Denne guide er oprindeligt
udgivet på Eksperten.dk

COMPUTERWORLD

# .NET og Java interoperabilitet I

## Denne artikel beskriver hvordan .NET og Java applikationer kan kommunikere via sockets.

## Artikel II beskriver web services.

## Den forudsætter lidt erfaring med både C#/VB.NET og Java.

Skrevet den **03. feb 2009** af **arne_v** I kategorien **Programmering / .NET** | ★★★☆☆

Historie:
V1.0 - 14/11/2004 - original
V1.1 - 05/08/2005 - tilføj note om little/big endian

### Indledning

Web services er meget hot for tiden. Men sockets som har været brugt i mere end 30 år er stadig et godt alternativ.

Man skal selv kode lidt mere, da det er message orienteret ikke RPC orienteret.

Men til gengæld er performance også meget bedre.

Brug af binære data fremfor tekst data performer endnu bedre, men så skal man til gengæld til at tage højde for nogle meget maskin nære ting såsom little versus big endian.

Man skal også gøre sig klart at det er svært at få lukket op for diverse mystiske porte i en firewall.

Sockets er et godt valg hvis:
* performance er vigtig
* det er simple data strukturer
* det er store data mængder
* det er kun LAN traffik
* man kender systemerne i begge ender

Nedenstående eksempel matcher faktisk ikke disse kriterier, men man lærer mere af noget svært kode end af noget nemt kode.

Der er rigtigt meget kode nedenfor, men de fleste læsere vil kun have brug for at studere to source kode filer. Men de to vil være forskellig for forskellige læsere, så jeg har valgt at have dem alle med.

### .NET client, Java server, tekst data

De vigtige .NET client klasser er:

* TcpClient
* StreamReader
* StreamWriter

De vigtige Java server klasser er:
* ServerSocket
* Socket
* Thread
* BufferedReader
* PrintWriter

C# client:

```
using System;
using System.IO;
using System.Net.Sockets;

class ClientText
{
    public static void Main(string[] args)
    {
        // connect til server
        TcpClient cli = new TcpClient("localhost", 12345);
        StreamWriter sw = new StreamWriter(cli.GetStream());
        StreamReader sr = new StreamReader(cli.GetStream());
        string line;
        string[] parts;
        // skriv til server
        int a = 12;
        int b = 34;
        sw.WriteLine("ADD " + a + " " + b);
        sw.Flush();
        // læs fra server og skriv resultat til skærm
        line = sr.ReadLine();
        parts = line.Split(" ".ToCharArray());
        int c = int.Parse(parts[1]);
        Console.WriteLine(c);
        // skriv til server
        string s = "abc";
        sw.WriteLine("DUP " + s);
        sw.Flush();
        // læs fra server og skriv resultat til skærm
        line = sr.ReadLine();
        parts = line.Split(" ".ToCharArray());
        string s2 = parts[1];
        Console.WriteLine(s2);
        // disconnect
        cli.Close();
    }
}
```

VB.NET client:

```
Imports System
Imports System.IO
Imports System.Net.Sockets

Class ClientText
    Public Shared Sub Main(ByVal args As String())
        ' connect til server
        Dim cli As TcpClient = New TcpClient("localhost", 12345)
        Dim sw As StreamWriter = New StreamWriter(cli.GetStream)
        Dim sr As StreamReader = New StreamReader(cli.GetStream)
        Dim line As String
        Dim parts As String()
        ' skriv til server
        Dim a As Integer = 12
        Dim b As Integer = 34
        sw.WriteLine("ADD " & a & " " & b)
        sw.Flush
        ' læs fra server og skriv resultat til skærm
        line = sr.ReadLine
        parts = line.Split(" ".ToCharArray)
        Dim c As Integer = Integer.Parse(parts(1))
        Console.WriteLine(c)
        ' skriv til server
        Dim s As String = "abc"
        sw.WriteLine("DUP " & s)
        sw.Flush
        ' læs fra server og skriv resultat til skærm
        line = sr.ReadLine
        parts = line.Split(" ".ToCharArray)
        Dim s2 As String = parts(1)
        Console.WriteLine(s2)
        ' disconnect
        cli.Close
    End Sub
End Class
```

Java server:

```
import java.io.*;
import java.net.*;

public class ServerText {
    public static void main(String[] args) {
        try {
            // lyt på port 12345
            ServerSocket ss = new ServerSocket(12345);
            while(true) {
                // accepter connection fra client
                Socket s = ss.accept();
                // start tråd til at håndtere client
                TextClientHandler tch = new TextClientHandler(s);
```

```
                tch.start();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
}

class TextClientHandler extends Thread {
    private Socket s;
    private BufferedReader br;
    private PrintWriter pw;
    public TextClientHandler(Socket s) {
        try {
            // initialiser læse og skrive klasser
            this.s = s;
            br = new BufferedReader(new
InputStreamReader(s.getInputStream()));
            pw = new PrintWriter(s.getOutputStream());
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    public void run() {
        try {
            String line;
            // læs input fra client
            while((line = br.readLine()) != null) {
                // split input fra client op i dele
                String[] parts = line.split(" ");
                if(parts[0].toUpperCase().equals("ADD")) {
                    // læg to heltal sammen
                    int a = Integer.parseInt(parts[1]);
                    int b = Integer.parseInt(parts[2]);
                    int c = a + b;
                    pw.println("RES " + c);
                    pw.flush();
                } else if(parts[0].toUpperCase().equals("DUP")) {
                    // dupliker en streng
                    String s = parts[1];
                    String s2 = s + s;
                    pw.println("RES " + s2);
                    pw.flush();
                } else {
                    System.err.println("Unknown command: " + line);
                }
            }
            // luk læse og skrive klasser
            br.close();
            pw.close();
            s.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
```

```
    }
}
```

**Java client, .NET server, tekst data**

De vigtige Java client klasser er:
* Socket
* BufferedReader
* PrintWriter

De vigtige .NET server klasser er:
* TcpListener
* TcpClient
* Thread
* StreamReader
* StreamWriter

Java client:

```
import java.io.*;
import java.net.*;

public class ClientText {
    public static void main(String[] args) {
        try {
            // connect til server
            Socket s = new Socket("localhost", 12345);
            PrintWriter pw = new PrintWriter(s.getOutputStream());
            BufferedReader br = new BufferedReader(new
InputStreamReader(s.getInputStream()));
            String line;
            String[] parts;
            // skriv til server
            int a = 12;
            int b = 34;
            pw.println("ADD " + a + " " + b);
            pw.flush();
            // læs fra server og skriv resultat til skærm
            line = br.readLine();
            parts = line.split(" ");
            int c = Integer.parseInt(parts[1]);
            System.out.println(c);
            // skriv til server
            String s1 = "abc";
            pw.println("DUP " + s1);
            pw.flush();
            // læs fra server og skriv resultat til skærm
            line = br.readLine();
            parts = line.split(" ");
            String s2 = parts[1];
            System.out.println(s2);
            // disconnect
```

```
            br.close();
            pw.close();
            s.close();
        } catch (NumberFormatException e) {
            e.printStackTrace();
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

C# server:

```
using System;
using System.IO;
using System.Net;
using System.Net.Sockets;
using System.Threading;

class ServerText
{
    public static void Main(string[] args)
    {
        // lyt på port 12345
        TcpListener srv = new TcpListener(IPAddress.Any, 12345);
        srv.Start();
        while(true) {
            // accepter connection fra client
            TcpClient cli = srv.AcceptTcpClient();
            // start tråd til at håndtere client
            TextClientHandler tch = new TextClientHandler(cli);
            (new Thread(new ThreadStart(tch.Run))).Start();
        }

    }
}

class TextClientHandler
{
    private TcpClient cli;
    private StreamReader sr;
    private StreamWriter sw;
    public TextClientHandler(TcpClient cli)
    {
        // initialiser læse og skrive klasser
        this.cli = cli;
        sr = new StreamReader(cli.GetStream());
        sw = new StreamWriter(cli.GetStream());
    }
    public void Run()
```

```
    {
        string line;
        // læs input fra client
        while((line = sr.ReadLine()) != null) {
            // split input fra client op i dele
            string[] parts = line.Split(" ".ToCharArray());
            if(parts[0].ToUpper() == "ADD") {
                // læg to heltal sammen
                int a = int.Parse(parts[1]);
                int b = int.Parse(parts[2]);
                int c = a + b;
                sw.WriteLine("RES " + c);
                sw.Flush();
            } else if(parts[0].ToUpper() == "DUP") {
                // dupliker en streng
                string s = parts[1];
                string s2 = s + s;
                sw.WriteLine("RES " + s2);
                sw.Flush();
            } else {
                Console.WriteLine("Unknown command: " + line);
            }
        }
        // luk læse og skrive klasser
        sr.Close();
        sw.Close();
        cli.Close();
    }
}
```

VB.NET server:

```
Imports System
Imports System.IO
Imports System.Net
Imports System.Net.Sockets
Imports System.Threading

Class ServerText
    Public Shared Sub Main(ByVal args As String())
        ' lyt på port 12345
        Dim srv As TcpListener = New TcpListener(IPAddress.Any, 12345)
        srv.Start
        While True
            ' accepter connection fra client
            Dim cli As TcpClient = srv.AcceptTcpClient
            ' start tråd til at håndtere client
            Dim tch As TextClientHandler = New TextClientHandler(cli)
            Dim t As Thread = New Thread(AddressOf tch.Run)
            t.Start
        End While
    End Sub
```

```
End Class

Class TextClientHandler
    Private cli As TcpClient
    Private sr As StreamReader
    Private sw As StreamWriter

    Public Sub New(ByVal cli As TcpClient)
        ' initialiser læse og skrive klasser
        Me.cli = cli
        sr = New StreamReader(cli.GetStream)
        sw = New StreamWriter(cli.GetStream)
    End Sub

    Public Sub Run()
        Dim line As String
        ' læs input fra client
        line = sr.ReadLine
        While Not (line Is Nothing)
            ' split input fra client op i dele
            Dim parts As String() = line.Split(" ".ToCharArray)
            If parts(0).ToUpper = "ADD" Then
                ' læg to heltal sammen
                Dim a As Integer = Integer.Parse(parts(1))
                Dim b As Integer = Integer.Parse(parts(2))
                Dim c As Integer = a + b
                sw.WriteLine("RES " & c)
                sw.Flush
            Else
                If parts(0).ToUpper = "DUP" Then
                    ' dupliker en streng
                    Dim s As String = parts(1)
                    Dim s2 As String = s & s
                    sw.WriteLine("RES " & s2)
                    sw.Flush
                Else
                    Console.WriteLine("Unknown command: " & line)
                End If
            End If
            ' læs input fra client
            line = sr.ReadLine
        End While
        ' luk læse og skrive klasser
        sr.Close
        sw.Close
        cli.Close
    End Sub
End Class
```

**.NET client, Java server, binære data**

De vigtige .NET client klasser er:

* TcpClient
* BinaryReader
* BinaryWriter

De vigtige Java server klasser er:
* ServerSocket
* Socket
* Thread
* DataInputStream
* DataOutputStream

Java klasserne bruger big endian og dermed net order. .NET klasserne bruger little endian. Jeg har valgt at konvertere på .NET siden fordi net order er "standarden". For dem som ikke kender big versus little endian kan jeg oplyse at det er de 2 forskellige måder at sende integers mede mere end en byte (short/int/long). Ved big endian sendes en int med værdi 1 som 0x00 0x00 0x00 0x01 mens den ved little endian sendes som 0x01 0x00 0x00 0x00. Det er selvfølgelig nødvenigt at bruge samme endianess i både client og server.

C# client:

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

class EndianUtil
{
   public static short Swap(short v) {
        return (short)(((v >> 8) & 0x00FF) | ((v << 8) & 0xFF00));
   }
   public static int Swap(int v) {
        uint v2 = (uint)v;
        return (int)(((v2 >> 24) & 0x000000FF) |
                     ((v2 >> 8) & 0x0000FF00) |
                     ((v2 << 8) & 0x00FF0000) |
                     ((v2 << 24) & 0xFF000000));
   }
}

class ClientBinary
{
    private const byte ADD = 1;
    private const byte DUP = 2;
    private const byte RES = 3;
    public static void Main(string[] args)
    {
        // connect til server
        TcpClient cli = new TcpClient("localhost", 12345);
        BinaryWriter bw = new BinaryWriter(cli.GetStream());
        BinaryReader br = new BinaryReader(cli.GetStream());
        byte res;
        // skriv til server
```

```
        int a = 12;
        int b = 34;
        bw.Write(ADD);
        bw.Write(EndianUtil.Swap(a));
        bw.Write(EndianUtil.Swap(b));
        bw.Flush();
        // læs fra server og skriv resultat til skærm
        res = br.ReadByte();
        int c = EndianUtil.Swap(br.ReadInt32());
        Console.WriteLine(c);
        // skriv til server
        string s = "abc";
        byte[] data = Encoding.Default.GetBytes(s);
        bw.Write(DUP);
        bw.Write(EndianUtil.Swap((short)data.Length));
        bw.Write(data);
        bw.Flush();
        // læs fra server og skriv resultat til skærm
        res = br.ReadByte();
        int len = EndianUtil.Swap(br.ReadInt16());
        byte[] data2 = new byte[len];
        int ix = 0;
        while(ix < len)
        {
            ix += br.Read(data2, ix, len - ix);
        }
        string s2 = Encoding.Default.GetString(data2);
        Console.WriteLine(s2);
        // disconnect
        cli.Close();
    }
}
```

VB.NET client:

```
Imports System
Imports System.IO
Imports System.Text
Imports System.Net.Sockets

Class EndianUtil
    Public Shared Function Swap(ByVal v As Short) As Short
        Return (((v >> 8) And &H00FF) Or ((v << 8) And &HFF00))
    End Function

    Public Shared Function Swap(ByVal v As Integer) As Integer
        Return (((v >> 24) And &H000000FF) Or _
                ((v >> 8) And &H0000FF00) Or _
                ((v << 8) And &H00FF0000) Or _
                ((v << 24) And &HFF000000))
    End Function
End Class
```

```
Class ClientBinary
    Private Const ADD As Byte = 1
    Private Const DUP As Byte = 2
    Private Const RES As Byte = 3

    Public Shared Sub Main(ByVal args As String())
        ' connect til server
        Dim cli As TcpClient = New TcpClient("localhost", 12345)
        Dim bw As BinaryWriter = New BinaryWriter(cli.GetStream)
        Dim br As BinaryReader = New BinaryReader(cli.GetStream)
        Dim res As Byte
        ' skriv til server
        Dim a As Integer = 12
        Dim b As Integer = 34
        bw.Write(ADD)
        bw.Write(EndianUtil.Swap(a))
        bw.Write(EndianUtil.Swap(b))
        bw.Flush
        'læs fra server og skriv resultat til skærm
        res = br.ReadByte
        Dim c As Integer = EndianUtil.Swap(br.ReadInt32)
        Console.WriteLine(c)
        ' skriv til server
        Dim s As String = "abc"
        Dim data As Byte() = Encoding.Default.GetBytes(s)
        bw.Write(DUP)
        bw.Write(EndianUtil.Swap(CType(data.Length, Short)))
        bw.Write(data)
        bw.Flush
        'læs fra server og skriv resultat til skærm
        res = br.ReadByte
        Dim len As Integer = EndianUtil.Swap(br.ReadInt16)
        Dim data2(len) As Byte
        Dim ix As Integer = 0
        While ix < len
            ix += br.Read(data2, ix, len - ix)
        End While
        Dim s2 As String = Encoding.Default.GetString(data2)
        Console.WriteLine(s2)
        ' disconnect
        cli.Close
    End Sub
End Class
```

Java server:

```
import java.io.*;
import java.net.*;

public class ServerBinary {
    public static void main(String[] args) {
        try {
```

```
            // lyt på port 12345
            ServerSocket ss = new ServerSocket(12345);
            while(true) {
                // accepter connection fra client
                Socket s = ss.accept();
                // start tråd til at håndtere client
                BinaryClientHandler bch = new BinaryClientHandler(s);
                bch.start();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
}

class BinaryClientHandler extends Thread {
    private final static byte ADD = 1;
    private final static byte DUP = 2;
    private final static byte RES = 3;
    private Socket s;
    private DataInputStream dis;
    private DataOutputStream dos;
    public BinaryClientHandler(Socket s) {
        try {
            // initialiser læse og skrive klasser
            this.s = s;
            dis = new DataInputStream(s.getInputStream());
            dos = new DataOutputStream(s.getOutputStream());
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    public void run() {
        try {
            boolean more = true;
            while(more) {
                // læs input fra client
                byte cmd = dis.readByte();
                switch(cmd) {
                    case ADD:
                        // læg to heltal sammen
                        int a = dis.readInt();
                        int b = dis.readInt();
                        int c = a + b;
                        dos.writeByte(RES);
                        dos.writeInt(c);
                        dos.flush();
                        break;
                    case DUP:
                        // dupliker en streng
                        int len = dis.readShort();
                        byte[] data = new byte[len];
                        int ix = 0;
                        while(ix < len) {
```

```
                            ix += dis.read(data, ix , len - ix);
                        }
                        String s = new String(data, "ISO-8859-1");
                        String s2 = s + s;
                        byte[] data2 = s2.getBytes("ISO-8859-1");
                        dos.writeByte(RES);
                        dos.writeShort(data2.length);
                        dos.write(data2);
                        dos.flush();
                        break;
                    default:
                        System.err.println("Unknown command: " + cmd);
                        more = false;
                        break;
                }
            }
            // luk læse og skrive klasser
            dis.close();
            dos.close();
            s.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

**Java client, .NET server, binære data**

De vigtige Java client klasser er:
* Socket
* DataInputStream
* DataOutputStream

De vigtige .NET server klasser er:
* TcpListener
* TcpClient
* Thread
* BinaryReader
* BinaryWriter

Java klasserne bruger big endian og dermed net order. .NET klasserne bruger little endian. Jeg har valgt at konvertere på .NET siden fordi net order er "standarden". For dem som ikke kender big versus little endian kan jeg oplyse at det er de 2 forskellige måder at sende integers mede mere end en byte (short/int/long). Ved big endian sendes en int med værdi 1 som 0x00 0x00 0x00 0x01 mens den ved little endian sendes som 0x01 0x00 0x00 0x00. Det er selvfølgelig nødvenigt at bruge samme endianess i både client og server.

Java client:

```
import java.io.*;
```

```
import java.net.*;

public class ClientBinary {
    private final static byte ADD = 1;
    private final static byte DUP = 2;
    private final static byte RES = 3;
    public static void main(String[] args) {
        try {
            // connect til server
            Socket s = new Socket("localhost", 12345);
            DataOutputStream dos = new DataOutputStream(s.getOutputStream());
            DataInputStream dis = new DataInputStream(s.getInputStream());
            byte res;
            // skriv til server
            int a = 12;
            int b = 34;
            dos.writeByte(ADD);
            dos.writeInt(a);
            dos.writeInt(b);
            dos.flush();
            // læs fra server og skriv resultat til skærm
            res = dis.readByte();
            int c = dis.readInt();
            System.out.println(c);
            // skriv til server
            String s1 = "abc";
            dos.writeByte(DUP);
            byte[] data = s1.getBytes("ISO-8859-1");
            dos.writeShort(data.length);
            dos.write(data);
            dos.flush();
            // læs fra server og skriv resultat til skærm
            res = dis.readByte();
            int len = dis.readShort();
            byte[] data2 = new byte[len];
            int ix = 0;
            while(ix < len) {
                ix += dis.read(data2, ix , len - ix);
            }
            String s2 = new String(data2, "ISO-8859-1");
            System.out.println(s2);
            // disconnect
            dis.close();
            dos.close();
            s.close();
        } catch (NumberFormatException e) {
            e.printStackTrace();
        } catch (UnknownHostException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

C# server:

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

class EndianUtil
{
   public static short Swap(short v) {
        return (short)(((v >> 8) & 0x00FF) | ((v << 8) & 0xFF00));
   }
   public static int Swap(int v) {
        uint v2 = (uint)v;
        return (int)(((v2 >> 24) & 0x000000FF) |
                     ((v2 >> 8) & 0x0000FF00) |
                     ((v2 << 8) & 0x00FF0000) |
                     ((v2 << 24) & 0xFF000000));
   }
}

class ServerText
{
    public static void Main(string[] args)
    {
        // lyt på port 12345
        TcpListener srv = new TcpListener(IPAddress.Any, 12345);
        srv.Start();
        while(true) {
            // accepter connection fra client
            TcpClient cli = srv.AcceptTcpClient();
            // start tråd til at håndtere client
            BinaryClientHandler bch = new BinaryClientHandler(cli);
            (new Thread(new ThreadStart(bch.Run))).Start();
        }

    }
}

class BinaryClientHandler
{
    private const byte ADD = 1;
    private const byte DUP = 2;
    private const byte RES = 3;
    private TcpClient cli;
    private BinaryReader br;
    private BinaryWriter bw;
    public BinaryClientHandler(TcpClient cli)
    {
        // initialiser læse og skrive klasser
        this.cli = cli;
        br = new BinaryReader(cli.GetStream());
```

```
        bw = new BinaryWriter(cli.GetStream());
    }
    public void Run()
    {
        bool more = true;
        while(more)
        {
            // læs input fra client
            byte cmd = br.ReadByte();
            switch(cmd) {
                case ADD:
                    // læg to heltal sammen
                    int a = EndianUtil.Swap(br.ReadInt32());
                    int b = EndianUtil.Swap(br.ReadInt32());
                    int c = a + b;
                    bw.Write(RES);
                    bw.Write(EndianUtil.Swap(c));
                    bw.Flush();
                    break;
                case DUP:
                    // dupliker en streng
                    int len = EndianUtil.Swap(br.ReadInt16());
                    byte[] data = new byte[len];
                    int ix = 0;
                    while(ix < len)
                    {
                        ix += br.Read(data, ix, len - ix);
                    }
                    String s = Encoding.Default.GetString(data);
                    String s2 = s + s;
                    byte[] data2 = Encoding.Default.GetBytes(s2);
                    bw.Write(RES);
                    bw.Write(EndianUtil.Swap((short)data2.Length));
                    bw.Write(data2);
                    bw.Flush();
                    break;
                default:
                    Console.WriteLine("Unknown command: " + cmd);
                    more = false;
                    break;
            }
        }
        // luk læse og skrive klasser
        br.Close();
        bw.Close();
        cli.Close();
    }
}
```

VB.NET server:

```
Imports System
```

```
Imports System.Text
Imports System.IO
Imports System.Net
Imports System.Net.Sockets
Imports System.Threading

Class EndianUtil
    Public Shared Function Swap(ByVal v As Short) As Short
        Return (((v >> 8) And &H00FF) Or ((v << 8) And &HFF00))
    End Function

    Public Shared Function Swap(ByVal v As Integer) As Integer
        Return (((v >> 24) And &H000000FF) Or _
                ((v >> 8) And &H0000FF00) Or _
                ((v << 8) And &H00FF0000) Or _
                ((v << 24) And &HFF000000))
    End Function
End Class

Class ServerText
    Public Shared Sub Main(ByVal args As String())
        ' lyt på port 12345
        Dim srv As TcpListener = New TcpListener(IPAddress.Any, 12345)
        srv.Start
        While True
            ' accepter connection fra client
            Dim cli As TcpClient = srv.AcceptTcpClient
            ' start tråd til at håndtere client
            Dim bch As BinaryClientHandler = New BinaryClientHandler(cli)
            Dim t As Thread = New Thread(AddressOf bch.Run)
            t.Start
        End While
    End Sub
End Class

Class BinaryClientHandler
    Private Const ADD As Byte = 1
    Private Const DUP As Byte = 2
    Private Const RES As Byte = 3
    Private cli As TcpClient
    Private br As BinaryReader
    Private bw As BinaryWriter

    Public Sub New(ByVal cli As TcpClient)
        ' initialiser læse og skrive klasser
        Me.cli = cli
        br = New BinaryReader(cli.GetStream)
        bw = New BinaryWriter(cli.GetStream)
    End Sub

    Public Sub Run()
        Dim more As Boolean = True
        While more
            ' læs input fra client
            Dim cmd As Byte = br.ReadByte
```

```
            Select cmd
            Case ADD
                ' læg to heltal sammen
                Dim a As Integer = EndianUtil.Swap(br.ReadInt32)
                Dim b As Integer = EndianUtil.Swap(br.ReadInt32)
                Dim c As Integer = a + b
                bw.Write(RES)
                bw.Write(EndianUtil.Swap(c))
                bw.Flush
            Case DUP
                ' dupliker en streng
                Dim len As Integer = EndianUtil.Swap(br.ReadInt16)
                Dim data(len-1) As Byte
                Dim ix As Integer = 0
                While ix < len
                    ix += br.Read(data, ix, len - ix)
                End While
                Dim s As String = Encoding.Default.GetString(data)
                Dim s2 As String = s + s
                Dim data2 As Byte() = Encoding.Default.GetBytes(s2)
                bw.Write(RES)
                bw.Write(EndianUtil.Swap(CType(data2.Length, Short)))
                bw.Write(data2)
                bw.Flush
            Case Else
                Console.WriteLine("Unknown command: " + cmd)
                more = False
            End Select
        End While
        ' luk læse og skrive klasser
        br.Close
        bw.Close
        cli.Close
    End Sub
End Class
```

**Kommentar af simonvalter d. 19. nov 2004 | 1**

Rigtig gode eksempler, som med arnes xml artikler så kan man finde lige det eksempel man har brug for. Man kunne måske godt bruge lidt mere forklaring omkring little/big endian

**Kommentar af mhl2k01 d. 06. nov 2006 | 2**

super :)

**Kommentar af md_craig d. 03. apr 2005 | 3**

**Kommentar af cronck d. 31. jan 2005 | 4**

Fint skal det jo være! :-)

**Kommentar af mathiash d. 06. apr 2006 | 5**

**Kommentar af spif2001 d. 05. aug 2005 | 6**

Meget brugbar, men enig med simonvalter mht. little/big endian

**Kommentar af fcjensendk d. 12. okt 2005 | 7**