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

بسم الله الرحمن الرحيم

الدرس الثالث عشر: Advanced Multicasting Systems :

قمنا سابقا بتعريف ال Multicasting وبينا الفرق بينها وبين ال Broadcasting وبينا أنواعها وكيفية التعامل معها في الدوت نيت وفي هذه الدرس سوف نتحدث عنها بشكل أكثر تفصيلا وذلك لأهميتها الكبيرة في برمجيات الشبكات وخاصة برمجيات ال Conferencing...

أولا : Architecture of Multicast Sockets :

من المعروف انه يتم التعامل مع ال Multicasting عبر برتوكول ال UDP وباستخدام ال Class D Subnet Mask وتتم عملية إدارة المجموعات باستخدام برتوكول ال – IGMP Internet Group Management Protocol والذي هو جزء من ال Internet Protocol Model وكما يتضح من الشكل التالي فإن برتوكول ال IGMP يحتوي على عمليات التحقق من الوصول السليم للبيانات (حيث يتم إرسال حجم البيانات الكلي لرسالة وهي اختيارية إذ يمكن إلغائها بوضع الرقم صفر) ، و تحتوي أيضا على ال TTL Time to Live والذي يحدد فيه العمر الافتراضي لكل رسالة، ونوع العملية الإدارية (ضم إلى مجموعة ، إلغاء من مجموعة ، أو إرجاع معلومات عن المجموعة Membership Query) وأخيرا عنوان المجموعة التي يتم تحديدها برمجيا ضمن ال Range المحدد لل Class D .

8-bit Type	8-bit Max Response Time	16-bit Checksum
32-bit Group Address		

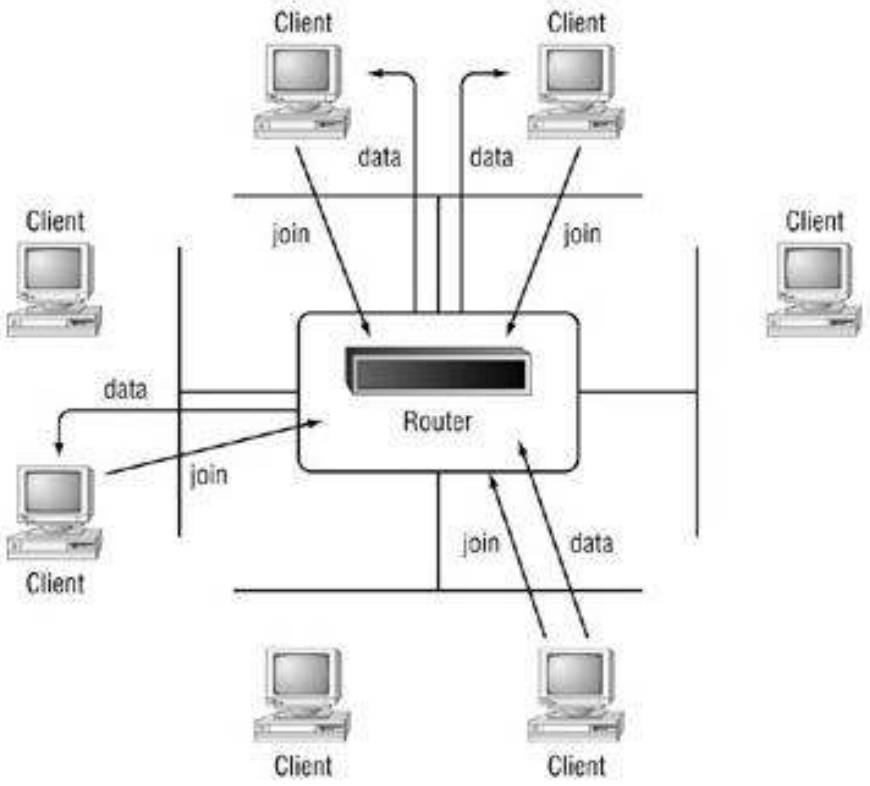
وتم تخصيص ال Range في ال Multicasting من 224.0.0.0 إلى 239.255.255.255 ونستطيع تحديده بثلاثة طرق فإما بشكل يدوي Static أو Dynamic أو على أساس ال Scope-Relative وبشكل عام تستخدم هذه التوزيعات كما يلي كمثال:
التخصيص 224.0.0.1 ويستخدم في جميع الشبكات المحلية فقط حيث لا يتم تمريره إلى شبكة أخرى عبر ال Router أما إذا أردنا التمرير إلى شبكات أخرى عبر ال Router فنستخدم التخصيص 224.0.0.2و لاكن بشرط استخدام نفس ال Subnet في الشبكات الأخرى ... ولمعرفة جميع التخصيصات لل Multicasting انظر الرابط التالي:

http://www.iana.org/assignments/multicast-addresses

يتم نقل ال Multicast Packets بين ال Backbone Tunnels باستخدام ال Unicast Tunnel حيث يتم إرسالها من داخل الشبكة إلى ال Router و ترسل من Router إلى أخر عبر ال Backbone Tunnel باستخدام أسلوب ال Unicast وهو ما يوفر الكثير من ال Bandwidth في الشبكة حيث ترسل نسخة واحدة إلى ال Router ويقوم هو بتوزيعها على الأجهزة

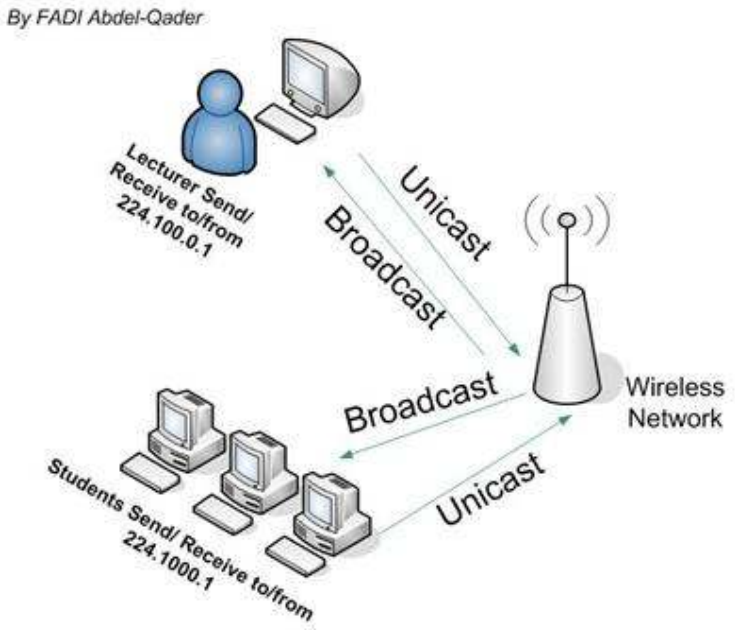
باستخدام ال Unicast المشكلة الوحيدة في ال Multicast هو انه يعتمد بشكل كامل على استخدام ال UDP Connectionless Protocol.

ويمكننا استخدام ال Multicasting في ثلاثة أنواع من الشبكات وهي شبكات ال Peer to Peer حيث لا وجود لجهاز Server والكل يستقبل و يرسل من و إلى ال Group الذي هو فيه، والنوع الثاني Server Based Network حيث يتم إرسال رسالة واحدة إلى ال Server ويقوم ال Server بتوزيعها على بقية الأجهزة في الشبكة ، أما النوع الثالث فيتم من خلال ال Router ، وكما يتضح من الشكل التالي فإن عملية الإرسال تتم بعد انضمام ال Client إلى المجموعة التي تملك ال IP Multicast ويرسل ال Client رسالة واحدة إلى ال Router حيث يقوم ال Router بتوزيعها على الأجهزة في المجموعة مستخدما ال Routing Table.



وكما كان الحال في الإرسال باستخدام ال Broadcasting يتم الإرسال في Multicasting من جهاز محدد إلى مجموعة معينة وليس إلى الكل كما في ال Broadcast ، حيث تُكون كل مجموعة من الأجهزة Group خاص ويتم التخصيص كما ذكرنا سابقا وفق ال IP Multicasting حيث تمتلك كل مجموعة نفس ال IP Multicast ويوجد عدة أشكال لل Multicasting ومن الأمثلة عليها الإرسال إلى مجموعة one to Group و الإرسال إلى أكثر من مجموعة one to Multi Group :

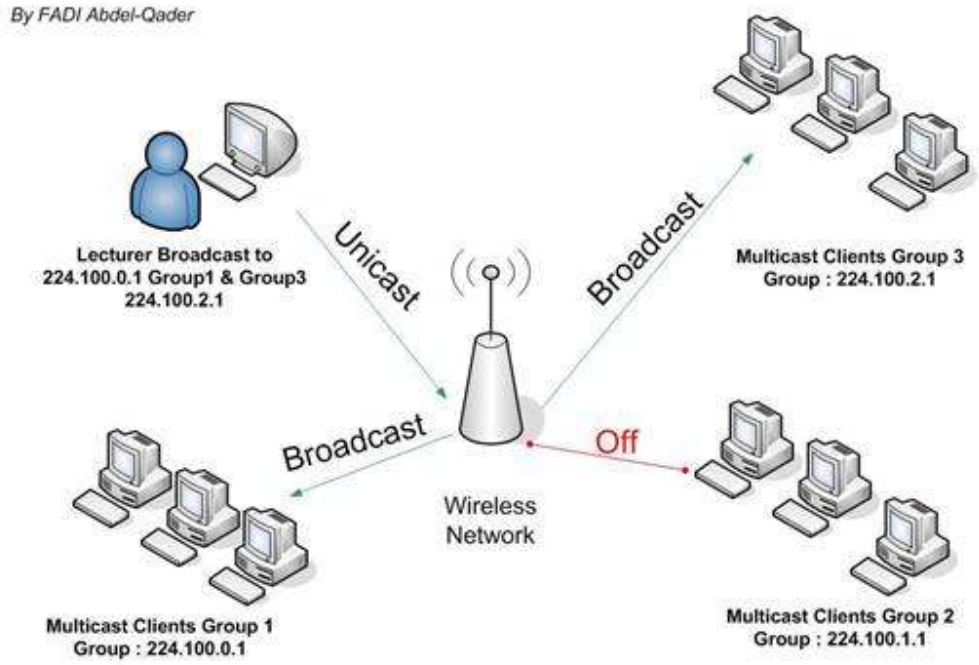
<u>1 – الإرسال مجموعة One to Group:</u>
وفيه يملك ال Sender User نفس ال IP Multicasting الذي يملكه ال Receiver Users ويتم الإرسال من داخل ال Group إلى جميع أعضائه حيث ترسل ك Unicast إلى ال Access Point حيث يقوم بتوزيعها على كافة الأعضاء في المجموعة بأسلوب ال Broadcast وكما في الشكل التالي:



Full Duplex Multicast System for one Group

2- الإرسال إلى أكثر من مجموعة One to Multi-Groups:

وفيه قد يكون ال IP Multicasting لل Sender User مختلف عن Receiver Users ويتم الإرسال من User داخل ال Group إلى المجموعة الذي هو عضو منها وإلى مجموعات أخرى ، ويتم تحديدها باستخدام Address List للمجموعات التي نريد الإرسال لها ...



Half Duplex Multicast System for Multi-Groups

ثانيا: Using Multicast Sockets with .NET:

شرحنا سابقا كيفية التعامل مع ال Multicasting في الدوت نيت وتعرفنا على ال Members وال Classes الخاصة بها وهنا سوف نبين بشيء من التفصيل هذه العمليات ونطبق عليها مجموعة من الأمثلة وبعد ذلك سنقوم ببناء نظام Conference System معتمدا على ال Multicasting ...

من العمليات الأساسية في التعامل مع ال Multicasting :

1- الانضمام أو الخروج من مجموعة Joining || Drop Group :

لا تلزم عملية الانضمام إلى ال Multicast Group أي عمليات تحقق سوى التصنت على ال port وال IP Multicasting المحدد ، ويتم ذلك بعد تعريف udpClient Object وباستخدام ال JoinMulticastGroup Method يتم تعريف ال IP Multicasting الذي سوف ننضم إليه وكما يلي:

```
UdpClient sock = new UdpClient(9050);
sock.JoinMulticastGroup(IPAddress.Parse("225.100.0.1"), 50);
IPEndPoint iep = new IPEndPoint(IPAddress.Any, 0);
```

وكما يلي لإلغاء عملية الانضمام من مجموعة:

```
sock.DropMulticastGroup(IPAddress.Parse("225.100.0.1"));
```

إذ تستخدم ال Methods **JoinMulticastGroup** و **DropMulticastGroup** لضم أو إلغاء عنوان أو مجموعة من العناوين من ال Multicast Group ، وباستخدام Class MulticastOption: يمكننا 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 SetSocketOption حيث تأخذ هذه الميثود ثلاثة باروميترات 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 الباروميتر 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);
```

2- الإرسال إلى مجموعة Sending Data to a Multicast Group:

حتى نستطيع الإرسال باستخدام ال IP Multicasting لابد أولا من تعريف ال Socket Object باستخدام ال UDP Connection وإسناد ال IP Multicasting ورقم ال Port إلى ال IPEndPoint Object ... ونستطيع الإرسال باستخدام ال sendto method حيث نسند لها ال data as Bytes Array وال IPEndPoint Object وكما يلي لإرسال رسالة نصية:

```
Socket server = new Socket(AddressFamily.InterNetwork,SocketType.Dgram,
ProtocolType.Udp);
IPEndPoint iep = new IPEndPoint(IPAddress.Parse(("225.100.0.1"), 9050);
byte[] data = Encoding.ASCII.GetBytes(msg.Text);
server.SendTo(data, iep);
server.Close();
msg.Clear();
msg.Focus();
```

ولإرسال Binary Data كإرسال صورة مثلا لابد من استخدام ال Memory Stream لتخزين الصورة في الذاكرة على هيئة Stream ثم تحويلها إلى Byte Array وبعد ذلك إرسالها باستخدام ال sendto Method وكما يلي:

```
MemoryStream ms = new MemoryStream();
PictureBox1.Image.Save(ms,System.Drawing.Imaging.ImageFormat.Jpeg);
byte[] arrImage = ms.GetBuffer();
ms.Close();
Socket server = new Socket(AddressFamily.InterNetwork,SocketType.Dgram,
ProtocolType.Udp);
IPEndPoint iep = new IPEndPoint(IPAddress.Parse("225.100.0.1"), 5020);
server.SendTo(arrImage,iep);
```

3- الاستقبال من مجموعة Receiving Data From a Multicast Group:

حتى نستطيع الاستقبال من مجموعة لابد أولا من تحديد ال IP Multicast الخاص بالمجموعة و الانضمام إليه ثم استقبال البيانات باستخدام ال Receive Method ويتم ذلك كما يلي لاستقبال رسالة نصية وعرضها في list Box:

```
UdpClient sock = new UdpClient(9050);
sock.JoinMulticastGroup(IPAddress.Parse("225.100.0.1"), 50);
IPEndPoint iep = new IPEndPoint(IPAddress.Any, 0);

byte[] data = sock.Receive(ref iep);
string stringData = Encoding.ASCII.GetString(data, 0, data.Length);
listBox1.Items.Add(iep.Address.ToString() +" :_  "+stringData );
```

ولاستقبال صورة نستخدم ال memory Stream لاستقبال البيانات من ال Receive Method وتخزينها في الذاكرة على هيئة Stream Data ثم تحويلها إلى صورة مرة أخرى باستخدام ال image.FromStream Method وكما يلي:

```
UdpClient sock = new UdpClient(5020);
sock.JoinMulticastGroup(IPAddress.Parse("225.100.0.1"));
IPEndPoint iep = new IPEndPoint(IPAddress.Any, 0);

byte[] data = sock.Receive(ref iep);
MemoryStream ms = new MemoryStream(data);
pictureBox1.Image = Image.FromStream(ms);
sock.Close();
```

ملاحظات هامة في استخدام ال Multicasting في برمجيات الشبكات :

1- من الملاحظ أننا لا نستطيع استخدام ال Network Stream لعملية إرسال ال Multicasting إذ يتطلب استخدامها وجود TCP Socket Connection وهو غير متاح في ال Multicasting ويستعاض عنها باستخدام ال memory Stream لإرسال Binary Stream عبر ال sendto method ...

2- لا يمكنك استخدام ال Multicasting ك loopback في حالة عدم وجود شبكة أو اتصال لذلك لن تستطيع تجربة أي من تطبيقات ال Multicasting في حالة عدم اتصالك بالشبكة.

3- يمكن لكل جهاز أن ينضم إلى أكثر من مجموعة بحيث يستقبل من جهات متعددة، كذلك يستطيع الإرسال إلى عدة مجموعات.

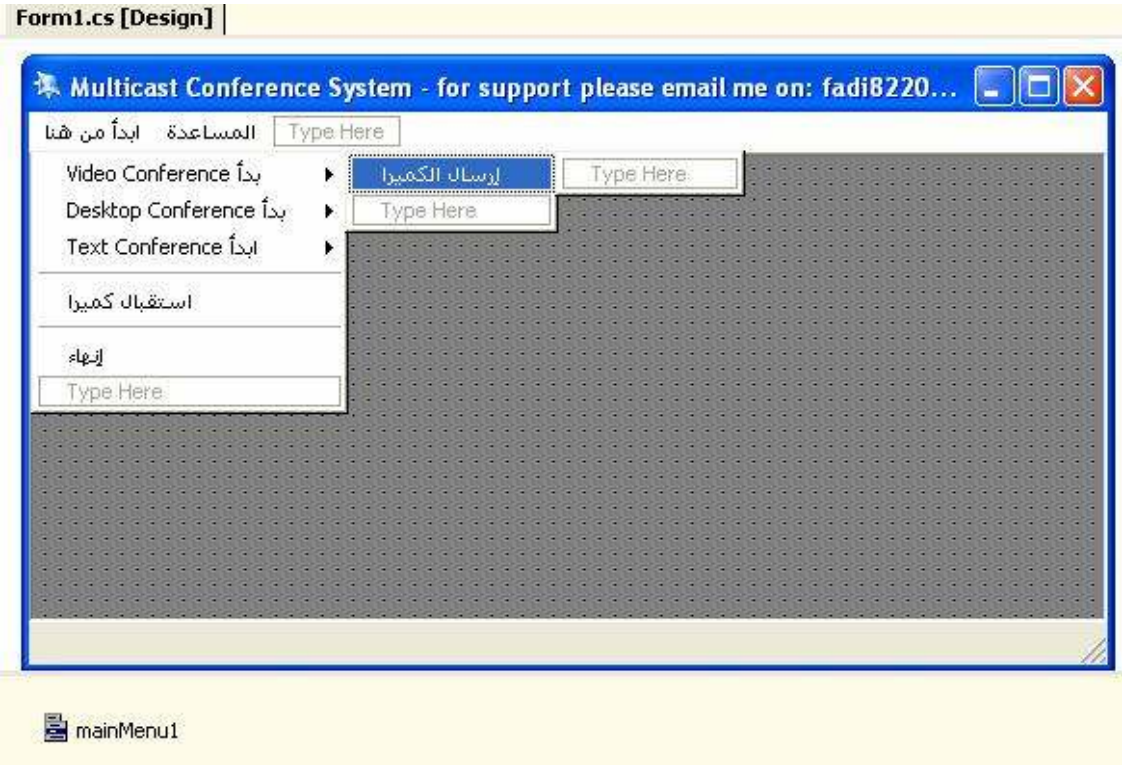
4- في العادة تكون السعة المسموحة لإرسال ال Multicasting Data عبر ال sendto Method محدودة لذلك يمكنك استخدام ال Binary Reader & Writer وال Stream Reader & Writer لإرسال والاستقبال بدلا منها ...

5- تتم عملية اختيار ال IP Multicast وفق لل Network Topology التي تملكها لذلك لابد من التقيد بالعناوين المحددة وهو ما بينته سابقا ..

ثالثا تطبيق مشروع نظام المؤتمرات Multicasting Conferencing Systems:

في هذا التطبيق سوف نفترض وجود غرفة صفية حيث يقوم المحاضر بإلقاء المحاضرة عن بعد أمام طلابه إذ نريد هنا جعل الطلاب يرون الأستاذ وكما يستطيع الأستاذ رؤية طلابه بالإضافة إلى إمكانية عرض المحاضرة على ال Power Point Slides كما يستطيع الطلاب التحدث مع الأستاذ باستخدام Text Chatting ...

سوف نقوم هنا بتقسيم نظام المؤتمرات إلى ثلاثة أنظمة رئيسية وهي نظام مؤتمرات الفيديو ونظام مؤتمرات سطح المكتب ونظام المؤتمرات النصية، في البداية سوف نقوم بعمل الشاشة الرئيسية للبرنامج و كما في الشكل التالي:



1- Full/Half Duplex Multicast Video Conferencing System :

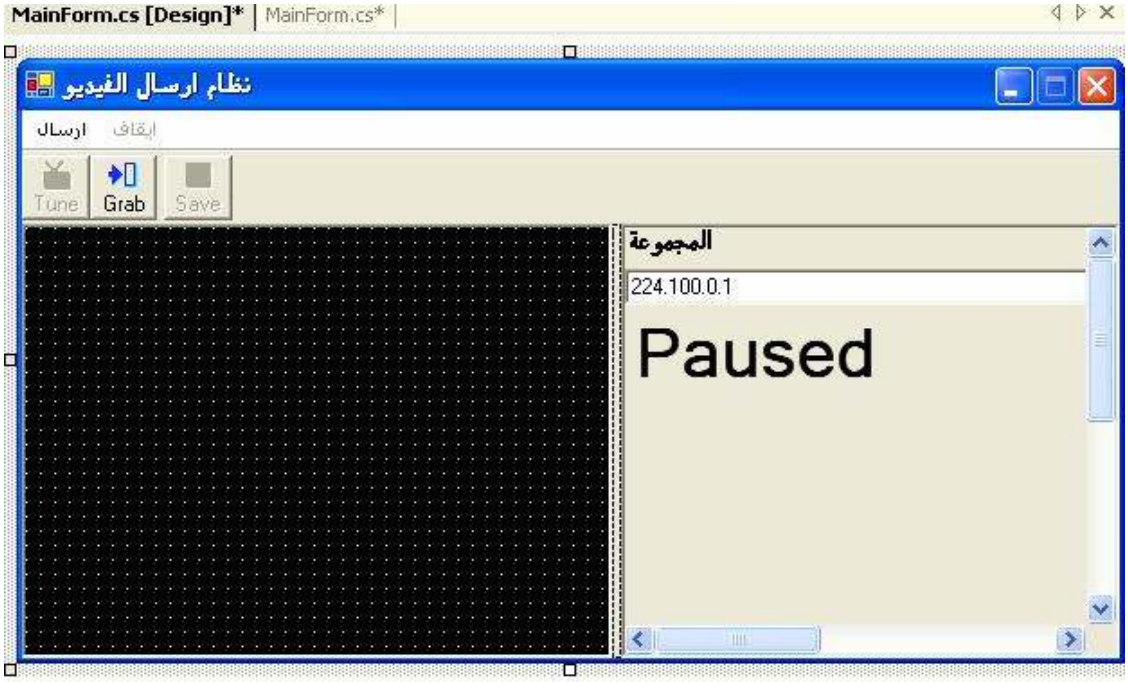
وفرت لنا Microsoft مجموعة من ال Classes الخارجية والتي تتعامل مع ال DirectX 9 مباشرة حيث نستطيع استخدامها لتعامل مع الكاميرا أو ال Scanner أو الصوت أو أي طرفية أخرى وفي هذا التطبيق سوف نستخدم الDirect Show Dot Net Classes لالتقاط صورة عبر الكاميرا وعرضها على ال Picture box حيث نستطيع إرسالها لاحقا إلى ال Multicast Group باستخدام ال memory Stream وال Sendto method وهو ما بيناه سابقا ..

وحتى نستطيع استخدامها سوف نضم ال Direct Show Classes إلى المشروع وكما يلي:

وحتى نتعامل معها سوف نستدعيها باستخدام :

```
using DShowNET;
using DShowNET.Device;
```

وسيكون شكل برنامج الإرسال عبر الكاميرا كما في الشكل التالي:



سوف نستخدم ال DeviceSelector Class لإختيار جهاز الإدخال عند بداية تشغيل البرنامج وكما يلي:

```
DeviceSelector selector = new DeviceSelector( capDevices );
selector.ShowDialog( this );
dev = selector.SelectedDevice;
```

و لإلتاط الصورة عبر الكاميرا سوف نقوم بإنشاء method جديدة كما يلي :

```
void OnCaptureDone()
{
        try {
Trace.WriteLine( "!!DLG: OnCaptureDone" );
toolBarBtnGrab.Enabled = true;
int hr;
if( sampGrabber == null )return;
hr = sampGrabber.SetCallback( null, 0 );
int w = videoInfoHeader.BmiHeader.Width;
int h = videoInfoHeader.BmiHeader.Height;
if( ((w & 0x03) != 0) || (w < 32) || (w > 4096) || (h < 32) || (h > 4096) )
return;
int stride = w * 3;
GCHandle handle = GCHandle.Alloc( savedArray, GCHandleType.Pinned );
int scan0 = (int) handle.AddrOfPinnedObject();
scan0 += (h - 1) * stride;
Bitmap b = new Bitmap( w, h, -stride, PixelFormat.Format24bppRgb, (IntPtr)
scan0 );
handle.Free();
savedArray = null;
Image old = pictureBox.Image;
pictureBox.Image = b;
if( old != null ) old.Dispose();
toolBarBtnSave.Enabled = true;}
        catch( Exception){}
}
```

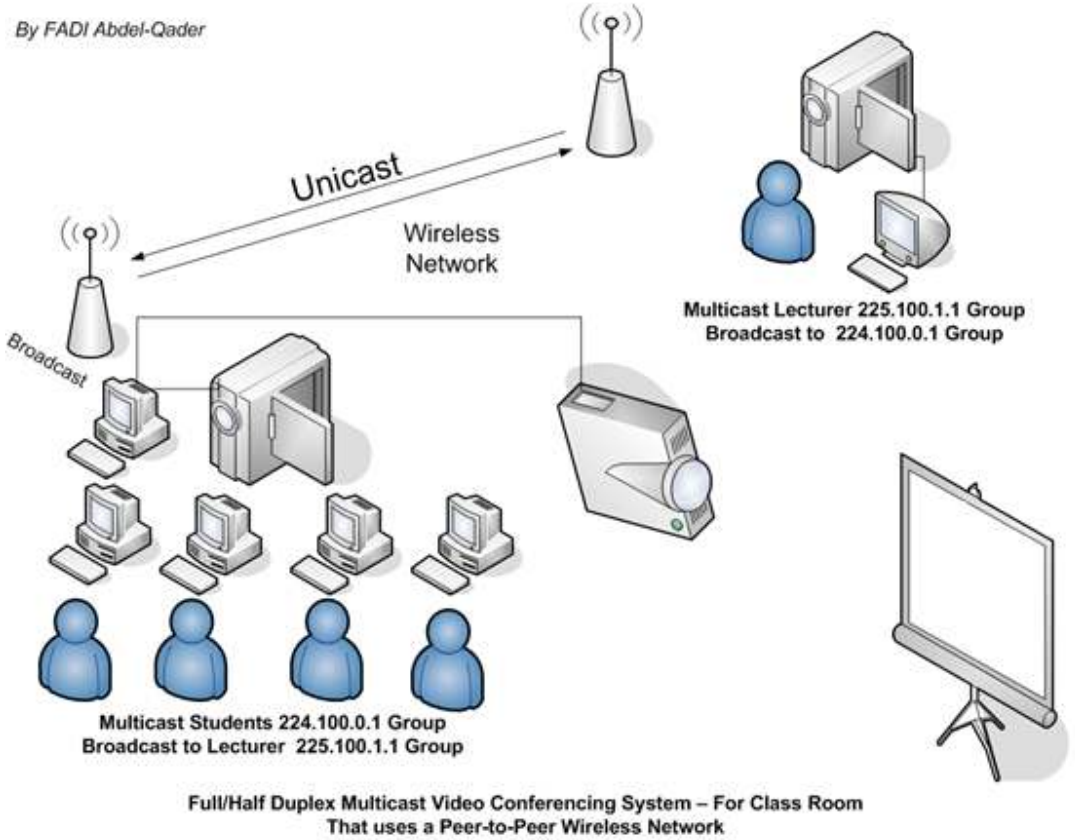
ثم عمل Timer وإضافة الكود التالي فيه لاستمرار عملية التقاط الصورة:

```
int hr;
int size = videoInfoHeader.BmiHeader.ImageSize;
savedArray = new byte[ size + 64000 ];
```

ولإرسال الصورة إلى الطرف الأخر سوف نستخدم method إرسال الصورة ونضعه في Timer وكما يلي:

```
try
{
MemoryStream ms = new MemoryStream();
pictureBox.Image.Save(ms,System.Drawing.Imaging.ImageFormat.Jpeg);
byte[] arrImage = ms.GetBuffer();
ms.Close();
Socket server = new Socket(AddressFamily.InterNetwork,SocketType.Dgram,
ProtocolType.Udp);
IPEndPoint iep = new IPEndPoint(IPAddress.Parse(textBox1.Text), 5020);
server.SendTo(arrImage, iep);
server.Close();}
catch (Exception){}
```

وهنا يستطيع المحاضر إرسال الصورة عبر الكاميرا إلى طلابه كما سوف يتمكن من رؤية طلابه عبر الكاميرا وسوف نفترض هنا استخدامه لشبكة لا سلكية حيث سيرسل البيانات إلى ال Access Point بأسلوب ال Unicast وسوف يتولا ال Access Point توزيع البيانات إلى جميع الأعضاء المنضمين إلى ال Multicast Group ويرسلها لهم باستخدام ال Broadcast وكما في الشكل التالي:



وكما نلاحظ في الشكل السابق فإن المحاضر ينضم إلى مجموعتين مجموعة الأساتذة وهي 225.100.1.1 حيث سيستقبل صورة طلابه عليها، ومجموعة الطلاب 224.100.0.1 والتي سوف يرسل الصورة إليها .. وكما نلاحظ ايضا فإن عملية الإرسال بين ال Access Point1 وال Access Point2 تتم باستخدام ال Unicast ...

وحتى يستطيع الطلاب رؤية أستاذهم والأستاذ رؤية طلابه ، لابد من إنشاء برنامج الاستقبال حيث سنستخدم نفس ال method التي شرحنها سابقا لاستقبال الصورة وللبدء قم بعمل New Form جديد كما في الشكل التالي:

سوف نستخدم ال Namespaces التالية لاستقبال الصورة من ال Multicast Group :

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

ثم قم بكتابة method الاستقبال كما يلي:

```
void Image_Receiver()
{
        UdpClient sock = new UdpClient(5020);
        sock.JoinMulticastGroup(IPAddress.Parse(textBox1.Text));
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 0);

        byte[] data = sock.Receive(ref iep);
        MemoryStream ms = new MemoryStream(data);
        pictureBox1.Image = Image.FromStream(ms);
        sock.Close();

}
```

وحتى نستدعيها لابد من استخدام ال Threading حتى لا يتأثر نظام التشغيل بعملية الاستقبال ، وحتى نقوم بذلك قم بعمل Timer وضع فيه الكود التالي لاستخدام ال Threading :

```
Thread myth;
myth= new Thread  (new System.Threading .ThreadStart(Image_Receiver));
myth.Start ();
```

وحتى تتمكن من تخزين الصورة الملتقطة عبر الكاميرا على هيئة JPEG Image File قم بإنشاء saveFileDialog واستدعيه كما يلي:

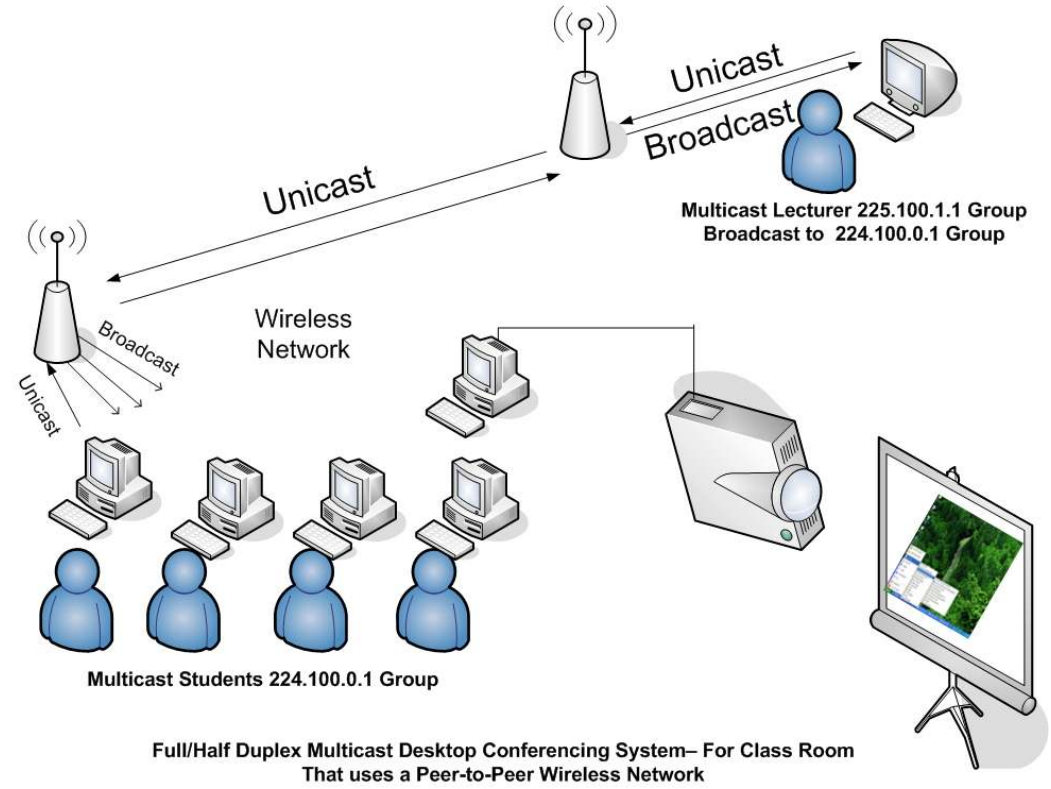
```
try
{

        saveFileDialog1.Filter = "JPEG Image (*.jpg)|*.jpg" ;
        if(saveFileDialog1.ShowDialog() == DialogResult.OK)
                {

string mypic_path = saveFileDialog1.FileName;
pictureBox1.Image.Save(mypic_path);
                }
}
catch (Exception){}
```

وهنا قد تم الانتهاء من المشروع الأول وهو ال Video Conference System ، وحتى يستطيع المحاضر عرض المحاضرة باستخدام برنامج ال Power Point سوف نقوم بعمل مشروع مؤتمرات سطح المكتب ...

2- Full/Half Duplex Multicast Desktop Conferencing System:

الهدف من هذا المشروع هو تمكين الأستاذ من عرض المحاضرة باستخدام برنامج ال Power Point حيث سترسل صورة سطح المكتب من جهاز الأستاذ إلى أجهزة الطلبة ، ولا تختلف عملية الإرسال عن البرنامج السابق في شيء سوى إنشاء Classes لتقوم بالتقاط صورة سطح المكتب ومن ثم إرسالها إلى ال Multicast Group ومن ثم استقبالها وعرضها على الطلاب باستخدام Data Show Projector ...

وهنا مخطط عمل البرنامج :



وكما نلاحظ من الشكل التالي فإن الأستاذ يقوم بشرح المحاضرة على جهازه الشخصي ويرسل الصورة إلى الطلاب وكما نلاحظ أيضا فإن هذه العملية هي أحادية الاتجاه وكما يمكن جعلها باتجاهين Half Duplex || Full لاكن لابد من إنشاء مجموعة جديدة لعملية الإرسال من الطالب إلى الأستاذ حيث يعرض الأستاذ محاضرته ويرسلها إلى مجموعة الطلاب ويستطيع أحد الطلاب عرض جهازه على الأستاذ إذ يرسل الصورة إلى مجموعة الأستاذ ...

ولإنشاء برنامج إرسال صورة سطح المكتب قم بعمل New Form جديد كما في الشكل التالي:

في البداية سوف نقوم بعمل Three Classes لالتقاط صورة سطح المكتب وكما يلي:

أولا PlatFormInvokeGDI32.cs لالتقاط صورة سطح المكتب باستخدام ال GDI+ وال API:

```
using System;
using System.Runtime.InteropServices;
namespace SampleGrabberNET
{

//This class shall keep the GDI32 APIs being used in our program.
        public class PlatformInvokeGDI32
        {

                #region Class Variables
                public  const int SRCCOPY = 13369376;
                #endregion

                #region Class Functions
                [DllImport("gdi32.dll",EntryPoint="DeleteDC")]
                public static extern IntPtr DeleteDC(IntPtr hDc);

                [DllImport("gdi32.dll",EntryPoint="DeleteObject")]
                public static extern IntPtr DeleteObject(IntPtr hDc);

                [DllImport("gdi32.dll",EntryPoint="BitBlt")]
                public static extern bool BitBlt(IntPtr hdcDest,int xDest,int yDest,int
wDest,int hDest,IntPtr hdcSource,int xSrc,int ySrc,int RasterOp);

                [DllImport ("gdi32.dll",EntryPoint="CreateCompatibleBitmap")]
                public static extern IntPtr CreateCompatibleBitmap(IntPtr hdc,
        int nWidth, int nHeight);

                [DllImport ("gdi32.dll",EntryPoint="CreateCompatibleDC")]
                public static extern IntPtr CreateCompatibleDC(IntPtr hdc);

                [DllImport ("gdi32.dll",EntryPoint="SelectObject")]
                public static extern IntPtr SelectObject(IntPtr hdc,IntPtr bmp);
                #endregion

                #region Public Constructor
                public PlatformInvokeGDI32()
                {
                }
                #endregion
        }}
```

ثانيا PlatformInvokeUSER32.cs إذ سوف نستخدمها مع ال Class السابق لالتقاط صورة سطح المكتب باستخدام ال user32 API :

```
using System;
using System.Runtime.InteropServices;
namespace SampleGrabberNET
{
// This class shall keep the User32 APIs being used in our program.
        public class PlatformInvokeUSER32
        {

                #region Class Variables
                public  const int SM_CXSCREEN=0;
                public  const int SM_CYSCREEN=1;
                #endregion

                #region Class Functions
                [DllImport("user32.dll", EntryPoint="GetDesktopWindow")]
                public static extern IntPtr GetDesktopWindow();

                [DllImport("user32.dll",EntryPoint="GetDC")]
                public static extern IntPtr GetDC(IntPtr ptr);

                [DllImport("user32.dll",EntryPoint="GetSystemMetrics")]
                public static extern int GetSystemMetrics(int abc);

                [DllImport("user32.dll",EntryPoint="GetWindowDC")]
                public static extern IntPtr GetWindowDC(Int32 ptr);

                [DllImport("user32.dll",EntryPoint="ReleaseDC")]
                public static extern IntPtr ReleaseDC(IntPtr hWnd,IntPtr hDc);

                #endregion

                #region Public Constructor
                public PlatformInvokeUSER32()
                {
                }
                #endregion
        }

        //This structure shall be used to keep the size of the screen.
        public struct SIZE
        {
                public int cx;
                public int cy;
        }
}
```

ثالثا: CaptureScreen.cs والتي سوف نستخدمها بشكل مباشر في البرنامج حيث يتعامل مع ال PlatFormInvokeGDI32 Class وال PlatformInvokeUSER32 Class :

```
using System;
using System.Drawing;

namespace SampleGrabberNET
{
//This class shall keep all the functionality for capturing the desktop.
        public class CaptureScreen
        {
                #region Public Class Functions
                public static Bitmap GetDesktopImage()
                {
//In size variable we shall keep the size of the screen.
        SIZE size;

//Variable to keep the handle to bitmap.
IntPtr hBitmap;
//Here we get the handle to the desktop device context.
IntPtr hDC =
PlatformInvokeUSER32.GetDC(PlatformInvokeUSER32.GetDesktopWindow());
//Here we make a compatible device context in memory for screen device
context.
IntPtr hMemDC = PlatformInvokeGDI32.CreateCompatibleDC(hDC);
//We pass SM_CXSCREEN constant to GetSystemMetrics to get the X coordinates
of screen.
size.cx=PlatformInvokeUSER32.GetSystemMetrics(PlatformInvokeUSER32.SM_CX
SCREEN);
//We pass SM_CYSCREEN constant to GetSystemMetrics to get the Y coordinates
of screen.
size.cy=PlatformInvokeUSER32.GetSystemMetrics(PlatformInvokeUSER32.SM_CY
SCREEN);
//We create a compatible bitmap of screen size using screen device context.
hBitmap = PlatformInvokeGDI32.CreateCompatibleBitmap(hDC, size.cx, size.cy);
//As hBitmap is IntPtr we can not check it against null. For this purspose
IntPtr.Zero is used.
if (hBitmap!=IntPtr.Zero)
        {
//Here we select the compatible bitmap in memory device context and keeps the
reference to Old bitmap.
IntPtr hOld = (IntPtr) PlatformInvokeGDI32.SelectObject(hMemDC, hBitmap);
//We copy the Bitmap to the memory device context.
PlatformInvokeGDI32.BitBlt(hMemDC, 0, 0,size.cx,size.cy, hDC, 0, 0,
PlatformInvokeGDI32.SRCCOPY);
//We select the old bitmap back to the memory device context.
PlatformInvokeGDI32.SelectObject(hMemDC, hOld);
//We delete the memory device context.
```

```
PlatformInvokeGDI32.DeleteDC(hMemDC);
 //We release the screen device context.
PlatformInvokeUSER32.ReleaseDC(PlatformInvokeUSER32.GetDesktopWindow(),
hDC);//Image is created by Image bitmap handle and stored in local variable.
Bitmap bmp = System.Drawing.Image.FromHbitmap(hBitmap);
//Release the memory to avoid memory leaks.
PlatformInvokeGDI32.DeleteObject(hBitmap);
//This statement runs the garbage collector manually.
GC.Collect();//Return the bitmap
return bmp;
}//If hBitmap is null return null.
return null;
}
#endregion
}
}
```

وحتى نستطيع التحكم في حجم الصورة سوف نكتب ال method التالية:

```
public Bitmap ResizeBitmap( Bitmap b, int nWidth, int nHeight )
{
Bitmap result = new Bitmap( nWidth, nHeight );  using( Graphics g =
Graphics.FromImage( (Image) result ) )    g.DrawImage( b, 0, 0, nWidth,
nHeight );
 return result;
}
```

سوف نستخدم ال Namespaces التالية في البرنامج لتعامل مع ال Multicasting :

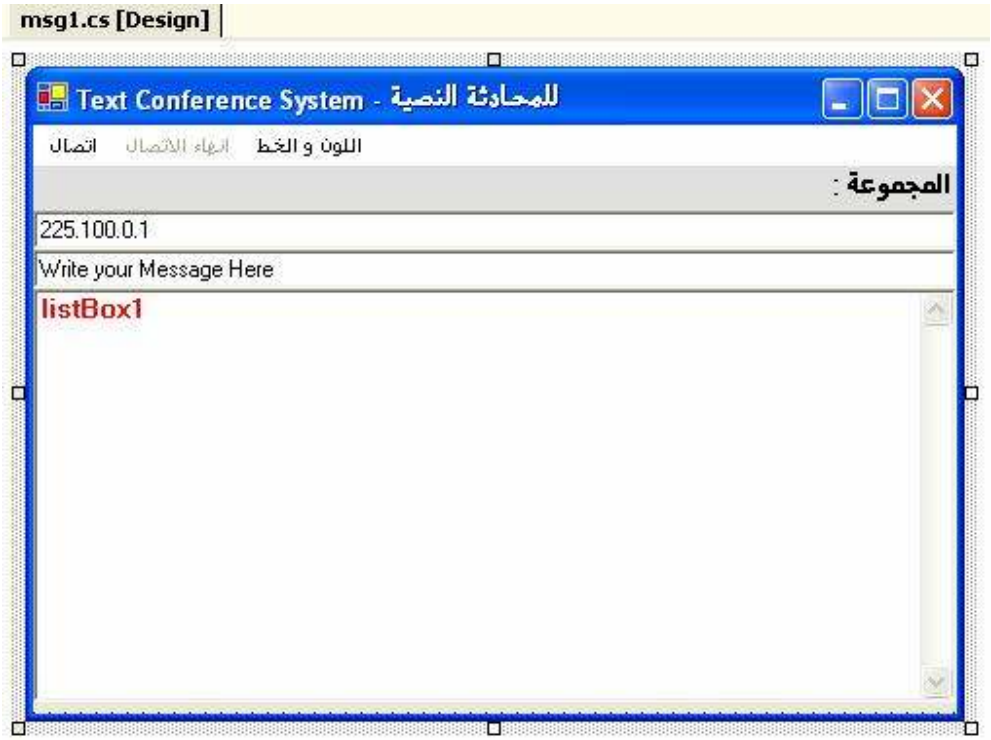
```
using System.Net;
using System.Net.Sockets;
using System.IO;
```

ثم نقوم بعمل Timer لالتقاط صورة سطح المكتب و إرسالها إلى ال Multicast Group المحدد :

```
Bitmap bt = new Bitmap(CaptureScreen.GetDesktopImage());
picScreen.Image = ResizeBitmap(bt, 352, 200 );
MemoryStream ms = new MemoryStream();
picScreen.Image.Save(ms,System.Drawing.Imaging.ImageFormat.Jpeg);
byte[] arrImage = ms.GetBuffer();
ms.Close();
Socket server = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);
IPEndPoint iep = new IPEndPoint(IPAddress.Parse(textBox1.Text), 5020);
server.SendTo(arrImage,iep);
server.Close();
```

:Full/Half Duplex Multicast Text Conferencing System -3

وحتى يستطيع الطلبة التحدث إلى الأستاذ باستخدام ال Text Chat Multicast Conference System سوف نقوم بإنشاء New Form جديد وكما في الشكل التالي:



ثم قم بإضافة ال Namespaces التالية:

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

سوف نستخدم ال method التالية لإجراء عملية الإرسال حيث سترسل الرسالة عند الضغط على ال Enter بعد كتابة الرسالة في ال Textbox المخصص :

```
private void msg_KeyPress(object sender,
System.Windows.Forms.KeyPressEventArgs e)
{if(e.KeyChar == '\r'){
try{
Socket server = new Socket(AddressFamily.InterNetwork,SocketType.Dgram,
ProtocolType.Udp);
IPEndPoint iep = new IPEndPoint(IPAddress.Parse(txt_host.Text), 9050);
  byte[] data = Encoding.ASCII.GetBytes(msg.Text);
server.SendTo(data, iep);
server.Close();
msg.Clear();
msg.Focus();
}catch(Exception){}}}
```

وسوف نستخدم الميثود التالية لعملية الاستقبال حيث ستعرض الرسالة المستقبلة في list Box مخصص:

```
public void server()
{
try
{
UdpClient sock = new UdpClient(9050);
sock.JoinMulticastGroup(IPAddress.Parse(txt_host.Text), 50);
IPEndPoint iep = new IPEndPoint(IPAddress.Any, 0);

byte[] data = sock.Receive(ref iep);
string stringData = Encoding.ASCII.GetString(data, 0, data.Length);
listBox1.Items.Add(iep.Address.ToString() +" :_  "+stringData );
sock.Close();
listBox1.Focus();
msg.Focus();
myth.Abort();
}
catch(Exception){}
}
```

ولاستدعائها لابد من استخدام ال Threading ، قم بعمل Timer واستدعي فيه ال method السابقة باستخدام ال Thread وكما يلي:

```
Thread myth;
myth= new Thread  (new System.Threading .ThreadStart(server));
myth.Start ();
```

سوف نشغل ال Timer عند الضغط على زر الاتصال باستخدام timer1.Enabled = true وفي زر إنهاء الاتصال قم بإضافة الكود التالي:

```
timer1.Enabled = false;
txt_host.ReadOnly = false;
msg.Enabled=false;
        try
        {
        Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
IPEndPoint iep = new IPEndPoint(IPAddress.Parse(txt_host.Text), 9050);
  byte[] data = Encoding.ASCII.GetBytes("has Left the Room");

server.SendTo(data, iep);
server.Close();
msg.Clear();
msg.Focus();

        }
catch(Exception){}
```

وهنا قد تم الانتهاء من الدرس الثالث عشر وفي الدرس التالي سوف نتحدث عن ال Network Security Programming وأساليب تشفير وحماية البيانات واستخدامها في برمجيات الشبكات ...

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