江南大学

JIANGNAN UNIVERSITY

物联网工程学院

TCP/IP 课程实验报告

班　　级：物联 1601

姓　　名：尹达恒

学　　号：1030616134

指导老师：马君霞

2018~2019 第一学期

2018 年 11 月 20 日

目录

实验五　基于对话框界面的消息发送应用程序

1　实验目的及要求

- 了解和掌握基于对话框的网络编程方法和界面设计工具；
- 学习在 C# 环境下，基于 C#Windows 窗体的编程，编写一个基于对话框界面的消息发送应用程序；
- 掌握从对话框界面输入 IP 地址的方法，以及添加类变量的方法。

2　实验环境

- 操作环境：Windows 10；
- 编程环境：Visual Studio 2015；
- 程序原理：Socket 网络程序设计
- 程序使用 Visual C# 下的“Windows 窗体应用程序”。

3　实验内容及步骤

3.1　服务器端程序

该程序中通信协议使用的是面向连接的 TCP 协议 (SOCK_STREAM)。服务器端的 IP 地址使用系统指定的 IP 地址，端口号在程序中指定为 2000，用符号常量来定义。

- 调试环境：Visual Stdio 2015
- 服务器 IP 地址：由系统指定
- 服务器端口号：2000
- 程序核心文件：TCPserver5/Program.cs(附录 A-1)、TCPserver5/Form1.cs(附录 A-2)

- 程序功能：点击“准备发送”按钮后服务器端开始接收用户请求，当有客户提出连接请求时，在端口 2000 与客户端进行 TCP 连接，连接成功后，点击“发送”按钮时读取文本框中的字符串向客户端进行发送，或点击“退出”按钮关闭连接。

3.2 客户端程序

- 调试环境：Visual Stdio 2015
- 客户 IP 地址和端口：由系统指定
- 程序核心文件：TCPclient5/Program.cs(附录 A-3)、TCPclient5/Form1.cs(附录 A-4)
- 程序功能：点击“连接”按钮后客户端程序读取 IP 文本框中的服务器 IP 地址并向服务器提出 TCP 连接的请求，当连接建立后，点击“接收”按钮从服务器的端口 2000 接收数据并显示在文本框中，或点击“退出”按钮关闭连接。

3.3 本机回环测试

- 测试环境：Visual Studio 2015
- 测试步骤：
  1. 在同一台主机上同时启动服务器和客户端程序；
  2. 在客户端程序中输入 IP 地址“127.0.0.1”进行连接；
  3. 在客户端中点击“接收”按钮；
  4. 在服务器端中编辑要发送的信息并点击“发送”按钮；
  5. 观察记录实验结果；
  6. 分别点击客户端和服务器端的“退出”按钮并关闭程序。

3.4 远程互通测试

- 测试环境：Visual Studio 2015
- 测试步骤：
  1. 将两台主机连入同一个网络；
  2. 分别在两台主机上的命令行窗口输入命令“ipconfig”查看并记录各自的 IP 地址；
  3. 在一台主机上启动服务器程序，另一台主机上启动客户端程序；

4. 在客户端程序中输入服务器端主机的 IP 地址进行连接；
5. 在客户端中点击“接收”按钮；
6. 在服务器端中编辑要发送的信息并点击“发送”按钮；
7. 观察记录实验结果；
8. 分别点击客户端和服务器端的“退出”按钮并关闭程序；
9. 交换运行两台主机的服务器和客户端程序并重复步骤 4 至步骤 8。

4 实验结果

4.1 本地回环测试结果

服务器端：图 1.1

图 1.1: 本地回环服务器端测试结果

客户端：图 1.2

图 1.2: 本地回环客户端测试结果

4.2 远程互通 IP 记录

主机 1：图 1.3

```
本地链接 IPv6 地址. . . . . : fe80::cd69:78c3:f849:6fed%4
IPv4 地址 . . . . . : 192.168.137.1
子网掩码 . . . . . : 255.255.255.0
默认网关. . . . . :
```

图 1.3: 主机 1IP

主机 2：图 1.4

```
本地链接 IPv6 地址. . . . . : fe80::f886:cd0d:63e8:5411%11
IPv4 地址 . . . . . : 192.168.137.236
子网掩码 . . . . . : 255.255.255.0
默认网关. . . . . : 192.168.137.1
```

图 1.4: 主机 2IP

4.3 远程互通测试 1 结果 (主机 A 做服务器端，主机 B 做客户端)

主机 1：图 1.5

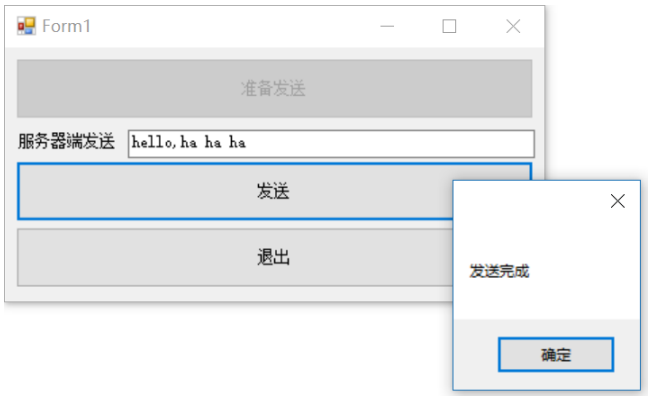

图 1.5: 远程互通测试 2 主机 2 结果

主机 2：图 1.6

图 1.6: 远程互通测试 1 主机 1 结果

4.4 远程互通测试 2 结果 (主机 A 做客户端，主机 B 做服务器端)

主机 1：图 1.7

图 1.7: 远程互通测试 2 主机 1 结果

主机 2：图 1.8

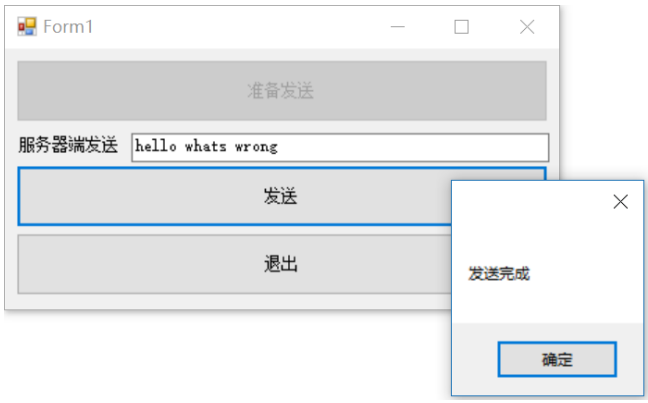

图 1.8: 远程互通测试 2 主机 2 结果

5 问题及心得

- 问题：在服务器端程序中，如果把服务器套接字的 Socket.Close() 调用放在发送完成后，则下一次开启监听时会报错。(图 1.9)
  - 原因：若在程序全部结束前调用了服务器套接字的 Socket.Close()，服务器套接字将完全失效，此后必须对服务器套接字重新初始化才能进行其他操作。
  - 解决：将服务器套接字的 Socket.Close() 调用放在窗口关闭后的主程序末尾。

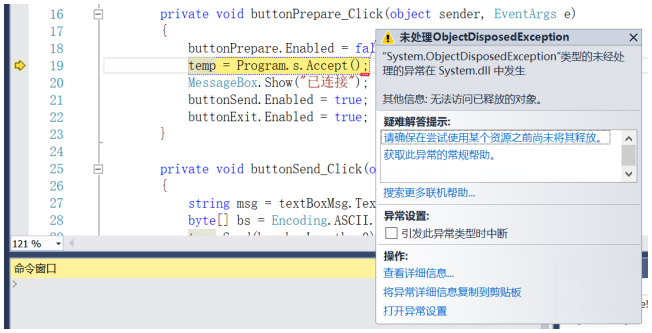

图 1.9: 错误信息

- 心得：
  1. 实践是检验真理的唯一标准；
  2. 实验是巩固知识的最快捷径；
  3. 掌握了 C# 中 TCP 协议的使用方法；
  4. 明白了 C#Socket 程序开发的一般模式；
  5. 精进了代码水平。

附录 A 核心代码清单

1 TCPserver5/Program.cs

```
using System;
using System.Windows.Forms;
using System.Net;
using System.Net.Sockets;

namespace TCPserver5
{
        static class Program
        {
                const int port = 2000;
                public static Socket s;
                /// <summary>
                ///  应用程序的主入口点。
                /// </summary>
                [STAThread]
                static void Main()
                {
                        s = new Socket(AddressFamily.InterNetwork,
                                       SocketType.Stream,
                                       ProtocolType.Tcp);
                        IPEndPoint ipe = new IPEndPoint(IPAddress.Any, port);
                        //用指定的端口和ip初始化IPEndPoint类的新实例
                        s.Bind(ipe);//绑定EndPoint对像（2000端口和ip地址）
                        s.Listen(0);//开始监听
                        Application.EnableVisualStyles();
                        Application.SetCompatibleTextRenderingDefault(false);
                        Application.Run(new Form1());
                        s.Close();
                }
        }
}
```

2 TCPserver5/Form1.cs

```
using System;
using System.Text;
using System.Windows.Forms;
using System.Net.Sockets;

namespace TCPserver5
{
```

```
8         public partial class Form1 : Form
9         {
10                Socket temp;
11                public Form1()
12                {
13                        InitializeComponent();
14                }
15
16                private void buttonPrepare_Click(object sender, EventArgs e)
17                {
18                        buttonPrepare.Enabled = false;
19                        temp = Program.s.Accept();
20                        MessageBox.Show("已连接");
21                        buttonSend.Enabled = true;
22                        buttonExit.Enabled = true;
23                }
24
25                private void buttonSend_Click(object sender, EventArgs e)
26                {
27                        string msg = textBoxMsg.Text;
28                        byte[] bs = Encoding.ASCII.GetBytes(msg);
29                        temp.Send(bs, bs.Length, 0);
30                        MessageBox.Show("发送完成");
31                }
32
33
34                private void buttonExit_Click(object sender, EventArgs e)
35                {
36                        temp.Close();
37                        buttonSend.Enabled = false;
38                        buttonExit.Enabled = false;
39                        buttonPrepare.Enabled = true;
40                }
41        }
42 }
```

3 TCPclient5/Program.cs

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5  using System.Windows.Forms;
6
7  namespace TCPclient5
8  {
```

```
9         static class Program
10        {
11                /// <summary>
12                /// 应用程序的主入口点。
13                /// </summary>
14                [STAThread]
15                static void Main()
16                {
17                        Application.EnableVisualStyles();
18                        Application.SetCompatibleTextRenderingDefault(false);
19                        Application.Run(new Form1());
20                }
21        }
22 }
```

4 TCPclient5/Form1.cs

```
1  using System;
2  using System.Text;
3  using System.Windows.Forms;
4  using System.Net;
5  using System.Net.Sockets;
6
7  namespace TCPclient5
8  {
9         public partial class Form1 : Form
10        {
11                Socket c;
12                const int port = 2000;
13                public Form1()
14                {
15                        InitializeComponent();
16                }
17
18                private void buttonConnect_Click(object sender, EventArgs e)
19                {
20                        string ip_str = textBoxIP.Text;
21                        IPEndPoint ipe = new IPEndPoint(IPAddress.Parse(ip_str), port);
22                        c = new Socket(AddressFamily.InterNetwork,
23                                        SocketType.Stream,
24                                        ProtocolType.Tcp);//创建Socket
25                        c.Connect(ipe);
26                        MessageBox.Show("连接成功");
27                        buttonConnect.Enabled = false;
28                        buttonRecv.Enabled = true;
29                        buttonExit.Enabled = true;
```

```
30                  }
31
32                  private void buttonRecv_Click(object sender, EventArgs e)
33                  {
34                          buttonRecv.Enabled = false;
35                          byte[] recvBytes = new byte[1024];
36                          int bytes = c.Receive(recvBytes, recvBytes.Length, 0);
37                          //从服务器端接受返回信息
38                          textBoxRecv.Text = Encoding.ASCII.GetString(recvBytes, 0, bytes);
39                          buttonRecv.Enabled = true;
40                  }
41
42                  private void buttonExit_Click(object sender, EventArgs e)
43                  {
44                          c.Close();
45                          buttonConnect.Enabled = true;
46                          buttonRecv.Enabled = false;
47                          buttonExit.Enabled = false;
48                  }
49          }
50  }
```