Basics of working with pipes in C# .NET part 2: continuous byte communication

JUNE 17, 2015 2 COMMENTS (HTTPS://DOTNETCODR.COM/2015/06/17/BASICS-OF-WORKING-WITH-PIPES-IN-C-NET-PART-2-CONTINUOUS-BYTE-COMMUNICATION/#COMMENTS)

In this (https://dotnetcodr.com/2015/06/16/basics-of-working-with-pipes-in-c-net-part-1-send-and-receive-a-single-byte/) post we briefly introduced how interprocess communication pipes are represented in .NET. The server sent a single byte to the client and the client sent a single byte in response. We'll add some more flesh to the code but still keep it very simple. We'll let the client and server send each other individual bytes as messages. Obviously that is still very childish but it will do for demo purposes. We'll continue with proper messages in the next post.

Here's the extended server code with a lot more output. We read a line of input from the console but only transmit the very first byte. We wait for the client's response. The byte value of 'x', i.e. 120 will indicate that the client wants to end the communication:

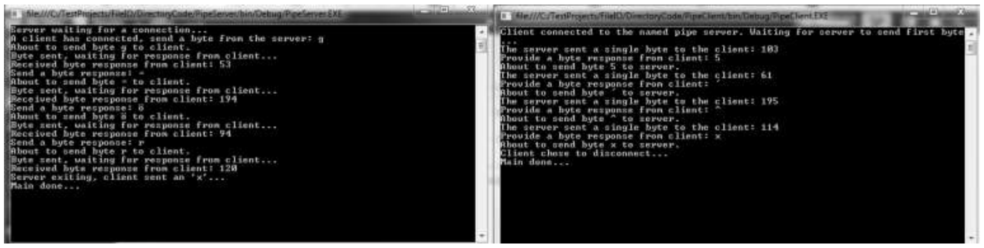
```
private static void SendByteAndReceiveResponseContinuous()
{
    using (NamedPipeServerStream namedPipeServer = new NamedPipeServerStream("test-pipe"))
    {
        Console.WriteLine("Server waiting for a connection...");
        namedPipeServer.WaitForConnection();
        Console.Write("A client has connected, send a byte from the server: ");
        string b = Console.ReadLine();
        Console.WriteLine("About to send byte {0} to client.", b);
        namedPipeServer.WriteByte(Encoding.UTF8.GetBytes(b).First());
        Console.WriteLine("Byte sent, waiting for response from client...");
        int byteFromClient = namedPipeServer.ReadByte();
        Console.WriteLine("Received byte response from client: {0}", byteFromClient);
        while (byteFromClient != 120)
        {
            Console.Write("Send a byte response: ");
            b = Console.ReadLine();
            Console.WriteLine("About to send byte {0} to client.", b);
            namedPipeServer.WriteByte(Encoding.UTF8.GetBytes(b).First());
            Console.WriteLine("Byte sent, waiting for response from client...");
            byteFromClient = namedPipeServer.ReadByte();
            Console.WriteLine("Received byte response from client: {0}", byteFromClient);
        }
        Console.WriteLine("Server exiting, client sent an 'x'...");
    }
}
```

The client code is very similar. It will connect to the server and keep exchanging bytes until that byte is 'x':

```
private static void ReceiveByteAndRespondContinuous()
{
    using (NamedPipeClientStream namedPipeClient = new NamedPipeClientStream("test-pipe"))
    {
        namedPipeClient.Connect();
        Console.WriteLine("Client connected to the named pipe server. Waiting for server to send firs
        Console.WriteLine("The server sent a single byte to the client: {0}", namedPipeClient.ReadByt
        Console.Write("Provide a byte response from client: ");
        string b = Console.ReadLine();
        Console.WriteLine("About to send byte {0} to server.", b);
        namedPipeClient.WriteByte(Encoding.UTF8.GetBytes(b).First());
        while (b != "x")
        {
            Console.WriteLine("The server sent a single byte to the client: {0}", namedPipeClient.Rea
            Console.Write("Provide a byte response from client: ");
            b = Console.ReadLine();
            Console.WriteLine("About to send byte {0} to server.", b);
            namedPipeClient.WriteByte(Encoding.UTF8.GetBytes(b).First());
        }

        Console.WriteLine("Client chose to disconnect...");
    }
}
```

You can call both methods from Main of the server and client console applications respectively and run them both. The communication flow might resemble the following:



```
Server waiting for a connection...
A client has connected, send a byte from the server: g
About to send byte g to client.
Byte sent, waiting for response from client...
Received byte response from client: 53
Send a byte response: =
About to send byte = to client.
Byte sent, waiting for response from client...
Received byte response from client: 194
Send a byte response: ö
About to send byte ö to client.
Byte sent, waiting for response from client...
Received byte response from client: 94
Send a byte response: r
About to send byte r to client.
Byte sent, waiting for response from client...
Received byte response from client: 120
Server exiting, client sent an 'x'...
Main done...
```

```
Client connected to the named pipe server. Waiting for server to send first byte
...
The server sent a single byte to the client: 103
Provide a byte response from client: 5
About to send byte 5 to server.
The server sent a single byte to the client: 61
Provide a byte response from client: ´
About to send byte ´ to server.
The server sent a single byte to the client: 195
Provide a byte response from client: ^
About to send byte ^ to server.
The server sent a single byte to the client: 114
Provide a byte response from client: x
About to send byte x to server.
Client chose to disconnect...
Main done...
```

(https://dotnetcodr.files.wordpress.com/2015/03/named-pipe-server-and-client-byte-communication-flow.png)

Read the next part here (https://dotnetcodr.com/2015/06/19/basics-of-working-with-pipes-in-c-net-part-3-message-transmission/).

View the list of posts on Messaging here (https://dotnetcodr.com/messaging/).
FILED UNDER .NET, MESSAGING TAGGED WITH C#, MESSAGING, PIPE

About Andras Nemes
I'm a .NET/Java developer living and working in Stockholm, Sweden.

2 Responses to *Basics of working with pipes in C# .NET part 2: continuous byte communication*

Josip Štajdohar says:
March 12, 2018 at 1:36 pm
Could you pleas give me advice on how to send whole arrays of byte data at once using a named pipe. Do i need to create an object like in the fifth part or is there a simpler way? I am used to pipes in c++ and there you can send any data type as long as the write buffer and the read buffer are declared as the specific type.

Reply

bonzadogBonzadog says:
August 25, 2019 at 2:02 pm
Any chance of a reply to Josip Štajdohar question?

Reply