UdpClient Class

Reference

Definition

Namespace: System.Net.Sockets

Assembly: System.Net.Sockets.dll

Provides User Datagram Protocol (UDP) network services.

In this article

C# Copy

```
public class UdpClient : IDisposable
```

Inheritance Object → UdpClient

Implements IDisposable

Examples

The following example establishes a UdpClient connection using the host name `www.contoso.com` on port 11000. A small string message is sent to two separate remote host machines. The Receive method blocks execution until a message is received. Using the IPEndPoint passed to Receive, the identity of the responding host is revealed.

C# Copy

```csharp
// This constructor arbitrarily assigns the local port number.
UdpClient udpClient = new UdpClient(11000);
    try{
         udpClient.Connect("www.contoso.com", 11000);

         // Sends a message to the host to which you have connected.
         Byte[] sendBytes = Encoding.ASCII.GetBytes("Is anybody there?");

         udpClient.Send(sendBytes, sendBytes.Length);

         // Sends a message to a different host using optional hostname and
port parameters.
         UdpClient udpClientB = new UdpClient();
         udpClientB.Send(sendBytes, sendBytes.Length,
"AlternateHostMachineName", 11000);

         //IPEndPoint object will allow us to read datagrams sent from any
source.
         IPEndPoint RemoteIpEndPoint = new IPEndPoint(IPAddress.Any, 0);

         // Blocks until a message returns on this socket from a remote
host.
         Byte[] receiveBytes = udpClient.Receive(ref RemoteIpEndPoint);
         string returnData = Encoding.ASCII.GetString(receiveBytes);

         // Uses the IPEndPoint object to determine which of these two hosts
responded.
         Console.WriteLine("This is the message you received " +
                                      returnData.ToString());
         Console.WriteLine("This message was sent from " +
                                     RemoteIpEndPoint.Address.ToString() +
                                     " on their port number " +
                                     RemoteIpEndPoint.Port.ToString());

          udpClient.Close();
          udpClientB.Close();
          }
       catch (Exception e ) {
                  Console.WriteLine(e.ToString());
        }
```

Remarks

The UdpClient class provides simple methods for sending and receiving connectionless UDP datagrams in blocking synchronous mode. Because UDP is a connectionless transport protocol, you do not need to establish a remote host connection prior to sending and receiving data. You do, however, have the option of establishing a default remote host in one of the following two ways:

- Create an instance of the UdpClient class using the remote host name and port number as parameters.
- Create an instance of the UdpClient class and then call the Connect method.

You can use any of the send methods provided in the UdpClient to send data to a remote device. Use the Receive method to receive data from remote hosts.

Note

Do not call **Send** using a host name or **IPEndPoint** if you have already specified a default remote host. If you do, **UdpClient** will throw an exception.

UdpClient methods also allow you to send and receive multicast datagrams. Use the JoinMulticastGroup method to subscribe a UdpClient to a multicast group. Use the DropMulticastGroup method to unsubscribe a UdpClient from a multicast group.

Constructors

UdpClient()	Initializes a new instance of the UdpClient class.
UdpClient(AddressFamily)	Initializes a new instance of the UdpClient class.
UdpClient(Int32)	Initializes a new instance of the UdpClient class and binds it to the local port number provided.
UdpClient(Int32, AddressFamily)	Initializes a new instance of the UdpClient class and binds it to the local port number provided.
UdpClient(IPEndPoint)	Initializes a new instance of the UdpClient class and binds it to the specified local endpoint.
UdpClient(String, Int32)	Initializes a new instance of the UdpClient class and establishes a default remote host.

Properties

Active	Gets or sets a value indicating whether a default remote host has been established.
Available	Gets the amount of data received from the network that is available to read.
Client	Gets or sets the underlying network Socket.
DontFragment	Gets or sets a Boolean value that specifies whether the UdpClient allows Internet Protocol (IP) datagrams to be fragmented.
EnableBroadcast	Gets or sets a Boolean value that specifies whether the UdpClient may send broadcast packets.
ExclusiveAddressUse	Gets or sets a Boolean value that specifies whether the UdpClient allows only one client to use a port.
MulticastLoopback	Gets or sets a Boolean value that specifies whether outgoing multicast packets are delivered to the sending application.
Ttl	Gets or sets a value that specifies the Time to Live (TTL) value of Internet Protocol (IP) packets sent by the UdpClient.

Methods

AllowNatTraversal(Boolean)	Enables or disables Network Address Translation (NAT) traversal on a UdpClient instance.
BeginReceive(AsyncCallback, Object)	Receives a datagram from a remote host asynchronously.
BeginSend(Byte[], Int32, AsyncCallback, Object)	Sends a datagram to a remote host asynchronously. The destination was specified previously by a call to Connect.
BeginSend(Byte[], Int32, IPEndPoint, AsyncCallback, Object)	Sends a datagram to a destination asynchronously. The destination is specified by a EndPoint.
BeginSend(Byte[], Int32, String, Int32, AsyncCallback, Object)	Sends a datagram to a destination asynchronously. The destination is specified by the host name and port number.

Close()	Closes the UDP connection.
Connect(IPAddress, Int32)	Establishes a default remote host using the specified IP address and port number.
Connect(IPEndPoint)	Establishes a default remote host using the specified network endpoint.
Connect(String, Int32)	Establishes a default remote host using the specified host name and port number.
Dispose()	Releases the managed and unmanaged resources used by the UdpClient.
Dispose(Boolean)	Releases the unmanaged resources used by the UdpClient and optionally releases the managed resources.
DropMulticast Group(IPAddress)	Leaves a multicast group.
DropMulticast Group(IPAddress, Int32)	Leaves a multicast group.
EndReceive(IAsyncResult, IPEndPoint)	Ends a pending asynchronous receive.
EndSend(IAsyncResult)	Ends a pending asynchronous send.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
JoinMulticastGroup(Int32, IPAddress)	Adds a UdpClient to a multicast group.
JoinMulticastGroup(IPAddress)	Adds a UdpClient to a multicast group.
JoinMulticastGroup(IPAddress, Int32)	Adds a UdpClient to a multicast group with the specified Time to Live (TTL).
JoinMulticastGroup(IPAddress, IPAddress)	Adds a UdpClient to a multicast group.

MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Receive(IPEndPoint)	Returns a UDP datagram that was sent by a remote host.
ReceiveAsync()	Returns a UDP datagram asynchronously that was sent by a remote host.
ReceiveAsync(CancellationToken)	Returns a UDP datagram asynchronously that was sent by a remote host.
Send(Byte[], Int32)	Sends a UDP datagram to a remote host.
Send(Byte[], Int32, IPEndPoint)	Sends a UDP datagram to the host at the specified remote endpoint.
Send(Byte[], Int32, String, Int32)	Sends a UDP datagram to a specified port on a specified remote host.
Send(ReadOnlySpan<Byte>)	Sends a UDP datagram to a remote host.
Send(ReadOnlySpan<Byte>, IPEndPoint)	Sends a UDP datagram to the host at the specified remote endpoint.
Send(ReadOnlySpan<Byte>, String, Int32)	Sends a UDP datagram to a specified port on a specified remote host.
SendAsync(Byte[], Int32)	Sends a UDP datagram asynchronously to a remote host.
SendAsync(Byte[], Int32, IPEndPoint)	Sends a UDP datagram asynchronously to a remote host.
SendAsync(Byte[], Int32, String, Int32)	Sends a UDP datagram asynchronously to a remote host.
SendAsync(ReadOnlyMemory<Byte>, CancellationToken)	Sends a UDP datagram asynchronously to a remote host.
SendAsync(ReadOnlyMemory<Byte>, IPEndPoint, CancellationToken)	Sends a UDP datagram asynchronously to a remote host.
SendAsync(ReadOnlyMemory<Byte>, String, Int32, CancellationToken)	Sends a UDP datagram asynchronously to a remote host.

ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET	Core 1.0, Core 1.1, Core 2.0, Core 2.1, Core 2.2, Core 3.0, Core 3.1, 5, 6, 7 Preview 1
.NET Framework	1.1, 2.0, 3.0, 3.5, 4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8
.NET Standard	1.3, 1.4, 1.6, 2.0, 2.1
Xamarin.iOS	10.8
Xamarin.Mac	3.0

See also

- TcpClient
- TCP-UDP

Recommended content

UdpClient.Send Method (System.Net.Sockets)

Sends a UDP datagram to a remote host.

UdpClient.Receive(IPEndPoint) Method (System.Net.Sockets)

Returns a UDP datagram that was sent by a remote host.

UdpClient.JoinMulticastGroup Method (System.Net.Sockets)

Adds a UdpClient to a multicast group.

UdpClient.BeginReceive(AsyncCallback, Object) Method (System.Net.Sockets)

Receives a datagram from a remote host asynchronously.

UdpClient Constructor (System.Net.Sockets)

Initializes a new instance of the UdpClient class.

UdpClient.Connect Method (System.Net.Sockets)

Establishes a default remote host.

Socket Class (System.Net.Sockets)

Implements the Berkeley sockets interface.

UdpClient.ReceiveAsync Method (System.Net.Sockets)

Returns a UDP datagram asynchronously that was sent by a remote host.

Show more