Socket Programming in C#

By:

Dr. Majid Tajeri

Example 1:

```
using System.Net;

namespace test
{
    class Program
    {
        static void Main(string[] args)
        {
            IPAddress test1 = IPAddress.Parse("192.168.0.1");
            IPAddress test2 = IPAddress.Loopback;
            IPAddress test3 = IPAddress.Broadcast;
            IPAddress test4 = IPAddress.Any;
            IPAddress test5 = IPAddress.None;
            IPHostEntry ihe =
Dns.GetHostByName(Dns.GetHostName());
            IPAddress myself = ihe.AddressList[0];
            if (IPAddress.IsLoopback(test2))
  Console.WriteLine("The Loopback address is:{0}",
test2.ToString());
            else
  Console.WriteLine("Error obtaining the loopback address");
  Console.WriteLine("The local IP address is  :{0}\n",
myself.ToString());

            if (myself == test2)
Console.WriteLine("The loopback address is the A same as local
address", myself.ToString());
            else
Console.WriteLine("loopback address is not  the local
address.\n");

Console.WriteLine("the test address is :{0}",
test1.ToString());

Console.WriteLine("broadcast address:{0}", test3.ToString());
Console.WriteLine("the any address is:{0}", test4.ToString());

Console.WriteLine("the none address is:{0}",
test5.ToString());

        }
    }
}
```

Example 2:

```
IPAddress test = IPAddress.Parse("192.168.1.1");
IPEndPoint ie = new IPEndPoint(test, 8000);
Console.WriteLine("the IPEndPoint is : {0}", ie.ToString());
Console.WriteLine("the Address Family is :{0}",
ie.AddressFamily);
Console.WriteLine("the address is: {0}, And the A port is
:{1}\n", ie.Address, ie.Port);
Console.WriteLine("the min port number is :{0}",
IPEndPoint.MinPort);
Console.WriteLine("the max port number is :
{0}",IPEndPoint.MaxPort);
 ie.Port = 80;
Console.WriteLine("The change IPEndPoint value a is:{0}",
ie.ToString());
SocketAddress sa = ie.Serialize();
Console.WriteLine("the socket address is:{0}", sa.ToString());
```

Example 3:

```
using System.Net;
namespace test2
{
    class Program
    {
        static void Main(string[] args)
        {
  IPHostEntry iphost = Dns.GetHostEntry("127.0.0.1");
  string hostname = iphost.HostName;
   Console.WriteLine(hostname);
        }
    }
}
```

Example 4:

```
using System;
using System.Text;
using System.Net;

namespace test3
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter a Computer Name:");
            IPHostEntry iphost;
            string s;
            try
            {
                s = Console.ReadLine();
                iphost = Dns.GetHostEntry(s);
                IPAddress[] addresses = iphost.AddressList;
           StringBuilder addressList = new StringBuilder();
                foreach (IPAddress address in addresses)
                {
    addressList.AppendFormat("IP Address: {0};",
address.ToString());

                }
                Console.WriteLine(addressList);
            }
            catch (Exception)
            {
               Console.WriteLine("Host Not Found...");


             }
        }
    }
}
```

Example 5:

```
using System.Net;
using System.Net.Sockets;
namespace SockProp
{
    class Program
    {
        static void Main(string[] args)
        {
            IPAddress ia = IPAddress.Parse("127.0.0.1");
            IPEndPoint ie = new IPEndPoint(ia, 8000);

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

Console.WriteLine("Socket type: {0}", test.SocketType);
Console.WriteLine("ProtocolType: {0}", test.ProtocolType);
test.Blocking=false;
Console.WriteLine("Blocking:{0}", test.Blocking);
Console.WriteLine("Connected:{0}", test.Connected);
test.Bind(ie);
IPEndPoint iep = (IPEndPoint)test.LocalEndPoint;
Console.WriteLine("LocalEndPoint:{0}", iep.ToString());
            test.Close();

        }
    }
}
```

Example 6:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.Net.Sockets;
namespace Server
{
    class Program
    {
        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[2048];
         IPEndPoint ipep = new IPEndPoint(IPAddress.Any,
9050);

            Socket newsock = new
Socket(AddressFamily.InterNetwork, SocketType.Stream,
ProtocolType.Tcp);
            newsock.Bind(ipep);
            newsock.Listen(10);
            Console.WriteLine("Waiting for a client...");
      Socket client = newsock.Accept();
      IPEndPoint clientep = (IPEndPoint)client.RemoteEndPoint;
            Console.WriteLine("Connected with {0} at port
{1}", clientep.Address, clientep.Port);
            string welcome = "Welcome to my test server";
            data = Encoding.ASCII.GetBytes(welcome);
            client.Send(data, data.Length, SocketFlags.None);
            while (true)
            {
                data = new byte[1024];
                recv = client.Receive(data);
                if (recv == 0)
                    break;
         Console.WriteLine(Encoding.ASCII.GetString(data, 0,
recv));
                client.Send(data, recv, SocketFlags.None);
            }
   Console.WriteLine("Disconnected from
{0}",clientep.Address);
            client.Close();
            newsock.Close();
        }
    }
}
```

Example 7:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.Net.Sockets;
namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[2048];
            string input, stringData;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new
Socket(AddressFamily.InterNetwork,SocketType.Stream,
ProtocolType.Tcp);
            try
            {
                server.Connect(ipep);
            }
            catch (SocketException e)
            {
             Console.WriteLine("Unable to connect to server.");
                Console.WriteLine(e.ToString());
                return;
            }
            int recv = server.Receive(data);
          stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")
                    break;
                server.Send(Encoding.ASCII.GetBytes(input));
                data = new byte[1024];
                recv = server.Receive(data);
          stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Disconnecting from server...");
            server.Shutdown(SocketShutdown.Both);
            server.Close();
        }
    }
}
```

Example 8:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.Net.Sockets;
namespace SimpleUdpSrvr
{
    class Program
    {
        static void Main(string[] args)
        {
            int recv;
            byte[] data = new byte[1024];
          IPEndPoint ipep = new IPEndPoint(IPAddress.Any, 9050);
          Socket newsock = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

        newsock.Bind(ipep);
        Console.WriteLine("Waiting for a client...");
        IPEndPoint sender = new IPEndPoint(IPAddress.Any, 0);
              EndPoint Remote = (EndPoint)(sender);

              recv = newsock.ReceiveFrom(data, ref Remote);

Console.WriteLine("Message received from {0}:",
Remote.ToString());

     Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
              string welcome = "Welcome to my test server";
              data = Encoding.ASCII.GetBytes(welcome);
newsock.SendTo(data, data.Length, SocketFlags.None, Remote);

              while (true)
              {
                  data = new byte[1024];
                  recv = newsock.ReceiveFrom(data, ref Remote);
Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
newsock.SendTo(data, recv, SocketFlags.None, Remote);
              }
        }
    }
}
```

Example 9:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.Net.Sockets;
namespace SimpleUdpClient
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string input, stringData;
IPEndPoint ipep = new IPEndPoint(IPAddress.Parse("127.0.0.1"),
9050);
            Socket server = new
Socket(AddressFamily.InterNetwork, SocketType.Dgram,
ProtocolType.Udp);
            string welcome = "Hello, are you there?";
            data = Encoding.ASCII.GetBytes(welcome);
 server.SendTo(data, data.Length, SocketFlags.None, ipep);
 IPEndPoint sender = new IPEndPoint(IPAddress.Any, 0);
            EndPoint Remote = (EndPoint)sender;
            data = new byte[1024];
            int recv = server.ReceiveFrom(data, ref Remote);
Console.WriteLine("Message received from {0}:",
Remote.ToString());
   Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")
                    break;
      server.SendTo(Encoding.ASCII.GetBytes(input), Remote);
                data = new byte[1024];

                recv = server.ReceiveFrom(data, ref Remote);
   stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Stopping client");
            server.Close();
        }
    }
}
```

Example 10:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class BadUdpClient
{
    public static void Main()
    {
        byte[] data = new byte[30];
        string input, stringData;
        IPEndPoint ipep = new IPEndPoint(
        IPAddress.Parse("127.0.0.1"), 9050);
        Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        string welcome = "Hello, are you there?";
        data = Encoding.ASCII.GetBytes(welcome);

  server.SendTo(data, data.Length, SocketFlags.None, ipep);
        IPEndPoint sender = new IPEndPoint(IPAddress.Any, 0);
        EndPoint tmpRemote = (EndPoint)sender;
        data = new byte[30];
        int recv = server.ReceiveFrom(data, ref tmpRemote);
Console.WriteLine("Message received from {0}:",
tmpRemote.ToString());
Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
        while (true)
        {
            input = Console.ReadLine();
            if (input == "exit")
                break;
   server.SendTo(Encoding.ASCII.GetBytes(input), tmpRemote);
            data = new byte[30];
            recv = server.ReceiveFrom(data, ref tmpRemote);
     stringData = Encoding.ASCII.GetString(data, 0, recv);
            Console.WriteLine(stringData);
        }
        Console.WriteLine("Stopping client");
        server.Close();
    }
}
```

Example 11:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class BetterdUdpClient
{
    public static void Main()
    {
        byte[] data = new byte[30];
        string input, stringData;
        IPEndPoint ipep = new IPEndPoint(
        IPAddress.Parse("127.0.0.1"), 9050);
        Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        string welcome = "Hello, are you there?";
        data = Encoding.ASCII.GetBytes(welcome);
server.SendTo(data, data.Length, SocketFlags.None, ipep);
        IPEndPoint sender = new IPEndPoint(IPAddress.Any, 0);
        EndPoint tmpRemote = (EndPoint)sender;
        data = new byte[30];
        int recv = server.ReceiveFrom(data, ref tmpRemote);
        Console.WriteLine("Message received from {0}:",
tmpRemote.ToString());
Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
        int i = 30;
        while (true)
        {
            input = Console.ReadLine();
            if (input == "exit")
                break;
            int a;
            a = input.Length;
    server.SendTo(Encoding.ASCII.GetBytes(input), tmpRemote);
            data = new byte[i];
            try
            {
    recv = server.ReceiveFrom(data, ref tmpRemote);
    stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            catch (SocketException)
            {
  Console.WriteLine("WARNING: data lost, retry message.");
                int v;
                v = a - i;
                i +=v ;
            }
        }
        Console.WriteLine("Stopping client");
        server.Close();
    }
}
```

Example 12:

```
using System;
using System.Text;
using System.Net;
using System.Net.Sockets;
namespace TimeoutUdpClient
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new
Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
   int sockopt =
(int)server.GetSocketOption(SocketOptionLevel.Socket,SocketOpt
ionName.ReceiveTimeout);
Console.WriteLine("Default timeout: {0}", sockopt);
server.SetSocketOption(SocketOptionLevel.Socket,
SocketOptionName.ReceiveTimeout, 3000);
sockopt =
(int)server.GetSocketOption(SocketOptionLevel.Socket,SocketOpt
ionName.ReceiveTimeout);
            Console.WriteLine("New timeout: {0}", sockopt);
            string welcome = "Hello, are you there?";
            data = Encoding.ASCII.GetBytes(welcome);
  server.SendTo(data, data.Length, SocketFlags.None, ipep);
  IPEndPoint sender = new IPEndPoint(IPAddress.Any, 0);
            EndPoint tmpRemote = (EndPoint)sender;
            data = new byte[1024];
            recv = server.ReceiveFrom(data, ref tmpRemote);
Console.WriteLine("Message received from
{0}:",tmpRemote.ToString());
Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")
                    break;
server.SendTo(Encoding.ASCII.GetBytes(input), tmpRemote);
                data = new byte[1024];
         recv = server.ReceiveFrom(data, ref tmpRemote);
         stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            Console.WriteLine("Stopping client");
            server.Close();
        }
    }
}
```

Example 13:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.Net.Sockets;
namespace ExceptionUdpClient
{
    class Program
    {
        static void Main(string[] args)
        {
            byte[] data = new byte[1024];
            string input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
                IPAddress.Parse("127.0.0.1"), 9050);

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


 int sockopt =
(int)server.GetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout);

Console.WriteLine("Default timeout: {0}", sockopt);
            server.SetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout, 3000);
            sockopt =
(int)server.GetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout);

            Console.WriteLine("New timeout: {0}", sockopt);
            string welcome = "Hello, are you there?";
            data = Encoding.ASCII.GetBytes(welcome);
    server.SendTo(data, data.Length, SocketFlags.None, ipep);
    IPEndPoint sender = new IPEndPoint(IPAddress.Any, 0);
            EndPoint Remote = (EndPoint)sender;

            data = new byte[1024];
            try
            {
                recv = server.ReceiveFrom(data, ref Remote);
Console.WriteLine("Message received from {0}:",
Remote.ToString());
Console.WriteLine(Encoding.ASCII.GetString(data, 0, recv));
            }
```

```
                catch (SocketException)
                {

         Console.WriteLine("Problem communicating with remote
        server");
                    return;
                }
                while (true)
                {
                    input = Console.ReadLine();
                    if (input == "exit")
                        break;
               server.SendTo(Encoding.ASCII.GetBytes(input), ipep);
                    data = new byte[1024];
                    try
                    {
          recv = server.ReceiveFrom(data, ref Remote);
         stringData = Encoding.ASCII.GetString(data, 0, recv);
                        Console.WriteLine(stringData);
                    }
                    catch (SocketException)
                    {
                 Console.WriteLine("Error receiving message");
                    }
                }
                Console.WriteLine("Stopping client");
                server.Close();
            }
        }
    }
```

Example 14:

```
private int SndRcvData(Socket s, byte[] message, EndPoint
rmtdevice)
    {
        int recv;
        int retry = 0;
        while (true)
        {
            Console.WriteLine("Attempt #{0}", retry);
            try
            {
                s.SendTo(message, message.Length,
SocketFlags.None, rmtdevice);
                data = new byte[1024];
                recv = s.ReceiveFrom(data, ref Remote);
            }
            catch (SocketException)
            {
                recv = 0;
            }
            if (recv > 0)
            {
                return recv;
            }
            else
            {
                retry++;
                if (retry > 4)
                {
                    return 0;
                }
            }
        }

 }
```

Example 15:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class RetryUdpClient
{
    private byte[] data = new byte[1024];
 private static IPEndPoint sender = new
IPEndPoint(IPAddress.Any, 0);
    private static EndPoint Remote = (EndPoint)sender;

 private int SndRcvData(Socket s, byte[] message, EndPoint
rmtdevice)
    {
        int recv;
        int retry = 0;
        while (true)
        {
            Console.WriteLine("Attempt #{0}", retry);
            try
            {
s.SendTo(message, message.Length, SocketFlags.None,
rmtdevice);
                data = new byte[1024];
                recv = s.ReceiveFrom(data, ref Remote);
            }

            catch (SocketException)
            {
                recv = 0;
            }

            if (recv > 0)
            {
                return recv;
            }
            else
            {
                retry++;
                if (retry > 4)
                {
                    return 0;
                }
            }
        }
    }
```

```
        public RetryUdpClient()
        {
            string input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
            int sockopt =
    (int)server.GetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout);
            Console.WriteLine("Default timeout: {0}", sockopt);
            server.SetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout, 3000);

            sockopt =
    (int)server.GetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout);

            Console.WriteLine("New timeout: {0}", sockopt);
            string welcome = "Hello, are you there?";
            data = Encoding.ASCII.GetBytes(welcome);
            recv = SndRcvData(server, data, ipep);
            if (recv > 0)
            {
            stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            else
            {
    Console.WriteLine("Unable to communicate with remote host");
                return;
            }
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")
                    break;
    recv = SndRcvData(server, Encoding.ASCII.GetBytes(input),
    ipep);
                if (recv > 0)
                {
             stringData = Encoding.ASCII.GetString(data, 0, recv);
                    Console.WriteLine(stringData);
                }
                else
               Console.WriteLine("Did not receive an answer");
            }
            Console.WriteLine("Stopping client");
            server.Close();
        }
        public static void Main()
        {
            RetryUdpClient ruc = new RetryUdpClient();
        }
    }
```

Example 16:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class BestUdpClient
{
    private byte[] data = new byte[1024];
private static IPEndPoint sender = new
IPEndPoint(IPAddress.Any, 0);
    private static EndPoint Remote = (EndPoint)sender;
    private static int size = 30;
    private static int AdvSndRcvData(Socket s, byte[] message,
    EndPoint rmtdevice)
    {
        int recv = 0;
        int retry = 0;
        while (true)
        {
            Console.WriteLine("Attempt #{0}", retry);
            try
            {
    s.SendTo(message, message.Length, SocketFlags.None,
rmtdevice);
                data = new byte[size];
                recv = s.ReceiveFrom(data, ref Remote);
            }
            catch (SocketException e)
            {
                if (e.ErrorCode == 10054)
                    recv = 0;
                else if (e.ErrorCode == 10040)
                {
           Console.WriteLine("Error receiving packet");
                    size += 10;
                    recv = 0;
                }
            }
            if (recv > 0)
            {
                return recv;
            }
            else
            {
                retry++;
                if (retry > 4)
                {
                    return 0;
                }
            }
        }
    }
```

```
        public static void Main()
        {
            string input, stringData;
            int recv;
            IPEndPoint ipep = new IPEndPoint(
            IPAddress.Parse("127.0.0.1"), 9050);
            Socket server = new Socket(AddressFamily.InterNetwork,
            SocketType.Dgram, ProtocolType.Udp);
    int sockopt =
    (int)server.GetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout);

            Console.WriteLine("Default timeout: {0}", sockopt);
            server.SetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout, 3000);
            sockopt =
    (int)server.GetSocketOption(SocketOptionLevel.Socket,
            SocketOptionName.ReceiveTimeout);

            Console.WriteLine("New timeout: {0}", sockopt);
            string welcome = "Hello, are you there?";
            data = Encoding.ASCII.GetBytes(welcome);
            recv = AdvSndRcvData(server, data, ipep);

            if (recv > 0)
            {
             stringData = Encoding.ASCII.GetString(data, 0, recv);
                Console.WriteLine(stringData);
            }
            else
            {
      Console.WriteLine("Unable to communicate with remote host");
                return;
            }
            while (true)
            {
                input = Console.ReadLine();
                if (input == "exit")

                    break;
    recv = AdvSndRcvData(server, Encoding.ASCII.GetBytes(input),
    ipep);

                if (recv > 0)
                {
                    stringData = Encoding.ASCII.GetString(data, 0,
    recv);
                    Console.WriteLine(stringData);
                }
                else
                    Console.WriteLine("Did not receive an
    answer");
            }
            Console.WriteLine("Stopping client");
            server.Close();
        }
    }
```

Example 17:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class BadBroadcast
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram,
        ProtocolType.Udp);
  IPEndPoint iep = new IPEndPoint(IPAddress.Broadcast, 9050);
byte[] data = Encoding.ASCII.GetBytes("This is a test
message");
        sock.SendTo(data, iep);
        sock.Close();
    }
}
```

Example 18:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class Broadcst
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram,
        ProtocolType.Udp);
IPEndPoint iep1 = new IPEndPoint(IPAddress.Broadcast, 9050);
        IPEndPoint iep2 = new
IPEndPoint(IPAddress.Parse("192.168.0.255"), 9050);
        string hostname = Dns.GetHostName();
        byte[] data = Encoding.ASCII.GetBytes(hostname);

sock.SetSocketOption(SocketOptionLevel.Socket,SocketOptionName
.Broadcast, 1);

        sock.SendTo(data, iep1);
        sock.SendTo(data, iep2);
        sock.Close();
        Console.ReadKey();
    }
}
```

Example 19:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class RecvBroadcst
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
        sock.Bind(iep);
        EndPoint ep = (EndPoint)iep;
        Console.WriteLine("Ready to receive ");
        byte[] data = new byte[1024];
        int recv = sock.ReceiveFrom(data, ref ep);
  string stringData = Encoding.ASCII.GetString(data, 0, recv);
Console.WriteLine("received: {0} from: {1}", stringData,
ep.ToString());

        data = new byte[1024];
        recv = sock.ReceiveFrom(data, ref ep);
        stringData = Encoding.ASCII.GetString(data, 0, recv);
Console.WriteLine("received: {0} from: {1}",stringData,
ep.ToString());

        sock.Close();
        Console.ReadKey();
    }
}
```

Example 20:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class MultiRecv
{
    public static void Main()
    {
        Socket sock = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        Console.WriteLine("Ready to receive ");
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
        EndPoint ep = (EndPoint)iep;
        sock.Bind(iep);
        sock.SetSocketOption(SocketOptionLevel.IP,
        SocketOptionName.AddMembership,
        new MulticastOption(IPAddress.Parse("224.100.0.1")));
        byte[] data = new byte[1024];
        int recv = sock.ReceiveFrom(data, ref ep);
        string stringData = Encoding.ASCII.GetString(data, 0,
recv);
        Console.WriteLine("received: {0} from: {1}",
stringData, ep.ToString());
        sock.Close();
        Console.ReadKey();
    }
}
```

Example 21:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class MultiSend
{
    public static void Main()
    {
        Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        IPEndPoint iep = new
IPEndPoint(IPAddress.Parse("224.100.0.1"), 9050);
        byte[] data = Encoding.ASCII.GetBytes("This is a test
message");
        server.SendTo(data, iep);
        server.Close();
        Console.ReadKey();
    }
}
```

Example 22:

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class NewMultiSend
{
    public static void Main()
    {
        Socket server = new Socket(AddressFamily.InterNetwork,
        SocketType.Dgram, ProtocolType.Udp);
        IPEndPoint iep = new IPEndPoint(IPAddress.Any, 9050);
        IPEndPoint iep2 = new
IPEndPoint(IPAddress.Parse("224.100.0.1"), 9050);
        server.Bind(iep);
        byte[] data = Encoding.ASCII.GetBytes("This is a test
message");
        server.SetSocketOption(SocketOptionLevel.IP,
SocketOptionName.AddMembership,
        new MulticastOption(IPAddress.Parse("224.100.0.1")));
        server.SetSocketOption(SocketOptionLevel.IP,
        SocketOptionName.MulticastTimeToLive, 50);
        server.SendTo(data, iep2);
        server.Close();
   }
}
```