

# Re: Making sure you receive it all

---

*Source:*

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.networks/2006-02/msg00370>

---

- *From:* "Alexander Nickolov" <[agnickolov@xxxxxxxx](mailto:agnickolov@xxxxxxxx)>
  - *Date:* Thu, 16 Feb 2006 17:50:29 -0800
- 

You can't get data out of order on a TCP connection.

Basically, on the TCP socket level in your app you simply read all incoming data as soon as it arrives and don't try to interpret it. On the next slightly higher level you split the data into packets.

Let's say you have a buffer of 4K (the size must be chosen to fit the largest packet you may receive, unless your higher level code can deal with multi-buffer messages, but let's not get into that...). This buffer may contain some data from previous socket reads, so it also needs a pointer to the free space (e.g. count of how many bytes are valid and haven't yet been consumed). When you read, you always pass the remaining number of bytes in your buffer and the address within the buffer where the free space starts (I leave the C# details up to you). Once data is read, you pass the entire buffer, along with the updated number of bytes valid to the next layer for processing. There it checks if there are at least 4 bytes in the buffer. If yes – that's your packet size and it decodes it from network byte order. Then it checks if there are at least that many bytes in the remainder of the buffer (after the first 4 bytes). If no – it returns without further processing (that's why the assumption the largest packet can fit into the buffer). If yes – it passes the packet to the next layer for processing. After that your code moves any remaining data to the beginning of the buffer and adjusts the count of bytes remainign accordingly. Finally – it goes back to step one and checks for 4 bytes available in the buffer. This is very important – you may have mutliple packets ready for processing and you don't want to hold them back!

--

=====

Alexander Nickolov  
Microsoft MVP [VC], MCS D  
email: [agnickolov@xxxxxxxx](mailto:agnickolov@xxxxxxxx)

Re: Making sure you receive it all

MVP VC FAQ: <http://www.mvps.org/vcfaq>

=====  
"Daniel" <DanielV@xxxxxxxxxxxxxxxx> wrote in message  
[news:OCpSsrzMGHA.1488@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:OCpSsrzMGHA.1488@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

oh! That function is below, but it just reads the first 4 bytes into an integer that represents number of bytes expected, clearly useless now you have told me this.

So yes I did believe that thank you for explaining. So how are you ever supposed to know how much is coming over the wire unless you know what to expect in advance? And again I don't see the difference between end receive and receive?

If I sit waiting for a 0 do I need to keep appending the bytes, and if they come over in a different order how are you supposed to know the order to put them in? Think I might start a new thread on this one.

```
/// <summary>  
  
/// Gets the size of the data being sent over the wire from the first 4  
bytes  
  
/// </summary>  
  
/// <param name="recvData">Data being sent</param>  
  
/// <returns>Size of total stream to come</returns>  
  
private int GetExpectedSize(byte [] recvData)  
{  
  
int size = BitConverter.ToInt32(recvData,0);  
  
return size;  
  
}
```

"Alexander Nickolov" <agnickolov@xxxxxxx> wrote in message  
[news:OJ1FiZyMGHA.1088@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:OJ1FiZyMGHA.1088@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

Your cardinal error is you assume you get the beginning of your data packet whenever you receive data on the socket. TCP is a streaming protocol and your message boundaries are not preserved. You need to simply store all data received in a buffer of your own, then analyze the stream and split it back into application packets.

Re: Making sure you receive it all

Re: Making sure you receive it all

You didn't show the source of your GetExpectedSize,  
but make sure you both encode at the sender and decode  
at the receiver the size in network byte order...

--  
=====  
Alexander Nickolov  
Microsoft MVP [VC], MCSD  
email: agnickolov@xxxxxxx  
MVP VC FAQ: <http://www.mvps.org/vcfaq>  
=====

"Daniel" <DanielV@xxxxxxxxxxxxxxxx> wrote in message  
<news:uv0mLfnMGHA.3392@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>

I am quite confused...here is some code with how i was  
going to implement  
it: This is called asynchronously.  
As it is called async my \_totalStreamSize will continue to be  
set using  
the first 4 bytes f the data so this obviously will not work as  
the next  
bit of data to come through wont have the total size at the  
front. So  
whats the best way?At which point should i store the total?

```
/// <summary>  
  
/// Callback running Asynchronously firing when data  
received from the  
server  
  
/// </summary>  
  
/// <param name="asyn"></param>  
  
void OnDataReceived( IAsyncResult asyn)  
  
{  
  
CSocketPacket theSockId =  
(CSocketPacket)asyn.AsyncState ;  
  
try  
  
{  
  
//end receive...
```

## Re: Making sure you receive it all

```
int size = 0 ;

size = theSockId.thisSocket.EndReceive (asyn); //end receive
on socket
passed in and return totalbytes read

_totalStreamSize = GetExpectedSize(theSockId.dataBuffer);
//returns
total bytes read from first 4 bytes in buffer

if ( size == 0 )

{

theSockId.thisSocket.Close();

}

else

{

lock(this)

{

HandleDataReceived(theSockId.dataBuffer, size);

}

WaitForData();

}

}

catch(SocketException exc)

{

MessageBox.Show(exc.Message,"Failed to receive – " +
exc.GetType());

theSockId.thisSocket.Close();

}

catch(Exception exc)

{
```

Re: Making sure you receive it all

```
MessageBox.Show(exc.Message,"Failed to receive - " +  
exc.GetType());
```

```
theSockId.thisSocket.Close();
```

```
}
```

```
}
```

"Alexander Nickolov" <agnickolov@xxxxxxx> wrote in  
message

[news:OP3YIDmMGHA.3264@xxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:OP3YIDmMGHA.3264@xxxxxxxxxxxxxxxxxxxxxxxxxxxx)

Not static of course – a member of your  
class will be much  
better. And you should not reset it to zero,  
but subtract the  
number of bytes used (you may have  
received more data  
which you'll ignore if you reset to zero).  
Also, shift the remaining  
data in your buffer to the beginning and  
store new data from  
that position. Additionally, keep in mind you  
may get several  
messages in one go, so don't stop processing  
after the first  
message, only stop when you encounter an  
incomplete message.  
I'm not familiar with the environment you  
are using (.NET and  
C# would be my first guess), so can't write  
you sample code.

--

```
=====  
Alexander Nickolov  
Microsoft MVP [VC], MCSD  
email: agnickolov@xxxxxxx  
MVP VC FAQ: http://www.mvps.org/vcfaq  
=====
```

"Dan" <dvazanias@xxxxxxx> wrote in  
message

[news:%23mklovkMGHA.3496@xxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:%23mklovkMGHA.3496@xxxxxxxxxxxxxxxxxxxxxxxxxxxx)

Hey

I want to ensure i receive

Re: Making sure you receive it all

my full data before i process  
how to  
respond but all the tutorials i  
seen use the Receive() part  
for the  
sockets.

I do whats pasted after this  
message, this is on the client  
end.

As you can see i use  
EndReceive(), and to be  
honest a little confused  
at the difference compared  
to receive(). I presume my  
size int will  
contain the amount of bytes  
read on that receive. So  
before calling my  
HandleDataReceived  
method i would need to do  
read the first 4 bytes  
which contain the length of  
the total data sent and check  
does my size  
received match that, if not i  
do not have all the data.

So should i make a static var  
of totalBytesReceived and  
when it  
matches the expected size  
by += it to size then do my  
handledatareceive() and set  
my static var back to 0? I  
presume also  
then that i have to keep  
appending these bytes  
received onto a byte  
array that i will then pass  
through to my handle  
function. If so how  
do you append bytes? I have  
ben using memory streams,  
doing a write  
and then when finished  
doing a getbytes, sound  
acceptable?

Let me know your opinions?

Re: Making sure you receive it all

I presume i must do the  
same server side?

```
void OnDataReceived(  
IAsyncResult asyn)  
  
{  
  
CSocketPacket theSockId =  
(CSocketPacket)asyn.AsyncState  
;  
  
try  
  
{  
  
//end receive...  
  
int size = 0 ;  
  
size =  
theSockId.thisSocket.EndReceive  
(asyn); //end receive on  
socket  
passed in and return  
totalbytes  
  
if ( size == 0 )  
  
{  
  
theSockId.thisSocket.Close();  
  
}  
  
else  
  
{  
  
lock(this)  
  
{  
  
HandleDataReceived(theSockId.dataBuffer,  
size);  
  
}  
  
WaitForData();
```

Re: Making sure you receive it all

```
}  
  
}  
  
catch(SocketException exc)  
  
{  
  
    MessageBox.Show(exc.Message,"Failed  
to receive - " +  
exc.GetType());  
  
    theSockId.thisSocket.Close();  
  
}  
  
catch(Exception exc)  
  
{  
  
    MessageBox.Show(exc.Message,"Failed  
to receive - " +  
exc.GetType());  
  
    theSockId.thisSocket.Close();  
  
}  
  
}  
  
--  
Dan
```