TestPingCSharpV3

Software Documentation

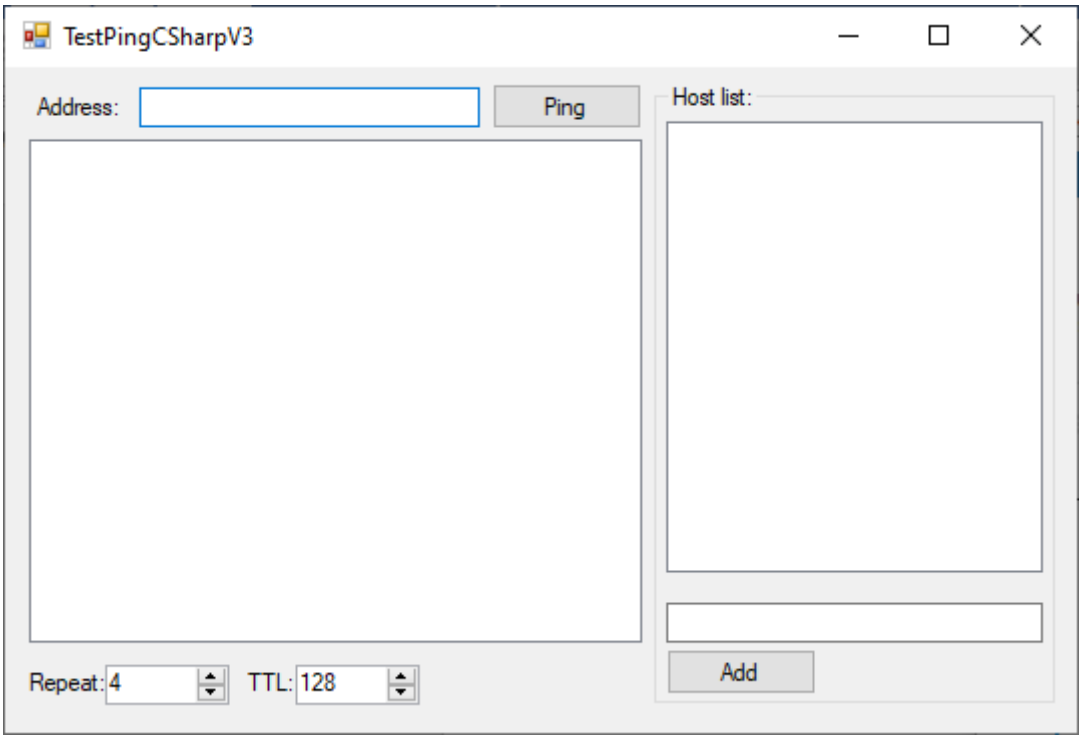
Author: matiwa

Table of contents

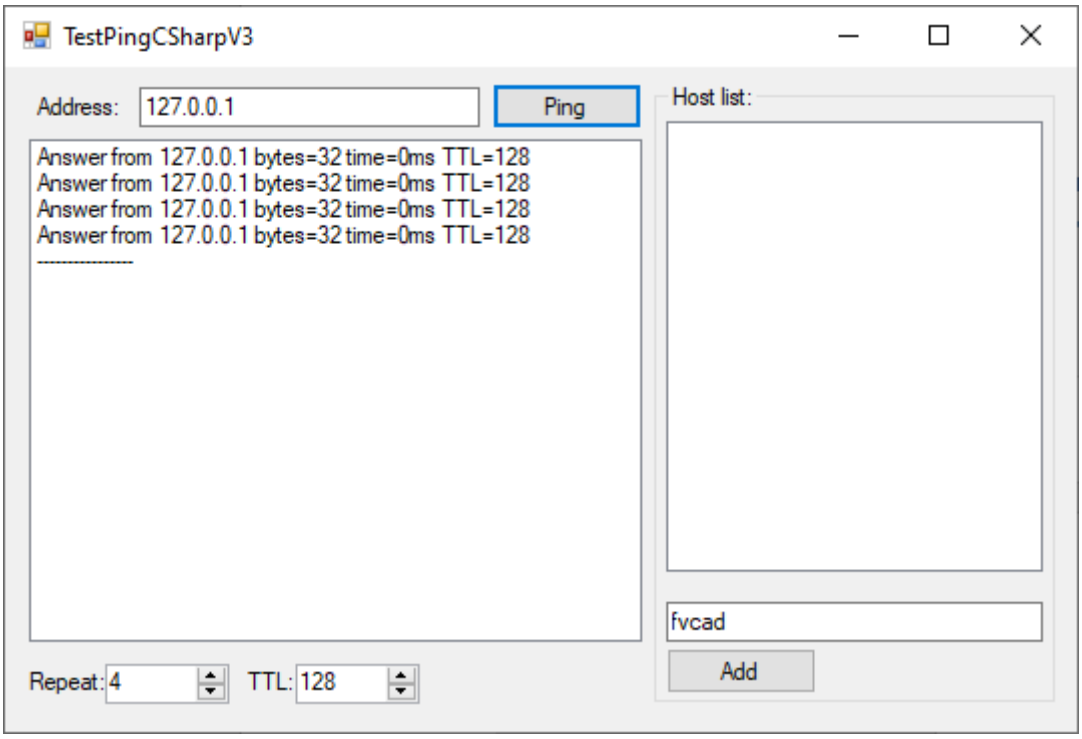
Introduction

This software documentation includes: description of the application's operation, what is needed for use, algorithms used, interface description and source code description. This application is used to diagnose network connections by Ping command.

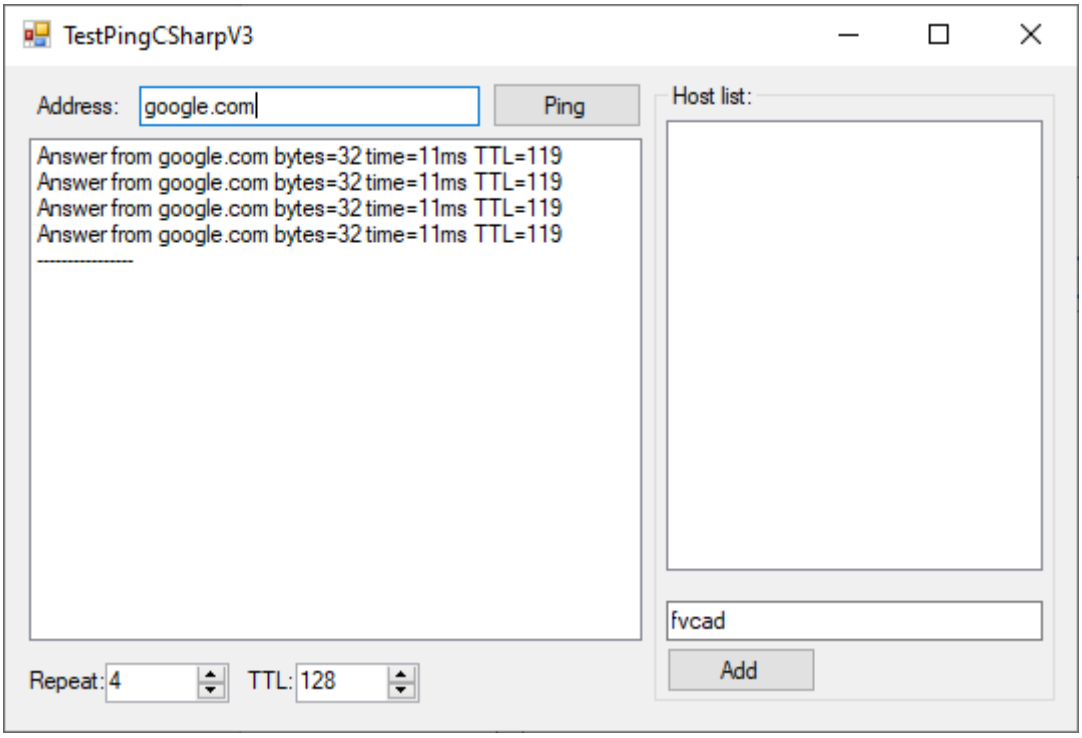
Describing of the application’s operation



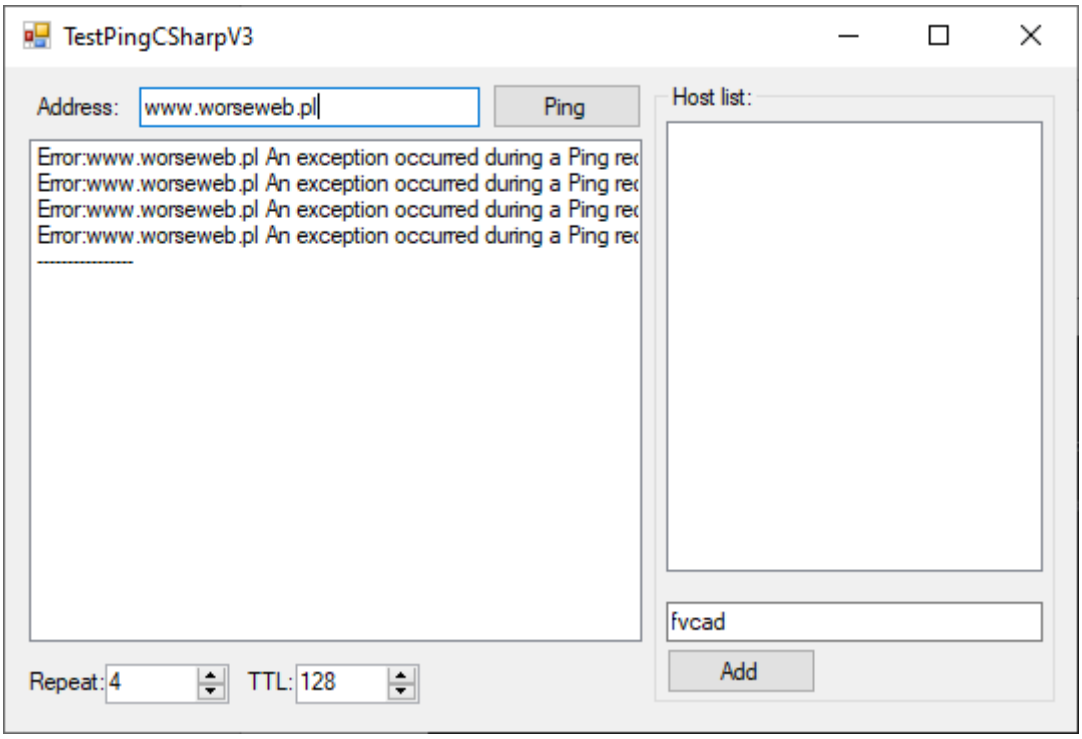
Drawing 1: The beginning of the application's operation [own study]



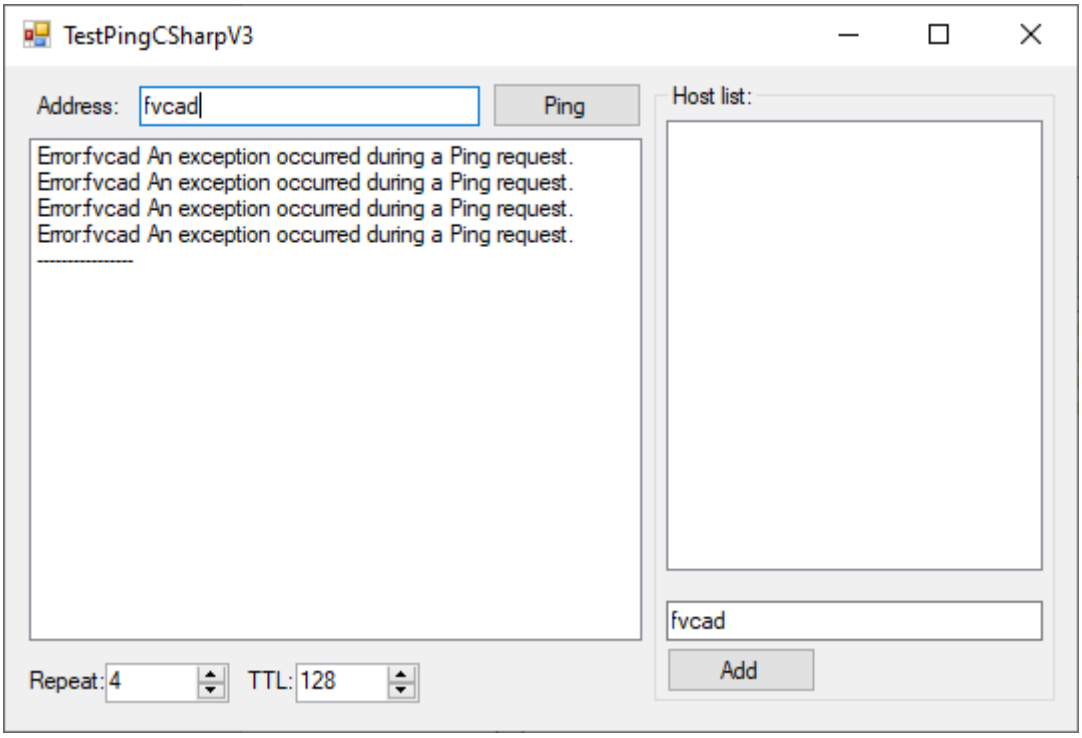
Drawing 2: The effect after entering the IPv4 address of the localhost[own study]



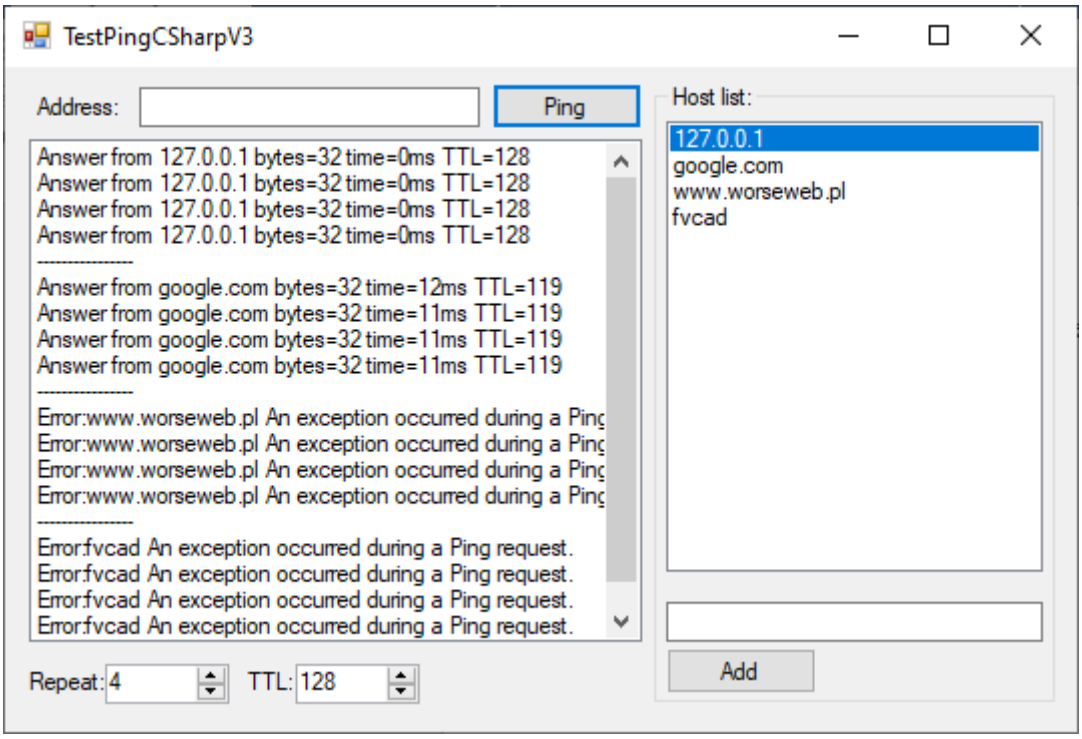
Drawing 3: The effect after entering the website [own study]



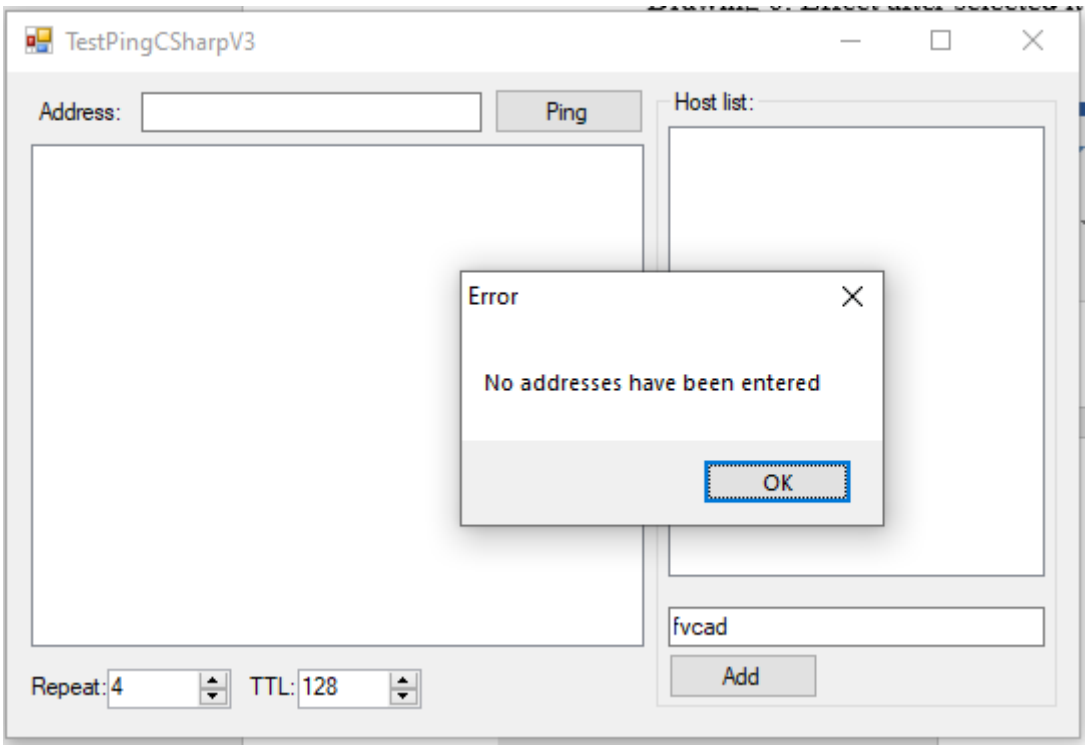
Drawing 4: Effect after entering a wrong address [own study]



Drawing 5: Effect after entering random characters [own study]



Drawing 6: Effect after selected items in the list [own study]



Drawing 7: Effect after typing nothing [own study]

After starting the program, the user enters the IP address or website address. It can also set the number of repetitions of the Ping command (max. 100 - default 4) and the TTL parameter

(max. 128 - default value). Accepts the entered data by pressing the Enter key or the Ping key. If there is a network connection and the recipient information is correct, the measurement data appears. Otherwise, in case of wrong address, random characters (not IP and WWW address) or no connection, a message will appear with the error details. It is possible that the user is looking at nothing and typing nothing, then the message "No addresses have been entered" will appear. There are certainly bugs that the developer did not discover while working on the application.

Ultimately, the concept assumed that after selecting a given item from the list of addresses, the application was to test it. However, what happened is that I am testing all the items on the list.

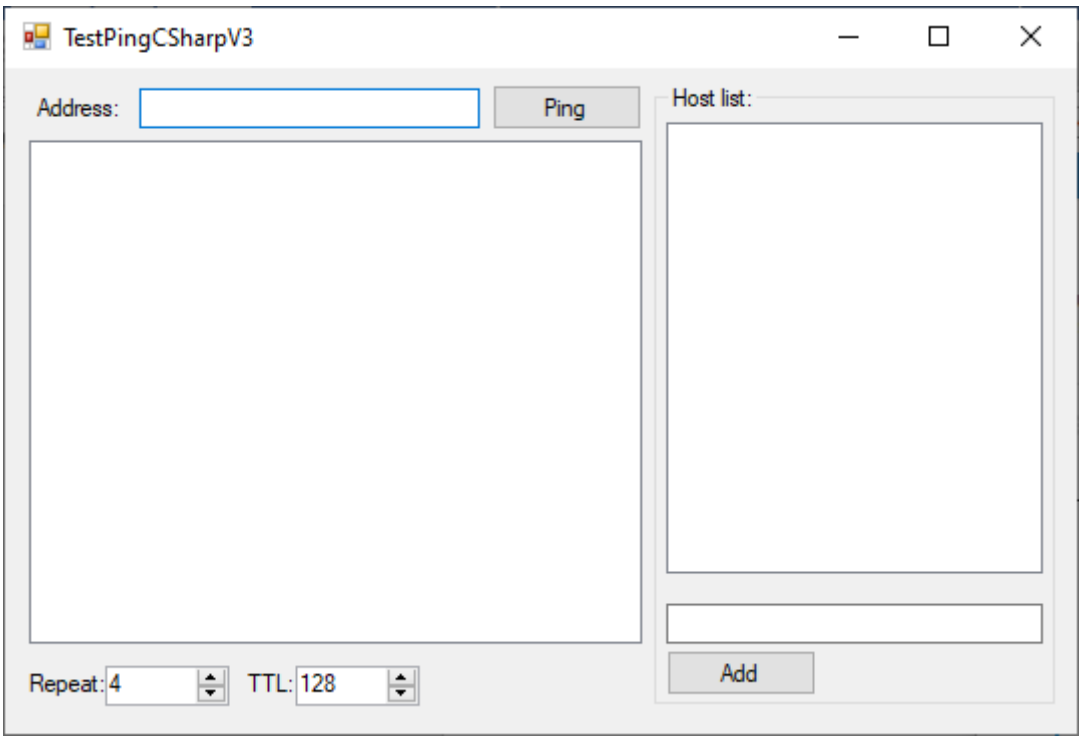
What is needed for use?

The application does not require installation. It only needs the Windows operating system.

Algorithm used

The basic form of the algorithm can be deduced from the previous section. It only needs the Windows operating system. In summary, the application is equivalent to the Ping command in the Windows Command Prompt.

Interface description



Drawing 7: Graphical interface [own study]

The interface is typical for a Windows Forms Application. There are essential components: groupbox, textboxes, labels, buttons, listbox and numbericupdown.

Source code description

The project was made in the C# programming language, in the Visual Studio Community 2017 programming environment. All work was done on the Windows 10 operating system. The application's source code looks like this.

```
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.NetworkInformation;

namespace TestPingCSharpV3
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void BPing_Click(object sender, EventArgs e)
        {
            listBox1.Items.Clear();
            if (TextBox1.Text != "" || ListBox2.Items.Count > 0)
            {
                PingOptions opcje = new PingOptions
                {
                    Ttl = (int)numericUpDown2.Value,
                    DontFragment = true
                };
                string dane = "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa";
                byte[] bufor = Encoding.ASCII.GetBytes(dane);
                int timeout = 120;
                if (TextBox1.Text != "")
                {
                    for (int i = 0; i < (int)numericUpDown1.Value; i++)
                        listBox1.Items.Add(this.WyslijPing(TextBox1.Text, timeout,
bufor, opcje));
                    listBox1.Items.Add("----------------");
                }
                if (ListBox2.Items.Count > 0)
                {
                    foreach (string host in ListBox2.Items)
                    {
                        for (int i = 0; i < (int)numericUpDown1.Value; i++)
                            listBox1.Items.Add(this.WyslijPing(host, timeout, bufor,
opcje));
                        listBox1.Items.Add("----------------");
                    }
                }
            }
```

```
            else
            {
                MessageBox.Show("No addresses have been entered", "Error");
            }
        }

        private void BAdd_Click(object sender, EventArgs e)
        {
            if (TextBox2.Text != String.Empty)
                if (TextBox2.Text.Trim().Length > 0)
                {
                    ListBox2.Items.Add(TextBox2.Text);
                    TextBox1.Clear();
                }
        }

        private void TextBox1_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == 13) BPing_Click(sender, e);
        }

        private void TextBox2_KeyPress(object sender, KeyPressEventArgs e)
        {
            if (e.KeyChar == 13) BAdd_Click(sender, e);
        }

        private void ListBox2_DoubleClick(object sender, EventArgs e)
        {
            if (ListBox2.SelectedIndex != -1)
                ListBox2.Items.RemoveAt(ListBox2.SelectedIndex);
        }

        private string WyslijPing(string adres, int timeout, byte[] bufor, PingOptions
opcje)
        {
            Ping ping = new Ping();
            try
            {
                PingReply odpowiedz = ping.Send(adres, timeout, bufor, opcje);
                if (odpowiedz.Status == IPStatus.Success)
                    return "Answer from " + adres + " bytes=" +
                    odpowiedz.Buffer.Length + " time=" + odpowiedz.RoundtripTime + "ms
TTL=" +
                    odpowiedz.Options.Ttl;
                else
                    return "Error:" + adres + " " + odpowiedz.Status.ToString();
            }
            catch (Exception ex)
            {
                return "Error:" + adres + " " + ex.Message;
            }
        }
    }
}
```

Listing 1: Source code [own study]

List of drawings

List of listings