Descubra como utilizar NetworkStream em aplicações Windows

No desenvolvimento de aplicações que requerem comunicação de rede em ambientes Windows, o NetworkStream é uma classe fundamental que permite o envio e recebimento de dados através de uma rede. Esta classe é parte do namespace System.Net.Sockets no .NET Framework e é amplamente utilizada para implementar comunicação baseada em sockets.

O NetworkStream fornece um fluxo de dados sequencial para leitura e escrita, que pode ser associado a um objeto Socket. Este artigo irá guiá-lo através de um exemplo prático de como criar um servidor e um cliente que se comunicam via NetworkStream em um ambiente Windows.

Exemplo Prático: Criando um Servidor e Cliente usando NetworkStream

Passo 1: Criando o Servidor

O servidor irá escutar em uma porta específica e aceitar conexões de clientes. Quando um cliente se conecta, o servidor usará NetworkStream para ler e escrever dados.

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

class Server
{
    public static void Main()
    {
        TcpListener server = null;
        try
        {
            // Define o IP do servidor e a porta de escuta
            Int32 port = 13000;
            IPAddress localAddr = IPAddress.Parse("127.0.0.1");

            // Cria um TcpListener
            server = new TcpListener(localAddr, port);

            // Inicia o servidor
            server.Start();

            // Buffer para leitura de dados
            Byte[] bytes = new Byte[256];
            String data = null;

            // Aguarda uma conexão
```

```
            Console.WriteLine("Aguardando uma conexão... ");

            // Aceita a conexão de um cliente
            TcpClient client = server.AcceptTcpClient();
            Console.WriteLine("Conectado!");

            // Obtém o NetworkStream do cliente
            NetworkStream stream = client.GetStream();

            int i;

            // Lê os dados enviados pelo cliente
            while ((i = stream.Read(bytes, 0, bytes.Length)) != 0)
            {
                // Converte os dados para string
                data = Encoding.ASCII.GetString(bytes, 0, i);
                Console.WriteLine("Recebido: {0}", data);

                // Processa os dados recebidos
                data = data.ToUpper();

                byte[] msg = Encoding.ASCII.GetBytes(data);

                // Envia os dados de volta ao cliente
                stream.Write(msg, 0, msg.Length);
                Console.WriteLine("Enviado: {0}", data);
            }

            // Fecha a conexão
            client.Close();
        }
        catch (SocketException e)
        {
            Console.WriteLine("SocketException: {0}", e);
        }
        finally
        {
            // Para o servidor
            server.Stop();
        }

        Console.WriteLine("\nPressione ENTER para continuar...");
        Console.Read();
    }
}
```

Passo 2: Criando o Cliente

O cliente se conectará ao servidor e enviará uma mensagem. Em seguida, ele receberá a resposta do servidor e a exibirá.

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

class Client
{
    public static void Main()
    {
        try
        {
            // Define o IP do servidor e a porta de conexão
            Int32 port = 13000;
            TcpClient client = new TcpClient("127.0.0.1", port);

            // Obtém o NetworkStream do cliente
            NetworkStream stream = client.GetStream();

            // Mensagem a ser enviada ao servidor
            string message = "Hello, Server!";
            byte[] data = Encoding.ASCII.GetBytes(message);

            // Envia a mensagem ao servidor
            stream.Write(data, 0, data.Length);
            Console.WriteLine("Enviado: {0}", message);

            // Buffer para armazenar a resposta do servidor
            data = new byte[256];
            String responseData = String.Empty;

            // Lê a resposta do servidor
            int bytes = stream.Read(data, 0, data.Length);
            responseData = Encoding.ASCII.GetString(data, 0, bytes);
            Console.WriteLine("Recebido: {0}", responseData);

            // Fecha a conexão
            stream.Close();
            client.Close();
        }
        catch (ArgumentNullException e)
        {
            Console.WriteLine("ArgumentNullException: {0}", e);
        }
        catch (SocketException e)
        {
            Console.WriteLine("SocketException: {0}", e);
        }
```

```
        Console.WriteLine("\nPressione ENTER para continuar...");
        Console.Read();
    }
}
```

Conclusão

O NetworkStream é uma ferramenta poderosa para implementar comunicação de rede em aplicações Windows. Ele permite a leitura e escrita de dados através de uma rede de forma eficiente e simplificada. Com os exemplos fornecidos, você pode começar a criar suas próprias aplicações de servidor e cliente utilizando NetworkStream.