**Using a Synchronous Server Socket**

**.NET Framework 4**

Synchronous server sockets suspend the execution of the application until a connection request is received on the socket. Synchronous server sockets are not suitable for applications that make heavy use of the network in their operation, but they can be suitable for simple network applications.

After a [Socket](http://msdn.microsoft.com/en-us/library/system.net.sockets.socket.aspx) is set to listen on an endpoint using the [Bind](http://msdn.microsoft.com/en-us/library/system.net.sockets.socket.bind.aspx) and [Listen](http://msdn.microsoft.com/en-us/library/system.net.sockets.socket.listen.aspx) methods, it is ready to accept incoming connection requests using the [Accept](http://msdn.microsoft.com/en-us/library/system.net.sockets.socket.accept.aspx) method. The application is suspended until a connection request is received when the **Accept** method is called.

When a connection request is received, **Accept** returns a new **Socket** instance that is associated with the connecting client. The following example reads data from the client, displays it on the console, and echoes the data back to the client. The **Socket** does not specify any messaging protocol, so the string "<EOF>" marks the end of the message data. It assumes that a **Socket** named listener has been initialized and bound to an endpoint.

**C#**

Console.WriteLine("Waiting for a connection...");

Socket handler = listener.Accept();

String data = null;

while (true) {

bytes = new byte[1024];

int bytesRec = handler.Receive(bytes);

data += Encoding.ASCII.GetString(bytes,0,bytesRec);

if (data.IndexOf("<EOF>") > -1) {

break;

}

}

Console.WriteLine( "Text received : {0}", data);

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

handler.Send(msg);

handler.Shutdown(SocketShutdown.Both);

handler.Close();