Writing by FADI Abdel-Qader/fadi822000@yahoo.com
My Space: http://spaces.msn.com/members/csharp2005

بسم الله الرحمن الرحيم

الدرس الحادى عشر: Socket & Network Layer Programming :

في هذا الجزء سوف نبين بشكل أكثر تفصيلا عن برمجة طبقة ال Network Layer وهي التي يتم التعامل معها لإرسال واستقبال البيانات بعد تحويلها من و إلى Stream عبر الشبكة، قمنا سابقا باستخدام ال TCP و UDP للإرسال وللاستقبال وبينا الفرق بينهما وفي هذا الجزء سوف نتحدث عن ال Socket Programming وال TCP & UDP Classes وفي النهاية سوف نتحدث عن Asynchronous Socket .

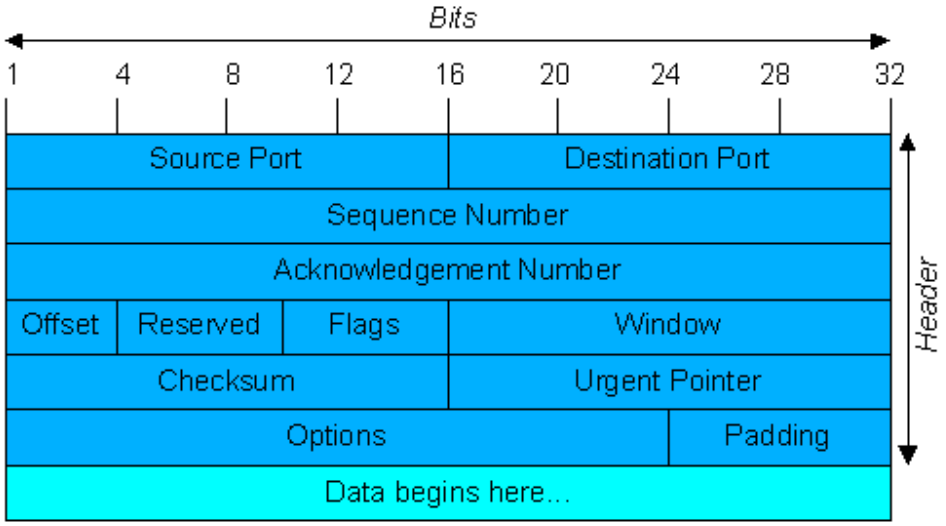
أولا: Socket Programming

من المعروف أن السوكت هي الأداة التي يتم نقل البيانات من خلالها من جهاز إلى أخر ولاستخدامها يلزم في البداية تعريف النيم سبيس System.Net.Sockets حيث يحتوي هذا النيم سبيس على عدد ضخم من ال Classes والتي يتم استخدامها في برمجيات الشبكة وسوف نتحدث عن أهما وهو Socket Class إذ يمكننا بمن التعامل مع ال TCP أو ال UDP أو مع أي نوع أخر من البرتوكولات بشكل مباشر ويتكون ال Socket Object Method من ثلاثة باروميترات كما يلي:

```
Socket MySocket = new Socket(AddressFamily. , SocketType., ProtocolType.);
```

حيث يتم في الباروميتر الأول تحديد نوعية ال IP Address والذي سوف تتعامل معه ويعطيك عدد كبير من الخيارات ومنها IPX والمستخدم في شبكات ال Novel أو ATM والمستخدم في شبكات ال ATM Networks أو NetBIOS Address وغيرها ... ومن أهم هذه الخيارات الInterNetwork وهو ما نستخدمه بشكل دائم مع البرمجيات الخاصة بالشبكات ويعرف على أن نوع IP هو من النوع IPv4 وهو المعتاد مع نظام مايكروسوفت وأغلب أنظمة التشغيل المعروفة حاليا وفي المستقبل القريب جدا سيتم الإستغناء عنه وليحل محله ال IPv6، في الباروميتر الثاني يتم تحديد نوع ال Socket أي هل سوف نستخدم Stream لإرسال البيانات أو شيء اخر وعادة ما يتم استخدام ال Stream لهذه المهمة حيث اننا سنعتمد نمطية التراسل من النوع Stream ، وأخيرا نحدد نوع البروتوكول المستخدم للإتصال فهل هو من النوع UDP أو TCP أو بروتوكولات أخرى مثلICMP Internet Control Message Protocol أو Internet Group Management Protocol IGMP أو اننا نريد مثلا إنشاء ال Socket لتعريف IP Security Header بإختيار IPSecAuthenticationHeader وغيرها وسوف نأتي على شرح مثل هذه الأمور لاحقا إنشاء الله، وهنا سوف نختار ال TCP أو UDP ومن المعروف أن برتوكول ال TCP هو برتوكول موجه وهذا يعني إجراء عملية التحقق من الوصول والتوصيل إلى شخص ما محدد أما برتوكول ال UDP فهو برتوكول سريع نسبيا و لاكنه لا يدعم عملية التحقق من الوصول السليم للبيانات المرسلة وهو مفيد جدا لإجراء عملية البث الإذاعي Broadcast وإنشاء مجموعات البث Multicast Group وهو ما شرحناه في الدرس الأول والثاني أنظر إلى الشكل التالي ويبين فيه ال Header الخاص بال TCP وال Header الخاص بال UDP ولاحظ الفرق بينهما:

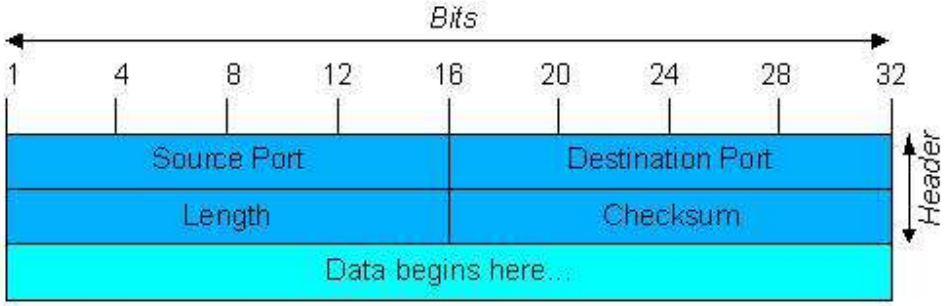
أولا ال TCP Header ويتكون من 32 Bits للبكت الواحد حيث يتم فيه تخزين عنوان المرسل في 16 Bits والمستقبل في 16 Bits والرقم التسلسلي للبكت في 32 Bits ورقم التحقق بالإضافة إلى ال Checksum وفي النهاية يتم وضع الجزء الخاص بالبيانات :



TCP Header

Data Offset: 4 bits the number of 32 bit words in the TCP Header. This indicates where the data begins. The TCP header (even one including options) is an integral number of 32 bits long.
Window: The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.
...

ثانيا ال UDP Header ويتكون من 32 Bits من البيانات للبكت الواحد ويحتوي على عنوان المرسل 16 Bits أما المستلم و ال Checksum فهما اختياريان وبشكل افتراضي لا يتم استخدامهم في عملية الإرسال:

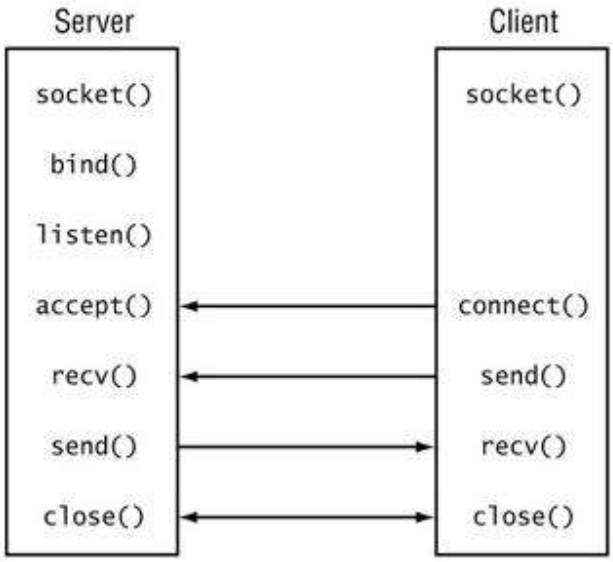


UDP Header

The Checksum in UDP Header. 16 bits.
Computed as the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the UDP header, and the data, padded as needed with zero bytes at the end to make a multiple of two bytes. If the checksum is cleared to zero, then checksuming is disabled. If the computed checksum is zero, then this field must be set to 0xFFFF.
...

استخدام ال Socket Programming لإنشاء TCP Connection :

تمر عملية الاتصال باستخدام ال TCP Socket Connection بمجموعة من المراحل وهي كما في الشكل التالي :



إذ تبدأ العملية في ال Client و ال server بإنشاء ال Socket كما يلي :

```
Socket MySocket = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
```

ثم ربط ال Socket مع الكمبيوتر الحالي باستخدام الميثود Bind وتستخدم فقط عند الاستقبال وكما يلي :

```
IPEndPoint ip = new IPEndPoint(IPAddress.Any, 5020);
MySocket.Bind(ip);
```

ثم القيام بعملية التصنت على البورت المحدد باستخدام الميثود listen ويمكنك تحدد عدد الأجهزة التي سيتم قبولها ولوضع عدد غير محدد نمرر له الرقم -1 ثم نقوم بالموافقة على الاتصال باستخدام الميثود accept وكما يلي :

```
MySocket.Listen(-1);
MySocket.Accept();
```

ويتم استقبال البيانات من خلال الميثود Receive حيث تعبئ البيانات في مصفوفة من النوع Byte وكما يلي :

```
byte[]Received=new byte[1024];
MySocket.Receive(Received);
```

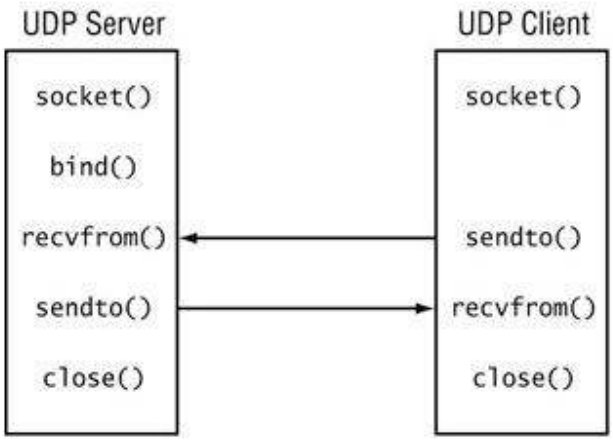
وهنا قمنا بإنشاء Connection من النوع TCP وبتعريفها على البورت(5020 كمثال) حيث يتم ربطها بال Socket باستخدام الميثود Bind وقمنا بتعريف Listen لا نهائي العدد -1 ..

ولتعريف برنامج الإرسال TCP Client باستخدام ال Socket لابد من تعريف السوكت مرة أخرى وإسناد عنوان السيرفر ورقم البورت بنقطة الهدف IPEndPoint ثم إرسال البيانات باستخدام الميثود Send وتتم عملية الإرسال بما تم تعريفه في ال socket حيث سنستخدم Stream Socket وكما يلي :

```
String str = Console.ReadLine();
ASCIIEncoding asen = new ASCIIEncoding();
byte[] msg = asen.GetBytes(str);

Socket MySocket = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
IPEndPoint remote = new IPEndPoint(IPAddress.Parse("192.168.1.101"), 5020);
MySocket.Connect(remote);
MySocket.Send(msg);
MySocket.Close();
```

استخدام ال Socket Programming لإنشاء UDP Connectionless :

تمر عملية الاتصال باستخدام ال UDP Socket Connection بمجموعة من المراحل وهي كما في الشكل التالي :



وتتشابه عملية الاتصال كما في ال TCP إذ تبدأ العملية في ال Client و ال server بإنشاء ال Socket كما يلي :

```
Socket MySocket = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Udp);
```

ثم ربط ال Socket مع الكمبيوتر الحالي باستخدام الميثود Bind وتستخدم فقط عند الاستقبال وكما يلي :

```
IPEndPoint sender = new IPEndPoint(IPAddress.Any, 5020);
MySocket.Bind(sender);
```

ولاستقبال البيانات نستخدم الميثود ReceiveFrom حيث نعرف في البداية End Point Reference بناء على ما تم تعريفه في السابقة ونمرره ك reference مع مصفوفة ال Byte إلى الميثود ReceiveFrom ومن ثم نستطيع تحويل المصفوفة إلى String من خلال الميثود GetString الموجودة ضمن الكلاس ASCII وكما يلي :

```
int recv;
byte[] data = new byte[1024];
EndPoint Remote = (EndPoint) (sender);
```

```
recv = newsock.ReceiveFrom(data, ref Remote);
Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
```

ويتم في الإرسال استخدام الميثود SendTo حيث نمرر لها البيانات بعد تحويلها من String إلى Byte Array وحجم البيانات المرسلة إذ يمكننا معرفته من خلال الميثود Length ونوع ال Flags حيث نريد عمل Broadcast لرسالة المرسلة واخيرا نمرر له ال EndPoint Object وكما يلي:

```
string welcome = "Hello All";
data = Encoding.ASCII.GetBytes(welcome);
newsock.SendTo(data, data.Length, SocketFlags.Broadcast, Remote);
```

يمكن وضع هذا الأكواد في Infinity While Loop بحيث لا تنتهي أو يمكن تحديدها بعدد معين من عمليات الإرسال والاستقبال ..

ثانيا: Socket Classes Members :

1- IPAddress Class : ويستخدم لتعريف IP Address حيث يمكن إسناده إلى ال IPEndPoint كمثال والصيغة العامة له كما يلي:

```
IPAddress newaddress = IPAddress.Parse("192.168.1.1");
```

ويمكن الإختيار بين اربعة خيارات في تحديد العنوان وهي كما يلي :

Any ويستخدم لتمثيل أي عنوان متاح على الشبكة

Broadcast ويستخدم لتمثيل البث الإذاعي لجميع الأجهزة على الشبكة

Loopback ويستخدم لتمثيل العنوان المعروف لل loopback وهو 127.0.0.1

None ويستخدم في حالة عدم وجود Network Interfase في النظام

كما يدعم مجموعة من الميثود وأهمها :

Equals يستخدم هذا الميثود بشكل عام للمقارنة بين tow Objects وهنا سيستخدم للمقارنة بين عنوانين ويرجع True إذا كانا متشابهين و False إذا كانا مختلفين.

GetHashCode وتستخدم لإرجاع العنوان إلى صيغة Hash Code

HostToNetworkOrder ويرجع الجزء الخاص بال Network من العنوان

NetworkToHostOrder ويرجع الجزء الخاص بال Host من العنوان

2- IPEndPoint Class : حيث استخدمناه لتحديد العنوان وال Port لل Host والذي نريد الاتصال به والصيغة العامة له كما يلي :

```
IPEndPoint end = new IPEndPoint(IPAddress.Parse("192.168.1.1"), 5020);
```

مجموعة الخواص التي تدعم في ال Socket Class وهي كما يلي:

AddressFamily ويرجع مجموعة العناوين المعرفة على ال Socket

Available ويرجع حجم البيانات الجاهزة للقراءة من ال Socket

Blocking ويعطي Get أو Set لمعرفة إذا كان ال socket يستخدم ال Blocking Mode أم لا

Connected وتستخدم هذه الخاصية بكثرة لمعرفة إذا كان ال Socket متصل مع ال Remote Host أم لا

Handle ويستخدم لمعرفة نظام التشغيل الذي يتعامل مع ال Socket

ProtocolType ويستخدم لمعرفة البروتوكول الذي يستخدم في ال Socket

RemoteEndPoint ويرجع معلومات عن ال Socket الذي يستخدم مع ال Remote Host

وكمثال لاستخداماتها:

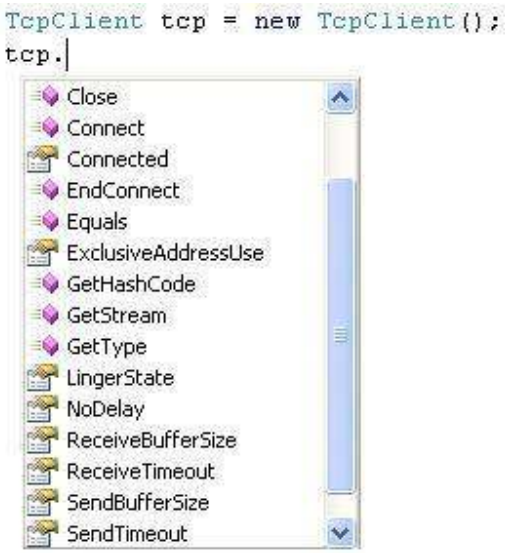
```
using System;
using System.Net;
using System.Net.Sockets;
class Socket_ Properties
{
    public static void Main()
    {
        IPAddress ia = IPAddress.Parse("127.0.0.1");
        IPEndPoint ie = new IPEndPoint(ia, 8000);
        Socket fmo = new Socket(AddressFamily.InterNetwork,
              SocketType.Stream, ProtocolType.Tcp);
        Console.WriteLine("AddressFamily: {0}",
              fmo.AddressFamily);
        Console.WriteLine("SocketType: {0}",
              fmo.SocketType);
        Console.WriteLine("ProtocolType: {0}",
              fmo.ProtocolType);
        Console.WriteLine("Blocking: {0}", fmo.Blocking);
        fmo.Blocking = false;
        Console.WriteLine("new Blocking: {0}", fmo.Blocking);
        Console.WriteLine("Connected: {0}", fmo.Connected);
        fmo.Bind(ie);
        IPEndPoint iep = (IPEndPoint)fmo.LocalEndPoint;
        Console.WriteLine("Local EndPoint: {0}",
               iep.ToString());
        fmo.Close();
    }

}
```

حيث سترجع المعلومات التالية:

```
AddressFamily: InterNetwork
SocketType: Stream
ProtocolType: Tcp
Blocking: True
new Blocking: False
Connected: False
Local EndPoint: 127.0.0.1:8000
Press any key to continue . . .
```

ثالثا: TCP & UDP Classes Members :

أولا ال Classes الخاصة بال TCP Connection Oriented Protocol :



1-TcpClient Class: حيث تحتوي على مجموعة من ال Methods وال Properties وهي كما يلي:

أولا: أهم الميثود الخاصة بها TCPClient Methods :

Connect: وتستخدم لأجراء عملية الاتصال مع ال server حيث نمرر فيها عنوان ال IP الخاص بال Server و رقم البورت وكما يلي:

```
TcpClient tcp = new TcpClient();
tcp.Connect(IPAddress.Parse("192.168.1.1"),5020);
```

Close: لإنهاء الاتصال مع ال TCP Socket.
EndConnect: لإنهاء Asynchronies Connection حيث ترجع Asynchronies Result.
GetStream: ويستخدم لقراءة ال Stream من ال Socket في عملية الإرسال و الاستقبال.

ثانيا: أهم الخصائص TCPClient Properties :

LingerState : وتأخذ get أو Set لتحديد أو معرفة ال Linger Time
NoDelay: وتأخذ get أو Set لتحديد أو معرفة إذا كان هناك وقت معين لتأخير أم لا
ExclusiveAddressUse: وتأخذ get أو Set لتحديد أو معرفة السوكت يسمح باستخدام ال Client Port أم لا.
SendBufferSize و ReceiveBufferSize: وتأخذ get أو Set لتحديد أو معرفة حجم ال Buffer المستخدم في ال stream والمعرف في TCP Client Object.
SendTimeout و ReceiveTimeout: وتأخذ get أو Set لتحديد أو معرفة الوقت المتاح لعملية الإرسال أو الإستقبال حيث يعطي Time Out في حالة أنه لم يجد الطرف الأخر خلال فترة زمنية معينة.

2-TcpListener Class: حيث تحتوي على مجموعة من ال Methods وال Properties وهي كما يلي:

```
TcpListener tcp_Listener = new TcpListener(IPAddress.Any,5020);
tcp_Listener.
```

أولا: أهم الميثود الخاصة بها TcpListener Methods :

AcceptSocket: وتستخدم لقبول عملية الاتصال مع ال Client.
Start : وهي Overloaded Method حيث انه في حالة تمرير رقم إليها يتم تحديد عدد الأجهزة التي تسمح بوجودها في الطابور أو ال Qouee وبدون تحديد رقم معين يصبح ال Qouee غير محدد.
Stop : وتستخدم لإخلاق عملية التصنت ويفضل وضعها في ال Finnaly عند استخدام ال Try و ال Catch حتى يتم إخلاق التصنت في حالة حدوث أي Exception.

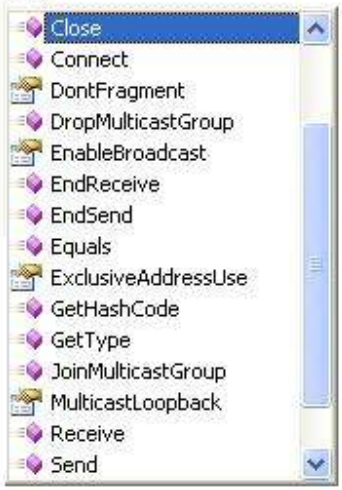
ثانيا: أهم الخصائص في TcpListener:

LocalEndpoint : حيث يرجع ال IP ورقم البورت المستخدم في ال LocalEndpoint المحدد.
Server: ومن خلالها نستطيع الوصول إلى كل الخصائص و الميثود في ال TCP Server والتي شرحناها سابقا مثل ال Accept وال Sendto وال Receive و Listen وغيرها

ثانيا ال Classes الخاصة بال UDP Connectionless Protocol :

UdpClient Class-1: وتستخدم لتعريف UDP Datagram Protocol Connection قمنا سابقا بتعريفها والتعامل معها وفي هذا الدرس سنبين أهم محتوياتها وهي كما يلي :

```
UdpClient udp = new UdpClient()
udp.
```



ومن أهم الميثود والخصائص الخاصة بها :

JoinMulticastGroup و DropMulticastGroup: لضم أو إلغاء عنوان أو مجموعة من العناوين من ال Multicast Group.
EnableBroadcast: وتأخذ Get أو Set لتفعيل ال Broadcasting في ال socket.
MulticastLoopback: وتأخذ Get أو Set لمعرفة أو تحديد ال Mlticast Loopback.

MulticastOption Class-2: ويستخدم في ال Multicasting حيث يتم تخزين IP Address List لتعامل معها في Multicast Group لعمل Join و Drop لأي Multicast Group وتستخدم كما يلي كمثال لإضافة عضوية لاستقبال رسائل Multicast :

أولا نعرف ال UDP Socket وكما يلي :

```
mcastSocket = new Socket(AddressFamily.InterNetwork,SocketType.Dgram,
ProtocolType.Udp);
```

ثانيا نقوم بتعريف Address List ثم نسند إليها ال IP الذي نريد إدخاله في ال Group أو نجعل ال User يدخل العنوان بنفسه نربطها بالسكوت باستخدام الميثود Bind وكما يلي :

```
IPAddress localIPAddr = IPAddress.Parse(Console.ReadLine());
mcastSocket.Bind(IPlocal);
```

ثالثا نقوم بتعريف ال Multicast Option ونسند لها العنوان المحدد كما يلي:

```
MulticastOption mcastOption;
mcastOption = new MulticastOption(localIPAddr);
```

ومن ثم نضيف التغير على ال حيث تأخذ هذه الميثود ثلاثة بارومیترات الأول لتحديد مستوى التغيير على IP أو على IPv6 أو على Socket أو TCP أو UDP وفي حالتنا هذه سوف نستخدم التغير على IP إذ ما نريده هو ضم IP إلى Multicast Group وفي البارومیتر الثاني نحدد نوع التغيير حيث نريد إضافة عضوية ويمكن الاختيار بين إضافة عضويه AddMembership أو إلغاء عضوية DropMembership وأخيرا نسند إليه ال MulticastOption Object والذي قمنا بإنشائه و كما يلي:

```
mcastSocket.SetSocketOption(SocketOptionLevel.IP,
SocketOptionName.AddMembership,mcastOption);
```

في الدرس القادم سوف نتحدث عن ال Asynchronous Sockets ونطبق مجموعة من الأمثلة العملية عليها إنشاء الله.

Copyrighted to: Mr. FADI Abdel-qader, Jordan
Fadi822000@yahoo.com
http://spaces.msn.com/members/csharp2005