COMPUTING SUBJECT:	**Socket programming**
TYPE:	Assignment
IDENTIFICATION:	SocketIterative
COPYRIGHT:	*Michael Claudius*
LEVEL:	Intermediate
TIME CONSUMPTION:	1-2 hours
EXTENT:	50 lines
OBJECTIVE:	TCP-sockets iterative style
PRECONDITIONS:	Computer Networks Ch. 2.7
COMMANDS:	

IDENTIFICATION: SocketIterative

The Mission
We are going to explore the TCP socket programming using an iterative server. First we change the server to handle several clients; secondly we shall look into good design principles.

Useful C# links

- http://www.codeproject.com/Articles/10649/An-Introduction-to-Socket-Programming-in-NET-using
- http://msdn.microsoft.com/en-us/library/bb397809(v=vs.90).aspx

Right now you have the following iterative socket server program:

```
TcpListener serverSocket = new TcpListener(6789);
serverSocket.Start();

TcpClient connectionSocket = serverSocket.AcceptTcpClient();
//Socket connectionSocket = serverSocket.AcceptSocket();
Console.WriteLine("Server activated");

Stream ns = connectionSocket.GetStream();
//Stream ns = new NetworkStream(connectionSocket);
StreamReader sr = new StreamReader(ns);
StreamWriter sw = new StreamWriter(ns);
sw.AutoFlush = true; // enable automatic flushing

string message = sr.ReadLine();
string answer = "";
while (message != null && message != "")
{
    Console.WriteLine("Client: " + message);
    answer = message.ToUpper();
    sw.WriteLine(answer);
    message = sr.ReadLine();
}
ns.Close();
connectionSocket.Close();
serverSocket.Stop();
```

Critical remarks
The present server program is unfavourable as it only handles one client-request one time and then it stops.

Assignment 1: Application class: TCPEchoServer

Create a new project SocketIterative with an application class TCPEchoServer with the usual Main method just like the TCPEchoServer, but it must be able to handle several clients infinitely; i.e. never stops. Therefore in

Main()
State an infinite while-loop after the server socket start.
.

Tip: Similar to the previous TCPEchoServer.
Compile and run!
Then start your TCPEchoClient1 and see that everything operates fine….and you don't need to restart the server program each time.

Assignment 2 Good programming and design principles

Although this program was only made for fast educational purpose, we shall now critise it!!
Use 5-10 minutes together with some other students to discuss the disadvantages of the program in relation to the OO-principles for good programming (coupling and coherence).
Don't look on the next page !!
Take the time to write down keywords and then proceed to the next page.

Critical remarks (I said don't look !)
The class does not have a good and unique (unambiguous) responsibility. The offered code is mixing socket purpose, I/O and management (main). For a model class method main should be considered as a test method and nothing else.
Also notice that there is no catch of exceptions, but either main should throw an exception, or we could set up a try-catch sentence.
Note, that although main() is in the class, other classes might use/utilize the TCPServer-class.
Why and how ?

Now we will restructure the program, so we have two classes:

TCPEchoServer1: A server class for management and setting up the server socket connection
EchoService: A service Model class handling the connection socket communication and data manipulation.

The class EchoService is defined as a public class, only containing the constructor EchoService and the method *doIt:*

Datafields:
TcpClient connectionSocket

Constructors
EchoService(TcpClient connection)

Methods
void doIt()

Assignment 2: Model class: EchoService
Create a new class EchoService with the mentioned data fields, the constructor and the method doIt:

EchoService(TcpClient connection)
initializes the data field with the parameter

doIt()
only handles all the connection socket communication.

Tip: Cut and paste a part from the TCPEchoServer1.
Compile!
You don't need a try–catch-sentence catching SocketException and IOException.

Assignment 4: Application class: TCPEchoServer1
Create a new application class TCPEchoServer1 with the usual main method with the following responsibility:

main()
Create server socket.
State an infinite while-loop and inside the loop:
Create connection socket by calling an accept-method
Create a EchoService object.
Then call the doIt method.

Tip: Very Similar to TCPEchoServer1. Compile and run!
Then start the TCPClient1 and see that everything operates fine….

Assignment 5: Several clients
Try to run many clients against the same server.
What do you observe ? What happens ? Why ?

Assignment 6: Exception handling Optional
Set up a try-catch construction catching exceptions like: SocketException and IOException.
Although not necessary. Just experiment.

Assignment 7: Several clients Optional.
In the TCPClient project create a new TCPClient2 class with a different for-loop so one client automatically can send many sentences (just with sender name and lines numbered) to the server, i.e. no user input from Console
Then start the client program on several machines.
What happens?
Tip: To understand what goes on, it might be necessary to slow down the speed using: Thread.sleep(100).

Critical remarks
The present program is unfavourable as it only handles client-requests sequentially; i.e. one by one.
We need a concurrent server.
Goto to the next assignment SocketConcurrent.