一个用来存放聊天记录以及系统消息



# 关键代码

**数据库连接:**

socConnection = socketWatch.Accept();

txtMsg.AppendText("Got Client:" + "\r\n");

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("insert into Server values('" + sys + GetCurrentTime() + "','" + sys2 + "')", myConn);

myConn.Open();

int i = cmd.ExecuteNonQuery();

if (i > 0)

{

}

else

{

MessageBox.Show("数据库出问题了!");

}

**聊天记录存放数据库:**

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("insert into Server values('" +user1+GetCurrentTime() + "','" + sendMsg.Trim() + "')", myConn);

myConn.Open();

int i = cmd.ExecuteNonQuery();

**套接字:**

Socket socketServer = socketClientPara as Socket;

while (true)

{

//创建一个内存缓冲区 其大小为1024\*1024字节 即1M

byte[] arrServerRecMsg = new byte[1024 \* 1024];

//将接收到的信息存入到内存缓冲区,并返回其字节数组的长度

int length = 0;

try

{

length = socketServer.Receive(arrServerRecMsg);

}

catch(Exception EX)

{

numOfUser -= 1;

break;

**显示聊天记录:**

// TODO: 这行代码将数据加载到表“talkDataSet1.Server”中。您可以根据需要移动或删除它。

this.serverTableAdapter.Fill(this.talkDataSet1.Server);

// TODO: 这行代码将数据加载到表“talkDataSet.talk”中。您可以根据需要移动或删除它。

this.talkTableAdapter.Fill(this.talkDataSet.talk);

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("select \* from Server", myConn);

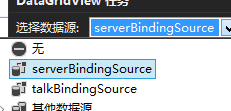
SqlDataAdapter sda = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

sda.Fill(ds, "Server");

dataGridView1.DataSource = ds.Tables["Server"];

**数据源:**



**监听:**

//套接字用于监听客户端发来的信息 包含3个参数(IP4寻址协议,流式连接,TCP协议)

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

//服务端发送信息 需要1个IP地址和端口号

IPAddress ipaddress = IPAddress.Parse(txtIP.Text.Trim()); //获取文本框输入的IP地址

//将IP地址和端口号绑定到网络节点endpoint上

IPEndPoint endpoint = new IPEndPoint(ipaddress, int.Parse(txtPORT.Text.Trim())); //获取文本框上输入的端口号

//监听绑定的网络节点

socketWatch.Bind(endpoint);

//将套接字的监听队列长度限制为20

socketWatch.Listen(20);

//创建一个监听线程

threadWatch = new Thread(WatchConnecting);

threadWatch.IsBackground = true; //窗体程序与后台程序同步

threadWatch.Start();

txtMsg.AppendText("Server Start Listen :" +"\r\n"+ GetCurrentTime()+ "\r\n"); //将信息显示到RichTextBox

//将日志加入到数据库中 ,数据库为本地文件,在我的文档中,连接方式见下

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("insert into Server values('" +sys+GetCurrentTime()+ "','" +sys1 + "')",myConn);

myConn.Open();

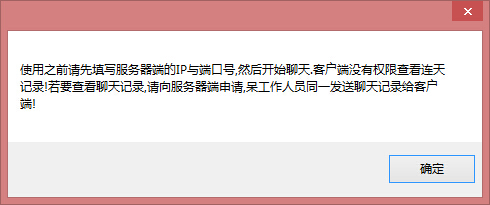
int i = cmd.ExecuteNonQuery();

# 测试截图

关于:



帮助:



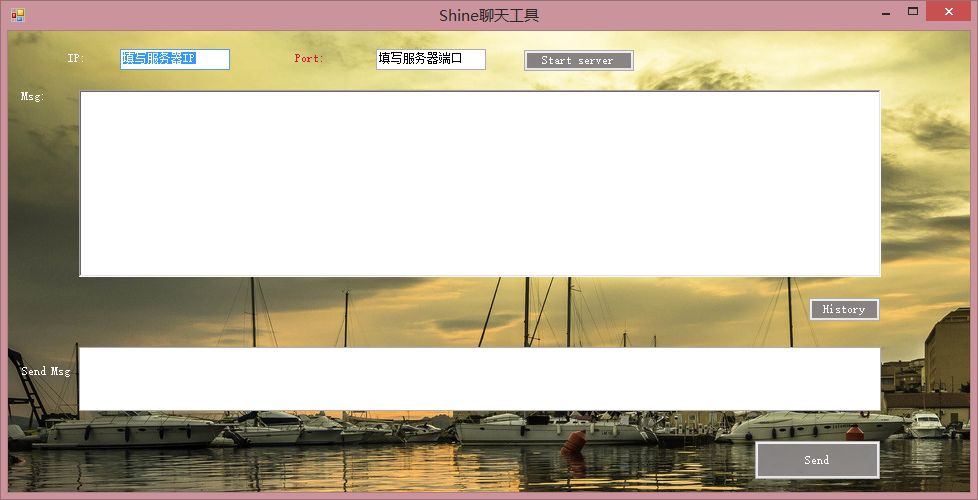
登录:



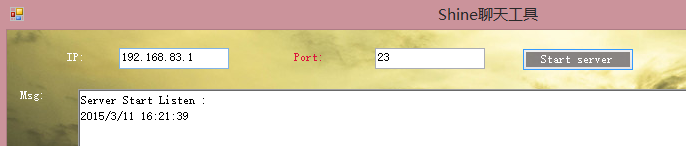
客户端:



服务端:



服务器开始监听:



客户端连接:



聊天:



聊天记录:



# 详细代码

**客户端:**

Form2.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Net.Sockets;

using System.Threading;

using System.Net;

namespace SocketClient

{

public partial class Form2 : Form

{

public Form2()

{

InitializeComponent();

//关闭对文本框的非法线程操作检查

TextBox.CheckForIllegalCrossThreadCalls = false;

}

//创建 1个客户端套接字 和1个负责监听服务端请求的线程

Socket socketClient = null;

Thread threadClient = null;

public string user;

private void btnBeginListen\_Click(object sender, EventArgs e)

{

//定义一个套字节监听 包含3个参数(IP4寻址协议,流式连接,TCP协议)

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

//需要获取文本框中的IP地址

IPAddress ipaddress = IPAddress.Parse(txtIP.Text.Trim());

//将获取的ip地址和端口号绑定到网络节点endpoint上

IPEndPoint endpoint = new IPEndPoint(ipaddress, int.Parse(txtPORT.Text.Trim()));

//这里客户端套接字连接到网络节点(服务端)用的方法是Connect 而不是Bind

socketClient.Connect(endpoint);

//创建一个线程 用于监听服务端发来的消息

threadClient = new Thread(RecMsg);

//将窗体线程设置为与后台同步

threadClient.IsBackground = true;

//启动线程

threadClient.Start();

}

/// <summary>

/// 接收服务端发来信息的方法

/// </summary>

private void RecMsg() //接受信息线程使用的函数

{

while (true) //持续监听服务端发来的消息

{

//定义一个1M的内存缓冲区 用于临时性存储接收到的信息

byte[] arrRecMsg = new byte[1024 \* 1024];

//将客户端套接字接收到的数据存入内存缓冲区, 并获取其长度

int length = 0;

try

{

length = socketClient.Receive(arrRecMsg);

}

catch(Exception Ex)

{

break;

}

//将套接字获取到的字节数组转换为人可以看懂的字符串

string strRecMsg = Encoding.UTF8.GetString(arrRecMsg, 0, length);

//将发送的信息追加到聊天内容文本框中

txtMsg.AppendText("来自服务器:" + GetCurrentTime() + "\r\n" + strRecMsg + "\r\n");

}

}

/// <summary>

/// 发送字符串信息到服务端的方法

/// </summary>

/// <param name="sendMsg">发送的字符串信息</param>

private void ClientSendMsg(string sendMsg)

{

//将输入的内容字符串转换为机器可以识别的字节数组

byte[] arrClientSendMsg = Encoding.UTF8.GetBytes(sendMsg);

//调用客户端套接字发送字节数组

socketClient.Send(arrClientSendMsg);

//将发送的信息追加到聊天内容文本框中

txtMsg.AppendText(user+"\r\n" +":"+"\r\n"+ GetCurrentTime() + "\r\n" + sendMsg + "\r\n");

}

//点击按钮btnSend 向服务端发送信息

/// <summary>

/// 获取当前系统时间的方法

/// </summary>

/// <returns>当前时间</returns>

private DateTime GetCurrentTime()

{

DateTime currentTime = new DateTime();

currentTime = DateTime.Now;

return currentTime;

}

private void btnConnect\_Click(object sender, EventArgs e)

{

//定义一个套字节监听 包含3个参数(IP4寻址协议,流式连接,TCP协议)

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

//需要获取文本框中的IP地址

IPAddress ipaddress = IPAddress.Parse(txtIP.Text.Trim());

//将获取的ip地址和端口号绑定到网络节点endpoint上

IPEndPoint endpoint = new IPEndPoint(ipaddress, int.Parse(txtPORT.Text.Trim()));

//这里客户端套接字连接到网络节点(服务端)用的方法是Connect

socketClient.Connect(endpoint);

//创建一个线程 用于监听服务端发来的消息

threadClient = new Thread(RecMsg);

//将窗体线程设置为与后台同步

threadClient.IsBackground = true;

//启动线程

threadClient.Start();

txtMsg.AppendText("Got Server:" + GetCurrentTime() + "\r\n");

}

private void btnSent\_Click(object sender, EventArgs e)

{

ClientSendMsg(user+": "+txtSend.Text.Trim());

txtSend.Text = "";

}

private void btnHistory\_Click(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

MessageBox.Show("这款软件由潘尚制作完成,主要功能有:聊天,聊天记录查询,聊天记录查询请向服务器端申请.联系作者:潘尚,QQ:574273250");

}

private void button2\_Click(object sender, EventArgs e)

{

MessageBox.Show("使用之前请先填写服务器端的IP与端口号,然后开始聊天.客户端没有权限查看连天记录!若要查看聊天记录,请向服务器端申请,呆工作人员同一发送聊天记录给客户端!");

}

private void txtIP\_MouseClick(object sender, MouseEventArgs e)

{

txtIP.Text = "";

}

private void txtPORT\_MouseClick(object sender, MouseEventArgs e)

{

txtPORT.Text = "";

}

private void Form2\_Load(object sender, EventArgs e)

{

}

private void txtSend\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

//则调用 服务器向客户端发送信息的方法

ClientSendMsg(txtSend.Text.Trim());

txtSend.Text = "";

}

}

private void txtSend\_MouseClick(object sender, MouseEventArgs e)

{

}

}

}

Log.cs

using SocketClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace SocektClient

{

public partial class Logo : Form

{

public Logo()

{

InitializeComponent();

}

private void btnLog\_Click(object sender, EventArgs e)

{

Form2 fm = new Form2();

fm.user = textBox1.Text.ToString();

fm.Show();

this.Hide();

}

private void textBox1\_MouseClick(object sender, MouseEventArgs e)

{

textBox1.Text = "";

}

private void textBox1\_KeyDown(object sender, KeyEventArgs e)

{

Form2 fm = new Form2();

fm.user = textBox1.Text;

Logo lg = new Logo();

lg.Close();

}

}

}

Program.cs

using SocketClient;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace SocektClient

{

static class Program

{

/// <summary>

/// 应用程序的主入口点。

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Logo());

}

}

}

**服务器端:**

Form1.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Threading;

using System.Net.Sockets;

using System.Net;

using System.Data.SqlClient;

namespace SocketServer

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

TextBox.CheckForIllegalCrossThreadCalls = false; //关闭文本框非法线程的检测

}

string dataDir = AppDomain.CurrentDomain.BaseDirectory;

Thread threadWatch = null; //负责监听客户端的线程

Socket socketWatch = null; //负责监听客户端的套接字

string Conn = "server=潘尚\\SQLEXPRESS;database=Server;uid=sa;pwd=123";

string sys = "系统消息:";

string sys1 = "系统开始监听:";

string sys2 = "监听到客户端:";

string user1 = "潘尚:";

private void btnServerConn\_Click(object sender, EventArgs e)

{

//套接字用于监听客户端发来的信息 包含3个参数(IP4寻址协议,流式连接,TCP协议)

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

//服务端发送信息 需要1个IP地址和端口号

IPAddress ipaddress = IPAddress.Parse(txtIP.Text.Trim()); //获取文本框输入的IP地址

//将IP地址和端口号绑定到网络节点endpoint上

IPEndPoint endpoint = new IPEndPoint(ipaddress, int.Parse(txtPORT.Text.Trim())); //获取文本框上输入的端口号

//监听绑定的网络节点

socketWatch.Bind(endpoint);

//将套接字的监听队列长度限制为20

socketWatch.Listen(20);

//创建一个监听线程

threadWatch = new Thread(WatchConnecting);

threadWatch.IsBackground = true; //窗体程序与后台程序同步

threadWatch.Start();

txtMsg.AppendText("Server Start Listen :" +"\r\n"+ GetCurrentTime()+ "\r\n"); //将信息显示到RichTextBox

//将日志加入到数据库中 ,数据库为本地文件,在我的文档中,连接方式见下

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("insert into Server values('" +sys+GetCurrentTime()+ "','" +sys1 + "')",myConn);

myConn.Open();

int i = cmd.ExecuteNonQuery();

if(i>0)

{

}

else

{

MessageBox.Show("数据库出问题了!");

}

}

Socket socConnection = null; //负责和客户端通通信的套接字

/// <summary>

/// 监听客户端发来的请求

/// </summary>

private void WatchConnecting()

{

while (true) //持续不断监听客户端发来的请求

{

socConnection = socketWatch.Accept();

txtMsg.AppendText("Got Client:" + "\r\n");

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("insert into Server values('" + sys + GetCurrentTime() + "','" + sys2 + "')", myConn);

myConn.Open();

int i = cmd.ExecuteNonQuery();

if (i > 0)

{

}

else

{

MessageBox.Show("数据库出问题了!");

}

//创建一个通信线程

ParameterizedThreadStart pts = new ParameterizedThreadStart(ServerRecMsg);

Thread thr = new Thread(pts);

thr.IsBackground = true;

//启动线程

numOfUser += 1;

thr.Start(socConnection);

}

}

/// <summary>

/// 发送信息到客户端的方法

/// </summary>

/// <param name="sendMsg">发送的字符串信息</param>

private void ServerSendMsg(string sendMsg)

{

//将输入的字符串转换成 机器可以识别的字节数组

byte[] arrSendMsg = Encoding.UTF8.GetBytes(sendMsg);

//向客户端发送字节数组信息

socConnection.Send(arrSendMsg);

//将发送的字符串信息附加到文本框txtMsg上

txtMsg.AppendText("潘尚:" + GetCurrentTime() + "\r\n" + sendMsg + "\r\n");

//发送消息后,清空发送栏

//将消息存入数据库

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("insert into Server values('" +user1+GetCurrentTime() + "','" + sendMsg.Trim() + "')", myConn);

myConn.Open();

int i = cmd.ExecuteNonQuery();

if (i > 0)

{

}

else

{

MessageBox.Show("数据库出问题了!");

}

}

public int numOfUser = 0;

/// <summary>

/// 接收客户端发来的信息

/// </summary>

/// <param name="socketClientPara">客户端套接字对象</param>

private void ServerRecMsg(object socketClientPara)

{

Socket socketServer = socketClientPara as Socket;

while (true)

{

//创建一个内存缓冲区 其大小为1024\*1024字节 即1M

byte[] arrServerRecMsg = new byte[1024 \* 1024];

//将接收到的信息存入到内存缓冲区,并返回其字节数组的长度

int length = 0;

try

{

length = socketServer.Receive(arrServerRecMsg);

}

catch(Exception EX)

{

numOfUser -= 1;

break;

//减少当前在线人数

}

//将机器接受到的字节数组转换为人可以读懂的字符串

string strSRecMsg = Encoding.UTF8.GetString(arrServerRecMsg, 0, length);

//将发送的字符串信息附加到文本框txtMsg上

txtMsg.AppendText("来自用户" + GetCurrentTime() + "\r\n" + strSRecMsg + "\r\n");

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("insert into Server values('" + "用户:" + GetCurrentTime() + "','" + strSRecMsg + "')", myConn);

myConn.Open();

int i = cmd.ExecuteNonQuery();

if (i > 0)

{

}

else

{

MessageBox.Show("数据库出问题了!");

}

}

}

//发送信息到客户端

private void btnSendMsg\_Click(object sender, EventArgs e)

{

//调用 ServerSendMsg方法 发送信息到客户端

ServerSendMsg(txtSendMsg.Text.Trim());

//清空发送消息栏

txtSendMsg.Text = "";

}

//快捷键 Enter 发送信息

private void txtSendMsg\_KeyDown(object sender, KeyEventArgs e)

{

//如果用户按下了Enter键

if (e.KeyCode == Keys.Enter)

{

//则调用 服务器向客户端发送信息的方法

ServerSendMsg(txtSendMsg.Text.Trim());

}

}

/// <summary>

/// 获取当前系统时间的方法

/// </summary>

/// <returns>当前时间</returns>

private DateTime GetCurrentTime()

{

DateTime currentTime = new DateTime();

currentTime = DateTime.Now;

return currentTime;

}

private void btnHistory\_Click(object sender, EventArgs e)

{

HForm hf = new HForm();

hf.Show();

}

private void txtSendMsg\_TextChanged(object sender, EventArgs e)

{

}

private void Form1\_Load(object sender, EventArgs e)

{

/\* 本来想使用VS2013内置的mdf,但是受技术限制,还是使用了SQL 2008的版本的数据库

if (dataDir.EndsWith(@"\bin\Debug") || dataDir.EndsWith(@"\bin\Release"))

{

dataDir = System.IO.Directory.GetParent(dataDir).Parent.FullName;

AppDomain.CurrentDomain.SetData("DataDirectory", dataDir);

}

SqlConnection conn = new SqlConnection("DataSource=(LocalDB)\\v11.0;AttachDbFilename=d:\\用户目录\\我的文档\\visual studio 2013\\Projects\\SocketServer\\SocketServer\\Database1.mdf;Integrated Security=True");

conn.Open();

txtMsg.AppendText("已连接至数据库" + GetCurrentTime() + "\r\n");

\* \*/

string myConn = "server=潘尚\\SQLEXPRESS;database=talk;uid=sa;pwd=123";

SqlConnection sc = new SqlConnection();

sc.ConnectionString = myConn;

try

{

sc.Open();

MessageBox.Show("已与数据库建立连接");

}

catch(Exception ex)

{

MessageBox.Show("打开数据库错误{0}", ex.Message);

}

}

private void txtIP\_MouseClick(object sender, MouseEventArgs e)

{

txtIP.Text = "";

}

private void txtPORT\_MouseClick(object sender, MouseEventArgs e)

{

txtPORT.Text = "";

}

private void txtSendMsg\_KeyDown\_1(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

{

//则调用 服务器向客户端发送信息的方法

ServerSendMsg(txtSendMsg.Text.Trim());

txtSendMsg.Text = "";

}

}

}

}

Hform.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace SocketServer

{

public partial class HForm : Form

{

public HForm()

{

InitializeComponent();

}

private void HForm\_Load(object sender, EventArgs e)

{

// TODO: 这行代码将数据加载到表“talkDataSet1.Server”中。您可以根据需要移动或删除它。

this.serverTableAdapter.Fill(this.talkDataSet1.Server);

// TODO: 这行代码将数据加载到表“talkDataSet.talk”中。您可以根据需要移动或删除它。

this.talkTableAdapter.Fill(this.talkDataSet.talk);

SqlConnection myConn = new SqlConnection("server=潘尚\\sqlexpress;database=talk;uid=sa;pwd=123");

SqlCommand cmd = new SqlCommand("select \* from Server", myConn);

SqlDataAdapter sda = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

sda.Fill(ds, "Server");

dataGridView1.DataSource = ds.Tables["Server"];

}

}

}

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace SocketServer

{

static class Program

{

/// <summary>

/// 应用程序的主入口点。

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

2015-3-5