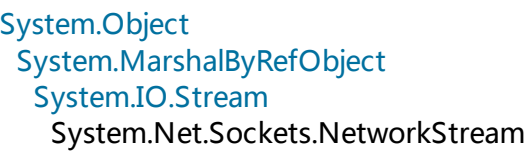


# NetworkStream 类

.NET Framework 4

提供网络访问的基础数据流。

## 继承层次结构



命名空间：[System.Net.Sockets](#)  
程序集：System (在 System.dll 中)

## 语法

C#

```
public class NetworkStream : Stream
```


NetworkStream 类型公开以下成员。






## 构造函数

显示: ☒ 继承 ☒ 保护

	名称	说明
	<a href="#">NetworkStream(Socket)</a>	为指定的 <a href="#">Socket</a> 创建 NetworkStream 类的新实例。
	<a href="#">NetworkStream(Socket, Boolean)</a>	用指定的 <a href="#">Socket</a> 所有权为指定的 <a href="#">Socket</a> 初始化 NetworkStream 类的新实例。
	<a href="#">NetworkStream(Socket, FileAccess)</a>	用指定的访问权限为指定的 <a href="#">Socket</a> 创建 NetworkStream 类的新实例。
	<a href="#">NetworkStream(Socket, FileAccess, Boolean)</a>	用指定的访问权限和指定的 <a href="#">Socket</a> 所有权为指定的 <a href="#">Socket</a> 创建 NetworkStream 类的新实例。

属性





显示: ☒ 继承 ☒ 保护 










	名称	说明
	<a href="#">CanRead</a>	获取一个值，该值指示 <code>NetworkStream</code> 是否支持读取。（重写 <a href="#">Stream.CanRead</a> 。）
	<a href="#">CanSeek</a>	获取一个值，该值指示流是否支持查找。当前不支持此属性，它始终返回 <b>false</b> 。（重写 <a href="#">Stream.CanSeek</a> 。）
	<a href="#">CanTimeout</a>	指示超时属性是否可用于 <code>NetworkStream</code> 。（重写 <a href="#">Stream.CanTimeout</a> 。）
	<a href="#">CanWrite</a>	获取一个值，该值指示 <code>NetworkStream</code> 是否支持写入。（重写 <a href="#">Stream.CanWrite</a> 。）
	<a href="#">DataAvailable</a>	获取一个值，该值指示在要读取的 <code>NetworkStream</code> 上是否有可用的数据。
	<a href="#">Length</a>	获取流上可用数据的长度。此属性当前不受支持，总是引发 <a href="#">NotSupportedException</a> 。（重写 <a href="#">Stream.Length</a> 。）
	<a href="#">Position</a>	获取或设置流中的当前位置。此属性当前不受支持，总是引发 <a href="#">NotSupportedException</a> 。（重写 <a href="#">Stream.Position</a> 。）
	<a href="#">Readable</a>	获取或设置一个值，该值指示 <code>NetworkStream</code> 是否可以读取。
	<a href="#">ReadTimeout</a>	获取或设置读取操作阻止等待数据的时间量。（重写 <a href="#">Stream.ReadTimeout</a> 。）
	<a href="#">Socket</a>	获取基础 <a href="#">Socket</a> 。
	<a href="#">Writeable</a>	获取一个值，该值指示 <code>NetworkStream</code> 是否可写。
	<a href="#">WriteTimeout</a>	获取或设置写入操作阻止等待数据的时间量。（重写 <a href="#">Stream.WriteTimeout</a> 。）

方法

显示: ☒ 继承 ☒ 保护 

	名称	说明
--	----	----

	BeginRead	从 NetworkStream 开始异步读取。（重写 Stream.BeginRead(Byte[], Int32, Int32, AsyncCallback, Object)。）
	BeginWrite	开始向流异步写入。（重写 Stream.BeginWrite(Byte[], Int32, Int32, AsyncCallback, Object)。）
	Close()	关闭当前流并释放与之关联的所有资源（如套接字和文件句柄）。（继承自 Stream。）
	Close(Int32)	等待指定的时间以便发送数据之后，关闭 NetworkStream。
	CopyTo(Stream)	从当前流中读取字节并将其写入到目标流中。（继承自 Stream。）
	CopyTo(Stream, Int32)	从当前流中读取所有字节并将其写入到目标流中（使用指定的缓冲区大小）。（继承自 Stream。）
	CreateObjRef	创建一个对象，该对象包含生成用于与远程对象进行通信的代理所需的全部相关信息。（继承自 MarshalByRefObject。）
	CreateWaitHandle	<b>已过时。</b> 分配 WaitHandle 对象。（继承自 Stream。）
	Dispose()	释放由 Stream 占用的所有资源。（继承自 Stream。）
	Dispose(Boolean)	释放由 NetworkStream 占用的非托管资源，还可以另外再释放托管资源。（重写 Stream.Dispose(Boolean)。）
	EndRead	处理异步读取的结束。（重写 Stream.EndRead(IAsyncResult)。）
	EndWrite	处理异步写入的结束。（重写 Stream.EndWrite(IAsyncResult)。）
	Equals(Object)	确定指定的 Object 是否等于当前的 Object。（继承自 Object。）
	Finalize	释放由 NetworkStream 使用的所有资源。（重写 Object.Finalize()。）
	Flush	刷新流中的数据。保留此方法供将来使用。（重写 Stream.Flush()。）
	GetHashCode	用作特定类型的哈希函数。（继承自 Object。）

	<a href="#">GetLifetimeService</a>	检索控制此实例的生存期策略的当前生存期服务对象。（继承自 <a href="#">MarshalByRefObject</a> 。）
	<a href="#">GetType</a>	获取当前实例的 <a href="#">Type</a> 。（继承自 <a href="#">Object</a> 。）
	<a href="#">InitializeLifetimeService</a>	获取控制此实例的生存期策略的生存期服务对象。（继承自 <a href="#">MarshalByRefObject</a> 。）
	<a href="#">MemberwiseClone()</a>	创建当前 <a href="#">Object</a> 的浅表副本。（继承自 <a href="#">Object</a> 。）
	<a href="#">MemberwiseClone(Boolean)</a>	创建当前 <a href="#">MarshalByRefObject</a> 对象的浅表副本。（继承自 <a href="#">MarshalByRefObject</a> 。）
	<a href="#">ObjectInvariant</a>	基础结构。提供对 <a href="#">Contract</a> 的支持。（继承自 <a href="#">Stream</a> 。）
	<a href="#">Read</a>	从 <a href="#">NetworkStream</a> 读取数据。（重写 <a href="#">Stream.Read(Byte[], Int32, Int32)</a> 。）
	<a href="#">ReadByte</a>	从流中读取一个字节，并将流内的位置向前推进一个字节，如果已到达流的末尾，则返回 -1。（继承自 <a href="#">Stream</a> 。）
	<a href="#">Seek</a>	将流的当前位置设置为给定值。此方法当前不受支持，总是引发 <a href="#">NotSupportedException</a> 。（重写 <a href="#">Stream.Seek(Int64, SeekOrigin)</a> 。）
	<a href="#">SetLength</a>	设置流的长度。此方法始终引发 <a href="#">NotSupportedException</a> 。（重写 <a href="#">Stream.SetLength(Int64)</a> 。）
	<a href="#">ToString</a>	返回表示当前对象的字符串。（继承自 <a href="#">Object</a> 。）
	<a href="#">Write</a>	将数据写入 <a href="#">NetworkStream</a> 。（重写 <a href="#">Stream.Write(Byte[], Int32, Int32)</a> 。）
	<a href="#">WriteByte</a>	将一个字节写入流内的当前位置，并将流内的位置向前推进一个字节。（继承自 <a href="#">Stream</a> 。）

[页首](#)

## 备注

[NetworkStream](#) 类提供在阻止模式下通过 [Stream](#) 套接字发送和接收数据的方法。有关阻止与非阻止 [Socket](#) 的更多信息，请参见[Using an Asynchronous Client Socket](#)。可以在同步和异步数据传输时使用 [NetworkStream](#) 类。有关同步和异步通信的更多信息，请参见 [Sockets](#)。

若要创建 `NetworkStream`，必须提供连接的 `Socket`。也可指定 `NetworkStream` 对所提供的 `Socket` 具有哪些 `FileAccess` 权限。默认情况下，关闭 `NetworkStream` 并不会关闭所提供的 `Socket`。如果希望 `NetworkStream` 具有关闭所提供的 `Socket` 的权限，必须将 `ownsSocket` 参数的值指定为 `true`。

将 `Write` 和 `Read` 方法用于简单的单线程同步阻止 I/O。若要使用不同的线程来处理 I/O，则应考虑使用 `BeginWrite` 和 `EndWrite` 方法，或 `BeginRead` 和 `EndRead` 方法进行通信。

`NetworkStream` 不支持对网络数据流的随机访问。`CanSeek` 属性用于指示流是否支持查找，它的值始终为 `false`；读取 `Position` 属性、读取 `Length` 属性或者调用 `Seek` 方法都会引发 `NotSupportedException`。

可以对 `NetworkStream` 类的实例同时执行读和写操作而无需同步。只要写操作有一个唯一线程，读操作有一个唯一线程，则读和写线程之间不会存在交叉引用，因而无需同步。

## 示例

下面的代码示例演示如何从连接的 `StreamSocket` 创建 `NetworkStream` 并执行基本同步阻止 I/O。

C#

```
// Examples for constructors that do not specify file permission.

// Create the NetworkStream for communicating with the remote host.
NetworkStream myNetworkStream;

if (networkStreamOwnsSocket){
    myNetworkStream = new NetworkStream(mySocket, true);
}
else{
    myNetworkStream = new NetworkStream(mySocket);
}
```

## 版本信息

.NET Framework

受以下版本支持：4、3.5、3.0、2.0、1.1、1.0

.NET Framework Client Profile

受以下版本支持：4、3.5 SP1

## 平台

Windows 7, Windows Vista SP1 或更高版本, Windows XP SP3, Windows XP SP2 x64 Edition, Windows Server 2008 (不支持服务器核心), Windows Server 2008 R2 (支持 SP1 或更高版本的服务器核心), Windows Server 2003 SP2

.NET Framework 并不是对每个平台的所有版本都提供支持。有关支持的版本的列表，请参见[.NET Framework 系统要求](#)。

## 线程安全

此类型的任何公共 **static**（在 Visual Basic 中为 **Shared**）成员都是线程安全的。但不保证所有实例成员都是线程安全的。

## 请参见

参考

[System.Net.Sockets](#) 命名空间  
[TcpClient](#)

---

## 社区附加资源

---