
# Projekt w języku C#: Komunikator Sieciowy

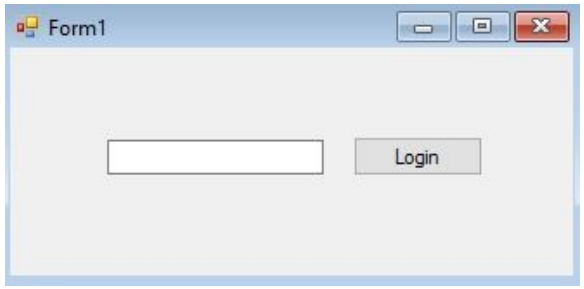
Natalia Wochtman, Leonard Dolecki

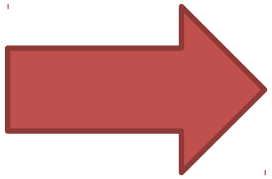
# Komunikator sieciowy

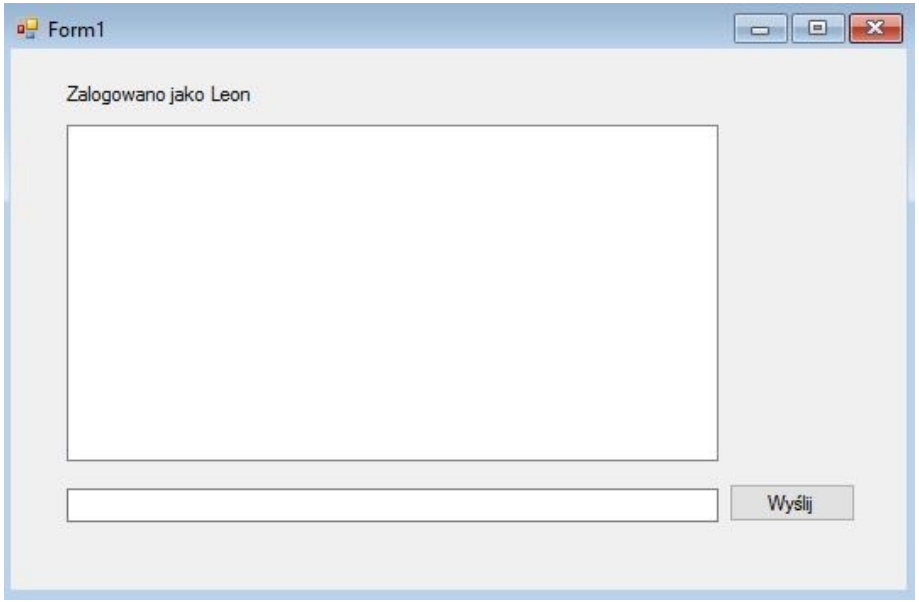
- Jest to program, który pozwala na natychmiastowe przesyłanie komunikatów miedzy dwoma lub wieloma komputerami. Przykłady to np.: Gadu Gadu czy Messenger.

# Możliwości programu

Nasz program pozwoli na wysyłanie wiadomości między klientami zalogowanymi na serwerze. Będą oni mieli oddzielne loginy, które nie będą mogły się powtarzać.





# Czego potrzebowaliśmy

Gniazda sieciowe – czyli punkty końcowe w komunikacji między urządzeniami sieciowymi. Dedykowana dla nich jest klasa Socket.

Można je podzielić na:

- Datagram sockets
- Stream sockets
- Raw sockets

My skorzystamy z gniazd TCP i UDP.

# Co udało nam się zrobić dotychczas

Stworzyliśmy projekt Console Application, który pozwala na połączenie między klientem, a serwerem. Do uzyskania połączenia sieciowego potrzebowaliśmy stworzyć dwa gniazda z czego jedno było serwerem, a drugie klientem.

Obecnie rozwiązany problem programistyczny:

– uzyskanie połączenia między klientem a serwerem.

# Server

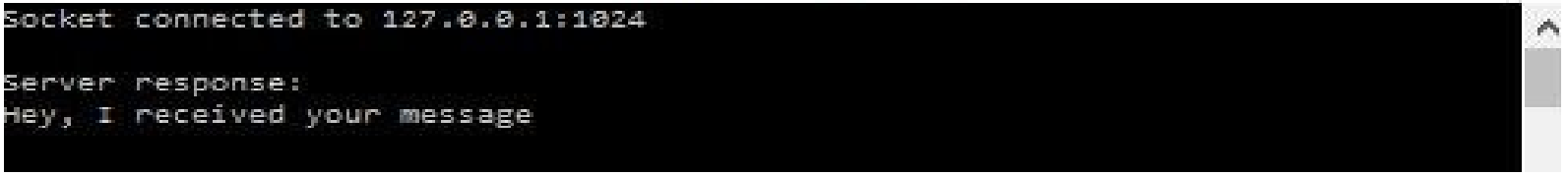
```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;

namespace Gniazda_Sieciowe
{
    class Serwer
    {
        static void Main(string[] args)
        {
            Socket serverSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);
            Socket internalSocket;
            byte[] recBuffer = new byte[256];

            try
            {
                serverSocket.Bind(new IPEndPoint(IPAddress.Parse("127.0.0.1"), 1024));
                serverSocket.Listen(100);

                while (true)
                {

                    Console.WriteLine("Waiting for a connection...");
                    internalSocket = serverSocket.Accept();
                    internalSocket.Receive(recBuffer);
                    Console.WriteLine("\nClient message:\n{0}", ASCIIEncoding.ASCII.GetString(recBuffer));
                    byte[] msg = Encoding.ASCII.GetBytes("Hey, I received your message");
                    internalSocket.Send(msg);


                }

            }
            catch (Exception e)
            {
                Console.WriteLine(e.ToString());
            }


            Console.Read();
        }
```

# Client

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

namespace Klient
{
    class Klient
    {

        static void Main(string[] args)
        {
            byte[] bytes = new byte[1024];

            Socket clientSocket = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);
            try
            {
                clientSocket.Connect("127.0.0.1", 1024);
                Console.WriteLine("Socket connected to {0}", clientSocket.RemoteEndPoint.ToString());
                clientSocket.Send(ASCIIEncoding.ASCII.GetBytes("Hey server, are you there?"));

                // Console.WriteLine("Socket connected to {0}", clientSocket.RemoteEndPoint.ToString());
                int bytesRec = clientSocket.Receive(bytes);
                Console.WriteLine("\nServer response:\n{0}",
                Encoding.ASCII.GetString(bytes, 0, bytesRec));
                Console.Read();
            }
            catch (Exception e)
            {
                Console.WriteLine("Unexpected exception : {0}", e.ToString());
            }

        }

    }
```

# Jak to działa:





# Co trzeba jeszcze zrobić:

- dołożyć więcej klientów,
- uzyskać połączenie między konkretnymi klientami mającymi dostęp do serwera,
- umożliwić przesyłanie wiadomości między tymi klientami,
- stworzyć lepszy interfejs graficzny aplikacji,
- opcjonalnie: stworzyć możliwość przechowywania wiadomości na serwerze.

Dziękujemy za uwagę.