Lidgren Network Library documentation
Lidgren.Network Namespace
Send Feedback
Lidgren Network Library
## Classes

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetAESEncryption</td>
<td>AES encryption</td>
</tr>
<tr>
<td>NetBitVector</td>
<td>Fixed size vector of booleans</td>
</tr>
<tr>
<td>NetBitWriter</td>
<td>Helper class for NetBuffer to write/read bits</td>
</tr>
<tr>
<td>NetBlockEncryptionBase</td>
<td>Base for a non-threadsafe encryption class</td>
</tr>
<tr>
<td>NetBuffer</td>
<td>Specialized version of NetPeer used for a &quot;client&quot; connection. It does not accept any incoming connections and maintains a ServerConnection property</td>
</tr>
<tr>
<td>NetClient</td>
<td>Represents a connection to a remote peer</td>
</tr>
<tr>
<td>NetConnection</td>
<td>Represents a connection to a remote peer</td>
</tr>
<tr>
<td>NetConnectionStatistics</td>
<td>Statistics for a NetConnection instance</td>
</tr>
<tr>
<td>NetDESEncryption</td>
<td>DES encryption</td>
</tr>
<tr>
<td>NetException</td>
<td>Exception thrown in the Lidgren Network Library</td>
</tr>
<tr>
<td>NetIncomingMessage</td>
<td>Incoming message either sent from a remote peer or generated within the library</td>
</tr>
<tr>
<td>NetOutgoingMessage</td>
<td>Outgoing message used to send data to remote peer(s)</td>
</tr>
<tr>
<td>NetPeer</td>
<td>Represents a local peer capable of holding zero, one or more connections to remote peers</td>
</tr>
<tr>
<td>NetPeerConfiguration</td>
<td>Partly immutable after NetPeer has been initialized</td>
</tr>
<tr>
<td>NetPeerStatistics</td>
<td>Statistics for a NetPeer instance</td>
</tr>
<tr>
<td>NetQueue&lt;Of&lt;('T')&gt;&gt;</td>
<td>Thread safe (blocking) expanding queue with TryDequeue() and EnqueueFirst()</td>
</tr>
<tr>
<td>NetRandom</td>
<td>A fast random number generator for .NET</td>
</tr>
<tr>
<td>NetRC2Encryption</td>
<td>RC2 encryption</td>
</tr>
<tr>
<td>NetServer</td>
<td>Specialized version of NetPeer used for</td>
</tr>
<tr>
<td>Library</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>NetSRP</strong></td>
<td>Helper methods for implementing SRP authentication</td>
</tr>
<tr>
<td><strong>NetTime</strong></td>
<td>Time service</td>
</tr>
<tr>
<td><strong>NetTripleDES Encryption</strong></td>
<td>Triple DES encryption</td>
</tr>
<tr>
<td><strong>NetUPnP</strong></td>
<td>UPnP support class</td>
</tr>
<tr>
<td><strong>NetUtility</strong></td>
<td>Utility methods</td>
</tr>
<tr>
<td><strong>NetXorEncryption</strong></td>
<td>Example class; not very good encryption</td>
</tr>
<tr>
<td><strong>NetXtea</strong></td>
<td>Methods to encrypt and decrypt data using the XTEA algorithm</td>
</tr>
<tr>
<td>Structure</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SingleUIntUnion</td>
<td></td>
</tr>
</tbody>
</table>
## Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INetEncryption</td>
<td>Interface for an encryption algorithm</td>
</tr>
</tbody>
</table>
# Delegates

<table>
<thead>
<tr>
<th>Delegate</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetUtility::ResolveAddressCallback</td>
<td></td>
</tr>
<tr>
<td>NetUtility::ResolveEndPointCallback</td>
<td></td>
</tr>
</tbody>
</table>
## Enumerations

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NetConnectionStatus</code></td>
<td>Status for a NetConnection instance</td>
</tr>
<tr>
<td><code>NetDeliveryMethod</code></td>
<td>How the library deals with resends and handling of late messages</td>
</tr>
<tr>
<td><code>NetIncomingMessageType</code></td>
<td>The type of a NetIncomingMessage</td>
</tr>
<tr>
<td><code>NetPeerStatus</code></td>
<td>Status for a NetPeer instance</td>
</tr>
<tr>
<td><code>NetSendResult</code></td>
<td>Result of a SendMessage call</td>
</tr>
</tbody>
</table>
Interface for an encryption algorithm

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#
public interface INetEncryption

Visual Basic
Public Interface INetEncryption

Visual C++
public interface class INetEncryption
See Also

INetEncryption Members
Lidgren.Network Namespace
The **INetEncryption** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt an incoming message in place</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt an outgoing message in place</td>
</tr>
</tbody>
</table>
See Also

INetEncryption Interface
Lidgren.Network Namespace
The **INetEncryption** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt an incoming message in place</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt an outgoing message in place</td>
</tr>
</tbody>
</table>
See Also

INetEncryption Interface
Lidgren.Network Namespace
Lidgren Network Library documentation
INetEncryption..::..Decrypt Method
INetEncryption Interface See Also Send Feedback

Decrypt an incoming message in place

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
bool Decrypt(NetIncomingMessage msg)
```

### Visual Basic

```vbnet
Function Decrypt(msg As NetIncomingMessage) As Boolean
```

### Visual C++

```cpp
bool Decrypt(NetIncomingMessage^ msg)
```

## Parameters

**msg**

Type: `Lidgren.Network.INetIncomingMessage`


## Return Value

See Also

INetEncryption Interface
Lidgren.Network Namespace
Encrypt an outgoing message in place

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
bool Encrypt(
    NetOutgoingMessage msg
)
```

Visual Basic

```vbnet
Function Encrypt ( _
    msg As NetOutgoingMessage _
) As Boolean
```

Visual C++

```cpp
bool Encrypt(
    NetOutgoingMessage^ msg
)
```

Parameters

msg
Type: Lidgren.Network.INetEncryption

[Missing <param name="msg"/> documentation for 
]

Return Value

[Missing <returns> documentation for 

See Also

INetEncryption Interface
Lidgren.Network Namespace
Lidgren Network Library documentation

NetAESEncryption Class

Members See Also Send Feedback

AES encryption

Namespace: Lidgren.Network
Syntax

C#

public class NetAESScryption : INetEncryption

Visual Basic

Public Class NetAESScryption _
   Implements INetEncryption

Visual C++

public ref class NetAESScryption : INetEncryption
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetAESEncryption
See Also

NetAESEncryption Members
Lidgren.Network Namespace
The **NetAESEncryption** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetAESEncryption(String)</td>
<td>NetAESEncryption constructor</td>
</tr>
<tr>
<td>NetAESEncryption(array&lt;Byte&gt;[][], array&lt;Byte&gt;[][])</td>
<td>NetAESEncryption constructor</td>
</tr>
<tr>
<td>NetAESEncryption(String, Int32)</td>
<td>NetAESEncryption constructor</td>
</tr>
</tbody>
</table>
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

NetAESEncryption Class
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetAESEncryption Constructor

NetAESEncryption Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetAESEncryption(String)</td>
<td>NetAESEncryption constructor</td>
</tr>
<tr>
<td>NetAESEncryption(array&lt;Byte&gt;[], array&lt;Byte&gt;[])</td>
<td>NetAESEncryption constructor</td>
</tr>
<tr>
<td>NetAESEncryption(String, Int32)</td>
<td>NetAESEncryption constructor</td>
</tr>
</tbody>
</table>
See Also

NetAESCryption Class
NetAESCryption Members
Lidgren.Network Namespace
NetAESShading Constructor (String)

Namespace: Lidgren.Network
### Syntax

**C#**

```csharp
public NetAESEncryption(
    string key
)
```

**Visual Basic**

```vbscript
Public Sub New (_
    key As String _
)
```

**Visual C++**

```csharp
public:
NetAESEncryption(
    String^ key
)
```

### Parameters

**key**

Type: `System::String`

See Also

NetAESEncryption Class
NetAESEncryption Overload
Lidgren.Network Namespace
NetAESEncryption Constructor (array<Byte>[0][0], array<Byte>[0][0])

NetAESEncryption Class See Also Send Feedback

NetAESEncryption constructor

Namespace: Lidgren.Network
Syntax

C#

public NetAESEncryption(
    byte[] key,
    byte[] iv
)

Visual Basic

Public Sub New (_
    key As Byte(), _
    iv As Byte() _
)

Visual C++

public:
NetAESEncryption(
    array<unsigned char>^ key,
    array<unsigned char>^ iv
)

Parameters

key
Type: array<System::::Byte>[]()[[]]

[Missing <param name="key"/> documentation for 
]

iv
Type: array<System::::Byte>[]()[[]]

[Missing <param name="iv"/> documentation for 
]
See Also

NetAESEncryption Class
NetAESEncryption Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetAESEncryption Constructor (String, Int32)

NetAESEncryption Class See Also Send Feedback

NetAESEncryption constructor

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public NetAESEncryption(
    string key,
    int bitsize
)
```

### Visual Basic

```vbasic
Public Sub New (
    key As String,
    bitsize As Integer
)
```

### Visual C++

```cpp
public:
NetAESEncryption(
    String^ key,
    int bitsize
)
```

## Parameters

### key

Type: `System::String`


### bitsize

Type: `System::Int32`


See Also

NetAESEncryption Class
NetAESEncryption Overload
Lidgren.Network Namespace
The NetAESEncryption type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

NetAESCryptography Class
Lidgren.Network Namespace
Decrypt incoming message

**Namespace:** [Lidgren.Network](http://Lidgren.Network)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll)  
Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool Decrypt(NetIncomingMessage msg)

Visual Basic

Public Function Decrypt(_
msg As NetIncomingMessage _) As Boolean

Visual C++

public:
virtual bool Decrypt(_
NetIncomingMessage^ msg
) sealed

Parameters

msg
Type: Lidgren.Network..::..NetIncomingMessage


Return Value


Implements

INetEncryption..::..Decrypt(NetIncomingMessage)
See Also

NetAESScryption Class
Lidgren.Network Namespace
Encrypt outgoing message

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public bool Encrypt(NetOutgoingMessage msg)
```

**Visual Basic**

```vbnet
Public Function Encrypt(_
    msg As NetOutgoingMessage _) As Boolean
```

**Visual C++**

```cpp
public:
    virtual bool Encrypt(_
        NetOutgoingMessage^ msg
    ) sealed
```

**Parameters**

`msg`

Type: `Lidgren.Network.NetOutgoingMessage`

[Missing `<param name="msg"/>` documentation for

**Return Value**

[Missing `<returns>` documentation for

**Implements**

`INetEncryption.NetAESEncryption.Encrypt(NetOutgoingMessage)`
See Also

NetAESEncryption Class
Lidgren.Network Namespace
NetBitVector Class

Members  See Also  Send Feedback

Fixed size vector of booleans

Namespace: Lidgren.Network
(2012.1.7.0)
Syntax

C#
public sealed class NetBitVector

Visual Basic
Public NotInheritable Class NetBitVector

Visual C++
public ref class NetBitVector sealed
Inheritance Hierarchy

System:::Object
Lidgren.Network:::NetBitVector
See Also

NetBitVector Members
Lidgren.Network Namespace
NetBitVector Members

NetBitVector Class Constructors Methods Properties See Also Send Feedback

The NetBitVector type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetBitVector</td>
<td>NetBitVector constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear</td>
<td>Sets all bits/booleans to zero/false</td>
</tr>
<tr>
<td>Count</td>
<td>Returns the number of bits/booleans set to one/true</td>
</tr>
<tr>
<td>Determines whether the specified Object is equal to the current Object.</td>
<td></td>
</tr>
<tr>
<td>(Inherited from Object.)</td>
<td></td>
</tr>
<tr>
<td>Equals</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td>(Inherited from Object.)</td>
<td></td>
</tr>
<tr>
<td>Finalize</td>
<td>Gets the bit/bool at the specified index</td>
</tr>
<tr>
<td>(Inherited from Object.)</td>
<td></td>
</tr>
<tr>
<td>Get</td>
<td>Gets the first (lowest) index set to true</td>
</tr>
<tr>
<td>GetFirstSetIndex</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td>(Inherited from Object.)</td>
<td></td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>(Inherited from Object.)</td>
<td></td>
</tr>
<tr>
<td>GetType</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td>(Inherited from Object.)</td>
<td></td>
</tr>
<tr>
<td>IsEmpty</td>
<td>Shift all bits one step down, cycling the first bit to the top</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Returns true if all bits/booleans are set to zero/false</td>
</tr>
<tr>
<td>(Inherited from Object.)</td>
<td></td>
</tr>
<tr>
<td>RotateDown</td>
<td>Sets or clears the bit/bool at the specified index</td>
</tr>
<tr>
<td>Set</td>
<td>Returns a string that represents this object</td>
</tr>
<tr>
<td>(Overrides Object..ToString).</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Gets the bit/bool at the specified index</td>
</tr>
<tr>
<td>Capacity</td>
<td>Gets the number of bits/booleans stored in this vector</td>
</tr>
</tbody>
</table>
See Also

NetBitVector Class
Lidgren.Network Namespace
NetBitVector constructor

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0  
(2012.1.7.0)
**Syntax**

C#

```csharp
public NetBitVector(
    int bitsCapacity
)
```

Visual Basic

```vbnet
Public Sub New ( _
    bitsCapacity As Integer _
)
```

Visual C++

```cpp
public:
NetBitVector(
    int bitsCapacity
)
```

**Parameters**

bitsCapacity
Type: System::System::Int32

See Also

NetBitVector Class
Lidgren.Network Namespace
The `NetBitVector` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clear</strong></td>
<td>Sets all bits/booleans to zero/false</td>
</tr>
<tr>
<td><strong>Count</strong></td>
<td>Returns the number of bits/booleans set to one/true</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Get</strong></td>
<td>Gets the bit/bool at the specified index</td>
</tr>
<tr>
<td><strong>GetFirstSetIndex</strong></td>
<td>Gets the first (lowest) index set to true</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>IsEmpty</strong></td>
<td>Returns true if all bits/booleans are set to zero/false</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td><strong>RotateDown</strong></td>
<td>Shift all bits one step down, cycling the first bit to the top</td>
</tr>
<tr>
<td><strong>Set</strong></td>
<td>Sets or clears the bit/bool at the specified index</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a string that represents this object (Overrides <a href="#">Object</a>::ToString().)</td>
</tr>
</tbody>
</table>
See Also

NetBitVector Class
Lidgren.Network Namespace
NetBitVector.Clear Method

Sets all bits/booleans to zero/false

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public void Clear()
```

**Visual Basic**

```vbnet
Public Sub Clear
```

**Visual C++**

```cpp
public:
void Clear()
```
See Also

NetBitVector Class
Lidgren.Network Namespace
Returns the number of bits/booleans set to one/true

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int Count()

Visual Basic

Public Function Count As Integer

Visual C++

public:
int Count()

Return Value

See Also

NetBitVector Class
Lidgren.Network Namespace
NetBitVector Class  See Also  Send Feedback

Gets the bit/bool at the specified index

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Netowrk (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool Get(
   int bitIndex
)

Visual Basic

Public Function Get ( _
   bitIndex As Integer _
) As Boolean

Visual C++

public:
   bool Get(
      int bitIndex
   )

Parameters

bitIndex
   Type: System::Int32

[Missing <param name="bitIndex"/> documentation for 
]

Return Value

]
See Also

NetBitVector Class
Lidgren.Network Namespace
NetBitVector.GetFirstSetIndex Method

Gets the first (lowest) index set to true

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public int GetFirstSetIndex()
```

**Visual Basic**

```vbnet
Public Function GetFirstSetIndex As Integer
```

**Visual C++**

```cpp
public:
    int GetFirstSetIndex()
```

## Return Value

See Also

NetBitVector Class
Lidgren.Network Namespace
NetBitVector::IsEmpty Method

Returns true if all bits/booleans are set to zero/false

Namespace: Lidgren.Network
Syntax

C#

public bool IsEmpty()

Visual Basic

Public Function IsEmpty As Boolean

Visual C++

public:
bool IsEmpty()

Return Value

See Also

NetBitVector Class
Lidgren.Network Namespace
Shift all bits one step down, cycling the first bit to the top

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void RotateDown()

Visual Basic

Public Sub RotateDown

Visual C++

public:

void RotateDown()
See Also

NetBitVector Class
Lidgren.Network Namespace
Sets or clears the bit/bool at the specified index

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```
public void Set(
    int bitIndex,
    bool value
)
```

### Visual Basic

```
Public Sub Set (_
    bitIndex As Integer, _
    value As Boolean _
)
```

### Visual C++

```
public:
void Set(
    int bitIndex,
    bool value
)
```

## Parameters

**bitIndex**

Type: `System::::Int32`


**value**

Type: `System::::Boolean`

See Also

NetBitVector Class
Lidgren.Network Namespace
Returns a string that represents this object

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public override string ToString()
```

**Visual Basic**

```vbnet
Public Overrides Function ToString As String
```

**Visual C++**

```cpp
public:
virtual String^ ToString() override
```

**Return Value**

See Also

NetBitVector Class
Lidgren.Network Namespace
The **NetBitVector** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>Gets the bit/bool at the specified index</td>
</tr>
<tr>
<td>Capacity</td>
<td>Gets the number of bits/booleans stored in this vector</td>
</tr>
</tbody>
</table>
See Also

NetBitVector Class
Lidgren.Network Namespace
Gets the bit/bool at the specified index

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool this[int index] { get; set; }
```

Visual Basic

```vbnet
Public Default Property Bit ( _
    index As Integer _
) As Boolean
    Get
    Set
```

Visual C++

```cpp
public:
    property bool default[int index] {
        bool get (int index);
        void set (int index, bool value);
    }
```

Parameters

index
    Type: System..::.Int32
See Also

NetBitVector Class
Lidgren.Network Namespace
Gets the number of bits/booleans stored in this vector

Namespace: Lidgren.Network
Syntax

C#

public int Capacity { get; }

Visual Basic

Public ReadOnly Property Capacity As Integer
    Get

Visual C++

public:
    property int Capacity {
        int get ();
    }

See Also

NetBitVector Class
Lidgren.Network Namespace
NetBitWriter Class

Helper class for NetBuffer to write/read bits

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public static class NetBitWriter
```

**Visual Basic**

```vbnet
Public NotInheritable Class NetBitWriter
```

**Visual C++**

```cpp
public ref class NetBitWriter abstract sealed
```
Inheritance Hierarchy

System...Object
 Lidgren.Network...NetBitWriter
See Also

NetBitWriter Members
Lidgren.Network Namespace
The **NetBitWriter** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadByte</strong></td>
<td>Read 1-8 bits from a buffer into a byte</td>
</tr>
<tr>
<td><strong>ReadBytes</strong></td>
<td>Read several bytes from a buffer</td>
</tr>
<tr>
<td><strong>ReadUInt16</strong></td>
<td>Reads the specified number of bits into an UInt32</td>
</tr>
<tr>
<td><strong>ReadUInt32</strong></td>
<td>Reads a UInt32 written using WriteUnsignedVarInt(); will increment offset!</td>
</tr>
<tr>
<td><strong>WriteByte</strong></td>
<td>Write a byte consisting of 1-8 bits to a buffer; assumes buffer is previously allocated</td>
</tr>
<tr>
<td><strong>WriteBytes</strong></td>
<td>Write several whole bytes</td>
</tr>
<tr>
<td><strong>WriteUInt16</strong></td>
<td>Writes the specified number of bits into a byte array</td>
</tr>
<tr>
<td><strong>WriteUInt32</strong></td>
<td>Writes the specified number of bits into a byte array</td>
</tr>
<tr>
<td><strong>WriteUInt64</strong></td>
<td>Write Base128 encoded variable sized unsigned integer</td>
</tr>
<tr>
<td><strong>WriteVariableUInt32</strong></td>
<td>Write Base128 encoded variable sized unsigned integer</td>
</tr>
</tbody>
</table>
See Also

NetBitWriter Class
Lidgren.Network Namespace
The NetBitWriter type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadByte</td>
<td>Read 1-8 bits from a buffer into a byte</td>
</tr>
<tr>
<td>ReadBytes</td>
<td>Read several bytes from a buffer</td>
</tr>
<tr>
<td>ReadUInt16</td>
<td>Reads the specified number of bits into an UInt32</td>
</tr>
<tr>
<td>ReadUInt32</td>
<td>Reads a UInt32 written using WriteUnsignedVarInt(); will increment offset!</td>
</tr>
<tr>
<td>ReadVariableUInt32</td>
<td>Write a byte consisting of 1-8 bits to a buffer; assumes buffer is previously allocated</td>
</tr>
<tr>
<td>WriteByte</td>
<td>Write several whole bytes</td>
</tr>
<tr>
<td>WriteBytes</td>
<td>Write the specified number of bits into a byte array</td>
</tr>
<tr>
<td>WriteUInt16</td>
<td>Write the specified number of bits into a byte array</td>
</tr>
<tr>
<td>WriteUInt32</td>
<td>Write Base128 encoded variable sized unsigned integer</td>
</tr>
<tr>
<td>WriteUInt64</td>
<td></td>
</tr>
<tr>
<td>WriteVariableUInt32</td>
<td></td>
</tr>
</tbody>
</table>
See Also

NetBitWriter Class
Lidgren.Network Namespace
Read 1-8 bits from a buffer into a byte

Namespace: Lidgren.Network
Syntax

C#

```csharp
public static byte ReadByte(
    byte[] fromBuffer,
    int numberOfBits,
    int readBitOffset
)
```

Visual Basic

```vbnet
Public Shared Function ReadByte (_
    fromBuffer As Byte(), _
    numberOfBits As Integer, _
    readBitOffset As Integer _
) As Byte
```

Visual C++

```cpp
public:
static unsigned char ReadByte(
    array<unsigned char>^ fromBuffer,  
    int numberOfBits,  
    int readBitOffset
)
```

Parameters

fromBuffer
   Type: array<System..::.Byte>[][]

   [Missing <param name="fromBuffer"/> documentation for
]

numberOfBits
   Type: System..::.Int32

   [Missing <param name="numberOfBits"/> documentation for
]
readBitOffset

Type: **System..::..Int32**

[Missing <param name="readBitOffset"/> documentation for
]

**Return Value**

[Missing <returns> documentation for
]
See Also

NetBitWriter Class
Lidgren.Network Namespace
Read several bytes from a buffer

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static void ReadBytes(
    byte[] fromBuffer,
    int numberOfBytes,
    int readBitOffset,
    byte[] destination,
    int destinationByteOffset
)

Visual Basic

Public Shared Sub ReadBytes (_
    fromBuffer As Byte(), _
    numberOfBytes As Integer, _
    readBitOffset As Integer, _
    destination As Byte(), _
    destinationByteOffset As Integer _
)

Visual C++

public:
    static void ReadBytes(
        array<unsigned char>^ fromBuffer,
        int numberOfBytes,
        int readBitOffset,
        array<unsigned char>^ destination,
        int destinationByteOffset
    )

Parameters

fromBuffer
    Type: array<System..::.Byte>[]()[[]]
numberOfBytes
   Type: System..::.Int32

[Missing <param name="numberOfBytes"/> documentation for

readBitOffset
   Type: System..::.Int32

[Missing <param name="readBitOffset"/> documentation for

destination
   Type: array<System..::.Byte>[]()[[]]

[Missing <param name="destination"/> documentation for

destinationByteOffset
   Type: System..::.Int32

[Missing <param name="destinationByteOffset"/> documentation for
See Also

NetBitWriter Class
Lidgren.Network Namespace
NetBitWriter Class

See Also

Send Feedback

[Missing <summary> documentation for

Namespace: Lidgren.Network
(2012.1.7.0)
Syntax

C#

public static ushort ReadUInt16(
    byte[] fromBuffer,
    int numberOfBits,
    int readBitOffset
)

Visual Basic

Public Shared Function ReadUInt16 ( _
    fromBuffer As Byte(), _
    numberOfBits As Integer, _
    readBitOffset As Integer _
) As UShort

Visual C++

public:
static unsigned short ReadUInt16(
    array<unsigned char>^ fromBuffer,
    int numberOfBits,
    int readBitOffset
)

Parameters

fromBuffer
Type: array<System..::..Byte>[]()[[]]


numberOfBits
Type: System..::..Int32

readBitOffset
Type: System..::.Int32


Return Value

See Also

NetBitWriter Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBitWriter...ReadUInt32 Method

NetBitWriter Class

See Also

Send Feedback

Reads the specified number of bits into an UInt32

Namespace: Lidgren.Network
Syntax

C#

```csharp
public static uint ReadUInt32(
    byte[] fromBuffer,
    int numberOfBits,
    int readBitOffset
)
```

Visual Basic

```vbnet
Public Shared Function ReadUInt32 ( _
    fromBuffer As Byte(), _
    numberOfBits As Integer, _
    readBitOffset As Integer _
) AsUInteger
```

Visual C++

```cpp
public:
static unsigned int ReadUInt32(
    array<unsigned char>^ fromBuffer,
    int numberOfBits,
    int readBitOffset
)
```

Parameters

fromBuffer
Type: array<System::::Byte>[]()[[]]

[Missing <param name="fromBuffer"/> documentation for
]

numberOfBits
Type: System::::Int32

[Missing <param name="numberOfBits"/> documentation for
]
readBitOffset
Type: System.Int32


Return Value

See Also

NetBitWriter Class
Lidgren.Network Namespace
NetBitWriter.....ReadVariableUInt32 Method

NetBitWriter Class See Also Send Feedback

Reads a UInt32 written using WriteUnsignedVarInt(); will increment offset!

Namespace: Lidgren.Network
Syntax

C#

public static uint ReadVariableUInt32(
    byte[] buffer,
    ref int offset
)

Visual Basic

Public Shared Function ReadVariableUInt32 (_
    buffer As Byte(), _
    ByRef offset As Integer _
) As UInteger

Visual C++

public:
static unsigned int ReadVariableUInt32(
    array<unsigned char>* buffer,
    int% offset
)

Parameters

buffer
    Type: array<System..::..Byte>[]([])

    [Missing <param name="buffer"/> documentation for
]

offset
    Type: System..::..Int32%

    [Missing <param name="offset"/> documentation for
]

Return Value
[Missing <returns> documentation for
See Also

NetBitWriter Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBitWriter.....WriteByte Method

NetBitWriter Class See Also Send Feedback

Write a byte consisting of 1-8 bits to a buffer; assumes buffer is previously allocated

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public static void WriteByte(
    byte source,
    int numberOfBits,
    byte[] destination,
    int destBitOffset
)
```

### Visual Basic

```vbscript
Public Shared Sub WriteByte ( _
    source As Byte, _
    numberOfBits As Integer, _
    destination As Byte(), _
    destBitOffset As Integer _
)
```

### Visual C++

```cpp
public:
static void WriteByte(
    unsigned char source,
    int numberOfBits, array<unsigned char>^ destination,
    int destBitOffset
)
```

## Parameters

- **source**
  - Type: `[System..::.Byte]`

- **numberOfBits**
  - Type: `[System..::.Int32]`
destination
Type: array<System::Byte>[]

destBitOffset
Type: System::Int32
See Also

NetBitWriter Class
Lidgren.Network Namespace
Write several whole bytes

**Namespace:** [Lidgren.Network](Lidgren.Network)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
### Syntax

**C#**

```csharp
public static void WriteBytes(
    byte[] source,
    int sourceByteOffset,
    int numberOfBytes,
    byte[] destination,
    int destBitOffset
)
```

**Visual Basic**

```vbnet
Public Shared Sub WriteBytes ( _
    source As Byte(), _
    sourceByteOffset As Integer, _
    numberOfBytes As Integer, _
    destination As Byte(), _
    destBitOffset As Integer _
)
```

**Visual C++**

```cpp
public:
static void WriteBytes(
    array<unsigned char>^ source,
    int sourceByteOffset,
    int numberOfBytes,
    array<unsigned char>^ destination,
    int destBitOffset
)
```

### Parameters

**source**

Type: array&lt;`System::::Byte`>[]()[[]]

source_byte_offset
Type: System..::..Int32

[Missing <param name="source_byte_offset"/> documentation for

number_of_bytes
Type: System..::..Int32

[Missing <param name="number_of_bytes"/> documentation for

destination
Type: array<System..::..Byte>[]()[[]]

[Missing <param name="destination"/> documentation for

dest_bit_offset
Type: System..::..Int32

[Missing <param name="dest_bit_offset"/> documentation for
See Also

NetBitWriter Class
Lidgren.Network Namespace
[Missing <summary> documentation for

Namespace: Lidgren.Network
(2012.1.7.0)
Syntax

C#

```csharp
public static int WriteUInt16(
    ushort source,
    int numberOfBits,
    byte[] destination,
    int destinationBitOffset
)
```

Visual Basic

```vbnet
Public Shared Function WriteUInt16 (_
    source As UShort, _
    numberOfBits As Integer, _
    destination As Byte(), _
    destinationBitOffset As Integer _
) As Integer
```

Visual C++

```cpp
public:
static int WriteUInt16(
    unsigned short source,
    int numberOfBits,
    array<unsigned char>& destination,
    int destinationBitOffset
)
```

Parameters

**source**
Type: `System::::UInt16`


**numberOfBits**
Type: `System::::Int32`
**destination**

Type: array<`System::Byte`>[]>()

**destinationBitOffset**

Type: `System::Int32`

**Return Value**

[Missing <returns> documentation for
See Also

NetBitWriter Class
Lidgren.Network Namespace
C# Visual Basic
Visual C++

Lidgren Network Library documentation
NetBitWriter....WriteUInt32 Method

NetBitWriter Class See Also Send Feedback

Writes the specified number of bits into a byte array

**Namespace:** [Lidgren.Network](#)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static int WriteUInt32(
    uint source,
    int numberOfBits,
    byte[] destination,
    int destinationBitOffset
)

Visual Basic

Public Shared Function WriteUInt32 ( _
    source As UInteger, _
    numberOfBits As Integer, _
    destination As Byte(), _
    destinationBitOffset As Integer _
) As Integer

Visual C++

public:
static int WriteUInt32(
    unsigned int source,
    int numberOfBits,
    array<unsigned char>^ destination,
    int destinationBitOffset
)

Parameters

source
Type: System..::.UInt32


numberOfOfBits
Type: System..::.Int32
destination
Type: array< System.Byte[] >

destinationBitOffset
Type: System.Int32

Return Value
See Also

NetBitWriter Class
Lidgren.Network Namespace
C# Visual Basic Visual C++

Lidgren Network Library documentation
NetBitWriter...:..WriteUInt64 Method

NetBitWriter Class See Also Send Feedback

Writes the specified number of bits into a byte array

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static int WriteUInt64(
    ulong source,
    int numberOfBits,
    byte[] destination,
    int destinationBitOffset
)

Visual Basic

Public Shared Function WriteUInt64 ( _
    source As ULong, _
    numberOfBits As Integer, _
    destination As Byte(), _
    destinationBitOffset As Integer _
) As Integer

Visual C++

public:
static int WriteUInt64(
    unsigned long long source,
    int numberOfBits,
    array<unsigned char>^ destination,
    int destinationBitOffset
)

Parameters

source
Type: System::::UInt64


numberOfBits
Type: System::::Int32
destination
Type: array<System.Byte>[]

destinationBitOffset
Type: System.Int32

Return Value
See Also

NetBitWriter Class
Lidgren.Network Namespace
Write Base128 encoded variable sized unsigned integer

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public static int WriteVariableUInt32(
    byte[] intoBuffer,
    int offset,
    uint value
)
```

Visual Basic

```vbnet
Public Shared Function WriteVariableUInt32 ( _
    intoBuffer As Byte(), _
    offset As Integer, _
    value As UInt32 _
) As Integer
```

Visual C++

```cpp
public:
static int WriteVariableUInt32(
    array<unsigned char>^ intoBuffer,
    int offset,
    unsigned int value
)
```

Parameters

intoBuffer

Type: array<System..::.Byte>[]()[[]]


offset

Type: System..::.Int32

value

Type: System::::UInt32


Return Value

number of bytes written
See Also

NetBitWriter Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBlockEncryptionBase Class

Members  See Also  Send Feedback

Base for a non-threadsafte encryption class

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public abstract class NetBlockEncryptionBase : INetEncryption
```

**Visual Basic**

```vbnet
Public MustInherit Class NetBlockEncryptionBase
    Implements INetEncryption
```

**Visual C++**

```cpp
public ref class NetBlockEncryptionBase abstract : INetEncryption
```
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetBlockEncryptionBase
  Lidgren.Network..::..NetXtea
See Also

NetBlockEncryptionBase Members
Lidgren.Network Namespace
The **NetBlockEncryptionBase** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetBlockEncryptionBase</td>
<td>NetBlockEncryptionBase constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt an incoming message encrypted with corresponding Encrypt</td>
</tr>
<tr>
<td>DecryptBlock</td>
<td>Decrypt a block of bytes</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt an outgoing message with this algorithm; no writing can be done to the message after encryption, or message will be corrupted</td>
</tr>
<tr>
<td>EncryptBlock</td>
<td>Encrypt a block of bytes</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlockSize</td>
<td>Block size in bytes for this cipher</td>
</tr>
</tbody>
</table>
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
NetBlockEncryptionBase constructor

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetBlockEncryptionBase()

Visual Basic

Public Sub New

Visual C++

public:
NetBlockEncryptionBase()
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
The `NetBlockEncryptionBase` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt an incoming message encrypted with corresponding Encrypt</td>
</tr>
<tr>
<td>DecryptBlock</td>
<td>Decrypt a block of bytes</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt an outgoing message with this algorithm; no writing can be done to</td>
</tr>
<tr>
<td></td>
<td>the message after encryption, or message will be corrupted</td>
</tr>
<tr>
<td>EncryptBlock</td>
<td>Encrypt a block of bytes</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
NetBlockEncryptionBase class

Decrypt Method

Decrypt an incoming message encrypted with corresponding Encrypt

Namespace: Lidgren.Network
Syntax

C#

public bool Decrypt(
    NetIncomingMessage msg
)

Visual Basic

Public Function Decrypt ( _
    msg As NetIncomingMessage _
) As Boolean

Visual C++

public:
    virtual bool Decrypt(
        NetIncomingMessage^ msg
    ) sealed

Parameters

msg
    Type: Lidgren.NetworkNetMessage message to decrypt

Return Value

true if successful; false if failed

Implements

INetEncryptionNetMessage.Decrypt(NetIncomingMessage)
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBlockEncryptionBase..:.DecryptBlock Method

**NetBlockEncryptionBase Class**

See Also

Send Feedback

Decrypt a block of bytes

**Namespace:** [Lidgren.Network](https://www.lidgrennetwork.com)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
protected abstract void DecryptBlock(
    byte[] source,
    int sourceOffset,
    byte[] destination
)
```

### Visual Basic

```vbnet
Protected MustOverride Sub DecryptBlock (_
    source As Byte(), _
    sourceOffset As Integer, _
    destination As Byte() _
)
```

### Visual C++

```cpp
protected:
virtual void DecryptBlock(
    array<unsigned char>^ source,
    int sourceOffset,
    array<unsigned char>^ destination
) abstract
```

## Parameters

**source**
Type: `array<System:::Byte>[]`  


**sourceOffset**
Type: `System:::Int32`  

destination

Type: array&lt;System:::Byte>[][][]

[Missing <param name="destination"/> documentation for
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
Encrypt outgoing message with this algorithm; no writing can be done to the message after encryption, or message will be corrupted.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool Encrypt(NetOutgoingMessage msg)
```

Visual Basic

```vbnet
Public Function Encrypt(_
    msg As NetOutgoingMessage _
) As Boolean
```

Visual C++

```cpp
public:
    virtual bool Encrypt(NetOutgoingMessage^ msg)
) sealed
```

Parameters

msg

Type: Lidgren.Network.NetOutgoingMessage


Return Value


Implements

INetEncryption.Encrypt(NetOutgoingMessage)
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
Encrypt a block of bytes

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#
protected abstract void EncryptBlock(
    byte[] source,
    int sourceOffset,
    byte[] destination
)

Visual Basic
Protected MustOverride Sub EncryptBlock (_
    source As Byte(), _
    sourceOffset As Integer, _
    destination As Byte() _
)

Visual C++
protected:
    virtual void EncryptBlock(
        array<unsigned char>^ source,
        int sourceOffset,
        array<unsigned char>^ destination
    ) abstract

Parameters

source
Type: array<System..::.Byte>[]()[[]]

sourceOffset
Type: System..::.Int32

destination
Type: array<System..::.Byte>[][]

[Missing <param name="destination"/> documentation for
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
The `NetBlockEncryptionBase` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlockSize</td>
<td>Block size in bytes for this cipher</td>
</tr>
</tbody>
</table>
See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace
Block size in bytes for this cipher

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public abstract int BlockSize { get; }

Visual Basic

Public MustOverride ReadOnly Property BlockSize As Integer
  Get

Visual C++

public:
virtual property int BlockSize {
  int get () abstract;
}


See Also

NetBlockEncryptionBase Class
Lidgren.Network Namespace

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll)  
Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public class NetBuffer

Visual Basic

Public Class NetBuffer

Visual C++

public ref class NetBuffer
Inheritance Hierarchy

System::Object
Lidgren.Network::NetBuffer
   Lidgren.Network::NetIncomingMessage
   Lidgren.Network::NetOutgoingMessage
See Also

NetBuffer Members
Lidgren.Network Namespace
The `NetBuffer` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetBuffer</td>
<td>Initializes a new instance of the NetBuffer class</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>EnsureBufferSize</strong></td>
<td>Ensures the buffer can hold this number of bits</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>PeekBoolean</strong></td>
<td>Reads a 1-bit Boolean without advancing the read pointer</td>
</tr>
<tr>
<td><strong>PeekByte</strong></td>
<td>Reads a Byte without advancing the read pointer</td>
</tr>
<tr>
<td><strong>PeekByte(Int32)</strong></td>
<td>Reads the specified number of bits into a Byte without advancing the read pointer</td>
</tr>
<tr>
<td><strong>PeekBytes(Int32)</strong></td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td><strong>PeekBytes(array&lt;Byte&gt;[], Int32, Int32)</strong></td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td><strong>PeekDataBuffer</strong></td>
<td>Gets the internal data buffer</td>
</tr>
<tr>
<td><strong>PeekDouble</strong></td>
<td>Reads a 64-bit Double without</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PeekFloat</td>
<td>Reads a 32-bit Single without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt16</td>
<td>Reads an Int16 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt32()()()()</td>
<td>Reads an Int32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads the specified number of bits into an Int32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt64()()()()()</td>
<td>Reads an Int64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekSByte</td>
<td>Reads an SByte without advancing the read pointer</td>
</tr>
<tr>
<td>PeekSingle</td>
<td>Reads a 32-bit Single without advancing the read pointer</td>
</tr>
<tr>
<td>PeekString</td>
<td>Reads a string without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt16</td>
<td>Reads a UInt16 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt32()()()()()</td>
<td>Reads a UInt32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt32(Int32)</td>
<td>Reads the specified number of bits into a UInt32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt64()()()()()</td>
<td>Reads a UInt64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt64(Int32)</td>
<td>Reads the specified number of bits into an UInt64 without advancing the read pointer</td>
</tr>
<tr>
<td>ReadAllFields(Object)</td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>ReadAllFields(Object, BindingFlags)</code></td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td><code>ReadAllProperties(Object)</code></td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td><code>ReadAllProperties(Object, BindingFlags)</code></td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td><code>ReadBits</code></td>
<td>Reads the specified number of bits into a preallocated array</td>
</tr>
<tr>
<td><code>ReadBoolean</code></td>
<td>Reads a boolean value (stored as a single bit) written using Write(bool)</td>
</tr>
<tr>
<td><code>.ReadByte()</code></td>
<td>Reads a byte</td>
</tr>
<tr>
<td><code>.ReadByte(Byte%)</code></td>
<td>Reads a byte and returns true or false for success</td>
</tr>
<tr>
<td><code>.ReadByte(Int32)</code></td>
<td>Reads 1 to 8 bits into a byte</td>
</tr>
<tr>
<td><code>ReadBytes(Int32)</code></td>
<td>Reads the specified number of bytes</td>
</tr>
<tr>
<td><code>ReadBytes(Int32, array&lt;Byte&gt;[][])</code></td>
<td>Reads the specified number of bytes and returns true for success</td>
</tr>
<tr>
<td><code>ReadBytes(array&lt;Byte&gt;[][], Int32, Int32)</code></td>
<td>Reads the specified number of bytes into a preallocated array</td>
</tr>
<tr>
<td><code>ReadDouble</code></td>
<td>Reads a 64 bit floating point value written using Write(Double)</td>
</tr>
<tr>
<td><code>ReadFloat</code></td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td><code>ReadInt16</code></td>
<td>Reads a 16 bit signed integer written using Write(Int16)</td>
</tr>
<tr>
<td><code>ReadInt32()</code></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 32 bits, written using Write(Int32, Int32)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32%)</code></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td></td>
<td>Reads a 64 bit signed integer written</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ReadInt64()</td>
<td>Reads a signed integer stored in 1 to 64 bits, written using Write(Int64)</td>
</tr>
<tr>
<td>ReadInt64(Int32)</td>
<td>Reads a signed integer stored in 1 to 64 bits, written using Write(Int64, Int32)</td>
</tr>
<tr>
<td>ReadIPEndpoint</td>
<td>Reads a stored IPv4 endpoint description</td>
</tr>
<tr>
<td>ReadPadBits</td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.</td>
</tr>
<tr>
<td>ReadRangedInteger</td>
<td>Reads a 32 bit integer value written using WriteRangedInteger()</td>
</tr>
<tr>
<td>ReadRangedSingle</td>
<td>Reads a 32 bit floating point value written using WriteRangedSingle()</td>
</tr>
<tr>
<td>ReadSByte</td>
<td>Reads a signed byte</td>
</tr>
<tr>
<td>ReadSignedSingle</td>
<td>Reads a 32 bit floating point value written using WriteSignedSingle()</td>
</tr>
<tr>
<td>ReadSingle()</td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td>ReadSingle(Single%)</td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td>ReadString()</td>
<td>Reads a string written using Write(string)</td>
</tr>
<tr>
<td>ReadString(String%)</td>
<td>Reads a string written using Write(string) and returns true for success</td>
</tr>
<tr>
<td>ReadTime</td>
<td>Reads a value, in local time comparable to NetTime.Now, written using WriteTime() for the connection supplied</td>
</tr>
<tr>
<td>ReadUInt16</td>
<td>Reads a 16 bit unsigned integer written using Write(UInt16)</td>
</tr>
<tr>
<td>ReadUInt32()</td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32)</td>
</tr>
<tr>
<td>ReadUInt32(Int32)</td>
<td>Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32)</td>
</tr>
</tbody>
</table>
- **ReadUInt32(UInt32%)**  
  Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success

- **ReadUInt64()**  
  Reads a 64 bit unsigned integer written using Write(UInt64)

- **ReadUInt64(Int32)**  
  Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64, Int32)

- **ReadUnitSingle**  
  Reads a 32 bit floating point value written using WriteUnitSingle()

- **ReadVariableInt32**  
  Reads a variable sized Int32 written using WriteVariableInt32()

- **ReadVariableInt64**  
  Reads a variable sized Int64 written using WriteVariableInt64()

- **ReadVariableUInt32()**  
  Reads a variable sized UInt32 written using WriteVariableUInt32()

- **ReadVariableUInt32(UInt32%)**  
  Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success

- **ReadVariableUInt64**  
  Reads a variable sized UInt32 written using WriteVariableUInt32()

- **SkipPadBits()**  
  Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.

- **SkipPadBits(Int32)**  
  Pads data with the specified number of bits.

- **ToString**  
  Returns a **String** that represents the current **Object**. (Inherited from **Object**.)

- **Write(Boolean)**  
  Writes a boolean value using 1 bit

- **Write(Byte)**  
  Write a byte

- **Write(array<Byte>[][])**  
  Writes all bytes in an array

- **Write(Double)**  
  Writes a 64 bit floating point value

- **Write(Int16)**  
  Writes a signed 16 bit integer

- **Write(Int32)**  
  Writes a 32 bit signed integer

- **Write(Int64)**  
  Writes a 64 bit signed integer
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Write(IPEndPoint)</strong></td>
<td>Writes an endpoint description</td>
</tr>
<tr>
<td><strong>Write(SByte)</strong></td>
<td>Writes a signed byte</td>
</tr>
<tr>
<td><strong>Write(Single)</strong></td>
<td>Writes a 32 bit floating point value</td>
</tr>
<tr>
<td><strong>Write(String)</strong></td>
<td>Write a string</td>
</tr>
<tr>
<td><strong>Write(UInt16)</strong></td>
<td>Writes an unsigned 16 bit integer</td>
</tr>
<tr>
<td><strong>Write(UInt32)</strong></td>
<td>Writes a 32 bit unsigned integer</td>
</tr>
<tr>
<td><strong>Write(UInt64)</strong></td>
<td>Writes a 64 bit unsigned integer</td>
</tr>
<tr>
<td><strong>Write(NetIncomingMessage)</strong></td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td><strong>Write(NetOutgoingMessage)</strong></td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td><strong>Write(Byte, Int32)</strong></td>
<td>Writes 1 to 8 bits of a byte</td>
</tr>
<tr>
<td><strong>Write(Int32, Int32)</strong></td>
<td>Writes a signed integer using 1 to 32 bits</td>
</tr>
<tr>
<td><strong>Write(Int64, Int32)</strong></td>
<td>Writes a signed integer using 1 to 64 bits</td>
</tr>
<tr>
<td><strong>Write(UInt16, Int32)</strong></td>
<td>Writes an unsigned integer using 1 to 16 bits</td>
</tr>
<tr>
<td><strong>Write(UInt32, Int32)</strong></td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td><strong>Write(UInt64, Int32)</strong></td>
<td>Writes an unsigned integer using 1 to 64 bits</td>
</tr>
<tr>
<td><strong>Write(array&lt;Byte&gt;[][], Int32, Int32)</strong></td>
<td>Writes the specified number of bytes from an array</td>
</tr>
<tr>
<td><strong>WriteAllFields(Object)</strong></td>
<td>Writes all public and private declared instance fields of the object in</td>
</tr>
<tr>
<td></td>
<td>alphabetical order using reflection</td>
</tr>
<tr>
<td><strong>WriteAllFields(Object, BindingFlags)</strong></td>
<td>Writes all fields with specified binding in alphabetical order using</td>
</tr>
<tr>
<td></td>
<td>reflection</td>
</tr>
<tr>
<td><strong>WriteAllProperties(Object)</strong></td>
<td>Writes all public and private declared instance properties of the object in</td>
</tr>
<tr>
<td></td>
<td>alphabetical order using reflection</td>
</tr>
<tr>
<td><strong>WriteAllProperties(Object, BindingFlags)</strong></td>
<td>Writes all properties with specified binding in alphabetical order using</td>
</tr>
<tr>
<td></td>
<td>reflection</td>
</tr>
<tr>
<td><strong>WriteAllProperties(Object, BindingFlags)</strong></td>
<td>Pads data with enough bits to reach a</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WritePadBits()()()</td>
<td>Full byte. Decreases cpu usage for subsequent byte writes.</td>
</tr>
<tr>
<td>WritePadBits(Int32)</td>
<td>Pads data with the specified number of bits.</td>
</tr>
<tr>
<td>WriteRangedInteger</td>
<td>Writes an integer with the least amount of bits need for the specified range</td>
</tr>
<tr>
<td>WriteRangedSingle</td>
<td>Compress a float within a specified range using a certain number of bits</td>
</tr>
<tr>
<td>WriteSignedSingle</td>
<td>Compress (lossy) a float in the range -1..1 using numberOfBits bits</td>
</tr>
<tr>
<td>WriteTime(Boolean)</td>
<td>Writes the current local time to a message; readable (and convertible to</td>
</tr>
<tr>
<td></td>
<td>local time) by the remote host using ReadTime()</td>
</tr>
<tr>
<td>WriteTime(Double, Boolean)</td>
<td>Writes a local timestamp to a message; readable (and convertible to local</td>
</tr>
<tr>
<td></td>
<td>time) by the remote host using ReadTime()</td>
</tr>
<tr>
<td>WriteUnitSingle</td>
<td>Compress (lossy) a float in the range 0..1 using numberOfBits bits</td>
</tr>
<tr>
<td>WriteVariableInt32</td>
<td>Write Base128 encoded variable sized signed integer of up to 32 bits</td>
</tr>
<tr>
<td>WriteVariableInt64</td>
<td>Write Base128 encoded variable sized signed integer of up to 64 bits</td>
</tr>
<tr>
<td>WriteVariableUInt32</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 32 bits</td>
</tr>
<tr>
<td>WriteVariableUInt64</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 64 bits</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><code>c_overAllocateAmount</code></td>
<td></td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Gets or sets the internal data buffer</td>
</tr>
<tr>
<td>LengthBits</td>
<td>Gets or sets the length of the used portion of the buffer in bits</td>
</tr>
<tr>
<td>LengthBytes</td>
<td>Gets or sets the length of the used portion of the buffer in bytes</td>
</tr>
<tr>
<td>Position</td>
<td>Gets or sets the read position in the buffer, in bits (not bytes)</td>
</tr>
<tr>
<td>PositionInBytes</td>
<td>Gets the position in the buffer in bytes; note that the bits of the first returned byte may already have been read - check the Position property to make sure.</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
Lidgren.Network Namespace
NetBuffer Constructor

Initializes a new instance of the NetBuffer class

Namespace: Lidgren.Network
Syntax

C#

public NetBuffer()

Visual Basic

Public Sub New

Visual C++

public:
NetBuffer()
See Also

NetBuffer Class
Lidgren.Network Namespace
The **NetBuffer** type exposes the following members.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>c_overAllocateAmount</td>
<td></td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
Lidgren.Network Namespace
C#  Visual Basic  Visual C++  
Lidgren Network Library documentation  
NetBuffer::c_overAllocateAmount Field  
NetBuffer Class  See Also  Send Feedback


**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

protected const int c_overAllocateAmount

Visual Basic

Protected Const c_overAllocateAmount As Integer

Visual C++

protected:
literal int c_overAllocateAmount
See Also

NetBuffer Class
Lidgren.Network Namespace
The **NetBuffer** type exposes the following members.
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EnsureBufferSize</strong></td>
<td>Ensures the buffer can hold this number of bits</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>** PeekBoolean**</td>
<td>Reads a 1-bit Boolean without advancing the read pointer</td>
</tr>
<tr>
<td>** PeekByte()**</td>
<td>Reads a Byte without advancing the read pointer</td>
</tr>
<tr>
<td>** PeekByte(Int32)**</td>
<td>Reads the specified number of bits into a Byte without advancing the read pointer</td>
</tr>
<tr>
<td>** PeekBytes(Int32)**</td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td>** PeekBytes(array&lt;Byte&gt;[][], Int32, Int32)**</td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td>** PeekDataBuffer**</td>
<td>Gets the internal data buffer</td>
</tr>
<tr>
<td>** PeekDouble**</td>
<td>Reads a 64-bit Double without</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>PeekFloat</td>
<td>Reads a 32-bit Single without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt16</td>
<td>Reads an Int16 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt32()</td>
<td>Reads an Int32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads the specified number of bits into an Int32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt64()</td>
<td>Reads an Int64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekSByte</td>
<td>Reads an SByte without advancing the read pointer</td>
</tr>
<tr>
<td>PeekSingle</td>
<td>Reads a 32-bit Single without advancing the read pointer</td>
</tr>
<tr>
<td>PeekString</td>
<td>Reads a string without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt16</td>
<td>Reads a UInt16 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt32()</td>
<td>Reads a UInt32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt32(Int32)</td>
<td>Reads the specified number of bits into a UInt32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt64()</td>
<td>Reads a UInt64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt64(Int32)</td>
<td>Reads the specified number of bits into a UInt64 without advancing the read pointer</td>
</tr>
<tr>
<td>ReadAllFields(Object)</td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>ReadAllFields(Object, BindingFlags)</code></td>
<td>Reads all fields with the specified binding of the object in alphabetical</td>
</tr>
<tr>
<td></td>
<td>order using reflection</td>
</tr>
<tr>
<td><code>ReadAllProperties(Object)</code></td>
<td>Reads all public and private declared instance fields of the object in</td>
</tr>
<tr>
<td></td>
<td>alphabetical order using reflection</td>
</tr>
<tr>
<td><code>ReadAllProperties(Object, BindingFlags)</code></td>
<td>Reads all fields with the specified binding of the object in alphabetical</td>
</tr>
<tr>
<td></td>
<td>order using reflection</td>
</tr>
<tr>
<td><code>ReadBits</code></td>
<td>Reads the specified number of bits into a preallocated array</td>
</tr>
<tr>
<td><code>ReadBoolean</code></td>
<td>Reads a boolean value (stored as a single bit) written using Write(bool)</td>
</tr>
<tr>
<td><code>.ReadByte()</code></td>
<td>Reads a byte</td>
</tr>
<tr>
<td><code>.ReadByte(Byte%)</code></td>
<td>Reads a byte and returns true or false for success</td>
</tr>
<tr>
<td><code>.ReadByte(Int32)</code></td>
<td>Reads 1 to 8 bits into a byte</td>
</tr>
<tr>
<td><code>ReadBytes(Int32)</code></td>
<td>Reads the specified number of bytes</td>
</tr>
<tr>
<td><code>ReadBytes(Int32, array&lt;Byte&gt;[][])</code></td>
<td>Reads the specified number of bytes and returns true for success</td>
</tr>
<tr>
<td><code>ReadBytes(array&lt;Byte&gt;[][], Int32, Int32)</code></td>
<td>Reads the specified number of bytes into a preallocated array</td>
</tr>
<tr>
<td><code>ReadDouble</code></td>
<td>Reads a 64 bit floating point value written using Write(Double)</td>
</tr>
<tr>
<td><code>ReadFloat</code></td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td><code>ReadInt16</code></td>
<td>Reads a 16 bit signed integer written using Write(Int16)</td>
</tr>
<tr>
<td><code>ReadInt32()()()()</code></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 32 bits, written using Write(Int32,</td>
</tr>
<tr>
<td></td>
<td>Int32)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32%)</code></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td></td>
<td>Reads a 64 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ReadInt64()</td>
<td>Reads a signed integer stored in 1 to 64 bits, written using Write(Int64)</td>
</tr>
<tr>
<td>ReadInt64(Int32)</td>
<td>Reads a signed integer stored in 1 to 64 bits, written using Write(Int64, Int32)</td>
</tr>
<tr>
<td>ReadIPEndpoint</td>
<td>Reads a stored IPv4 endpoint description</td>
</tr>
<tr>
<td>ReadPadBits</td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.</td>
</tr>
<tr>
<td>ReadRangedInteger</td>
<td>Reads a 32 bit integer value written using WriteRangedInteger()</td>
</tr>
<tr>
<td>ReadRangedSingle</td>
<td>Reads a 32 bit floating point value written using WriteRangedSingle()</td>
</tr>
<tr>
<td>ReadSByte</td>
<td>Reads a signed byte</td>
</tr>
<tr>
<td>ReadSignedSingle</td>
<td>Reads a 32 bit floating point value written using WriteSignedSingle()</td>
</tr>
<tr>
<td>ReadSingle()</td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td>ReadSingle(Single%)</td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td>ReadString()</td>
<td>Reads a string written using Write(string)</td>
</tr>
<tr>
<td>ReadString(String%)</td>
<td>Reads a string written using Write(string) and returns true for success</td>
</tr>
<tr>
<td>ReadTime</td>
<td>Reads a value, in local time comparable to NetTime.Now, written using WriteTime() for the connection supplied</td>
</tr>
<tr>
<td>ReadUInt16</td>
<td>Reads a 16 bit unsigned integer written using Write(UInt16)</td>
</tr>
<tr>
<td>ReadUInt32()</td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32)</td>
</tr>
<tr>
<td>ReadUInt32(Int32)</td>
<td>Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>ReadUInt32(UInt32%)</td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success</td>
</tr>
<tr>
<td>ReadUInt64()()()()</td>
<td>Reads a 64 bit unsigned integer written using Write(UInt64)</td>
</tr>
<tr>
<td>ReadUInt64(Int32)</td>
<td>Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64, Int32)</td>
</tr>
<tr>
<td>ReadUnitSingle</td>
<td>Reads a 32 bit floating point value written using WriteUnitSingle()</td>
</tr>
<tr>
<td>ReadVariableInt32</td>
<td>Reads a variable sized Int32 written using WriteVariableInt32()</td>
</tr>
<tr>
<td>ReadVariableInt64</td>
<td>Reads a variable sized Int64 written using WriteVariableInt64()</td>
</tr>
<tr>
<td>ReadVariableUInt32()()()()</td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32()</td>
</tr>
<tr>
<td>ReadVariableUInt32(UInt32%)</td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success</td>
</tr>
<tr>
<td>ReadVariableUInt64</td>
<td>Reads a variable sized UInt64 written using WriteVariableUInt64()</td>
</tr>
<tr>
<td>SkipPadBits()()()()</td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.</td>
</tr>
<tr>
<td>SkipPadBits(Int32)</td>
<td>Pads data with the specified number of bits.</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Write(Boolean)</td>
<td>Writes a boolean value using 1 bit</td>
</tr>
<tr>
<td>Write(Byte)</td>
<td>Write a byte</td>
</tr>
<tr>
<td>Write(array&lt;Byte&gt;[][])</td>
<td>Writes all bytes in an array</td>
</tr>
<tr>
<td>Write(Double)</td>
<td>Writes a 64 bit floating point value</td>
</tr>
<tr>
<td>Write(Int16)</td>
<td>Writes a signed 16 bit integer</td>
</tr>
<tr>
<td>Write(Int32)</td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td>Write(Int64)</td>
<td>Writes a 64 bit signed integer</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>Write(IPEndPoint)</td>
<td>Writes an endpoint description</td>
</tr>
<tr>
<td>Write(SByte)</td>
<td>Writes a signed byte</td>
</tr>
<tr>
<td>Write(Single)</td>
<td>Writes a 32 bit floating point value</td>
</tr>
<tr>
<td>Write(String)</td>
<td>Write a string</td>
</tr>
<tr>
<td>Write(UInt16)</td>
<td>Writes an unsigned 16 bit integer</td>
</tr>
<tr>
<td>Write(UInt32)</td>
<td>Writes a 32 bit unsigned integer</td>
</tr>
<tr>
<td>Write(UInt64)</td>
<td>Writes a 64 bit unsigned integer</td>
</tr>
<tr>
<td>Write(NetIncomingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(NetOutgoingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(Byte, Int32)</td>
<td>Writes 1 to 8 bits of a byte</td>
</tr>
<tr>
<td>Write(Int32, Int32)</td>
<td>Writes a signed integer using 1 to 32 bits</td>
</tr>
<tr>
<td>Write(Int64, Int32)</td>
<td>Writes a signed integer using 1 to 64 bits</td>
</tr>
<tr>
<td>Write(UInt16, Int32)</td>
<td>Writes an unsigned integer using 1 to 16 bits</td>
</tr>
<tr>
<td>Write(UInt32, Int32)</td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td>Write(UInt64, Int32)</td>
<td>Writes an unsigned integer using 1 to 64 bits</td>
</tr>
<tr>
<td>Write(array&lt;Byte&gt;[][], Int32, Int32)</td>
<td>Writes the specified number of bytes from an array</td>
</tr>
<tr>
<td>WriteAllFields(Object)</td>
<td>Writes all public and private declared instance fields of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td>WriteAllFields(Object, BindingFlags)</td>
<td>Writes all fields with specified binding in alphabetical order using reflection</td>
</tr>
<tr>
<td>WriteAllProperties(Object)</td>
<td>Writes all public and private declared instance properties of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td>WriteAllProperties(Object, BindingFlags)</td>
<td>Writes all properties with specified binding in alphabetical order using reflection</td>
</tr>
<tr>
<td></td>
<td>Pads data with enough bits to reach a</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WritePadBits()</td>
<td>Pads data with the specified number of bits.</td>
</tr>
<tr>
<td>WritePadBits(Int32)</td>
<td>Writes an integer with the least amount of bits need for the specified range.</td>
</tr>
<tr>
<td>WriteRangedInteger</td>
<td>Returns number of bits written.</td>
</tr>
<tr>
<td>WriteRangedSingle</td>
<td>Compress a float within a specified range using a certain number of bits.</td>
</tr>
<tr>
<td>WriteSignedSingle</td>
<td>Compress (lossy) a float in the range -1..1 using numberOfBits bits.</td>
</tr>
<tr>
<td>WriteTime(Boolean)</td>
<td>Writes the current local time to a message; readable (and convertible to local time) by the remote host using ReadTime()</td>
</tr>
<tr>
<td>WriteTime(Double, Boolean)</td>
<td>Writes a local timestamp to a message; readable (and convertible to local time) by the remote host using ReadTime()</td>
</tr>
<tr>
<td>WriteUnitSingle</td>
<td>Compress (lossy) a float in the range 0..1 using numberOfBits bits.</td>
</tr>
<tr>
<td>WriteVariableInt32</td>
<td>Write Base128 encoded variable sized signed integer of up to 32 bits.</td>
</tr>
<tr>
<td>WriteVariableInt64</td>
<td>Write Base128 encoded variable sized signed integer of up to 64 bits.</td>
</tr>
<tr>
<td>WriteVariableUInt32</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 32 bits.</td>
</tr>
<tr>
<td>WriteVariableUInt64</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 64 bits.</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
Lidgren.Network Namespace
Ensures the buffer can hold this number of bits

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void EnsureBufferSize(
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Sub EnsureBufferSize ( _
    numberOfBits As Integer _
)
```

Visual C++

```cpp
public:
void EnsureBufferSize(  
    int numberOfBits
)
```

Parameters

numberOfBits
Type: `System::::Int32`

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer.::.PeekBoolean Method

NetBuffer Class See Also Send Feedback

Reads a 1-bit Boolean without advancing the read pointer

Namespace: Lidgren.Network
Syntax

C#

public bool PeekBoolean()

Visual Basic

Public Function PeekBoolean As Boolean

Visual C++

public:
bool PeekBoolean()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetBuffer...: PeekByte Method

NetBuffer Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekByte()()()</td>
<td>Reads a Byte without advancing the read pointer</td>
</tr>
<tr>
<td>PeekByte(Int32)</td>
<td>Reads the specified number of bits into a Byte without advancing the read pointer</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer:::PeekByte Method

NetBuffer Class See Also Send Feedback

Reads a Byte without advancing the read pointer

Namespace: Lidgren.Network
### Syntax

**C#**

```csharp
public byte PeekByte()
```

**Visual Basic**

```vbnet
Public Function PeekByte As Byte
```

**Visual C++**

```cpp
public:
unsigned char PeekByte()
```

### Return Value

See Also

NetBuffer Class
PeekByte Overload
Lidgren.Network Namespace
Reads the specified number of bits into a Byte without advancing the read pointer

Namespace: Lidgren.Network
Syntax

C#

```csharp
public byte PeekByte(
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Function PeekByte ( _
    numberOfBits As Integer _
) As Byte
```

Visual C++

```cpp
public:
    unsigned char PeekByte(
        int numberOfBits
    )
```

Parameters

numberOfBits
Type: System::Int32

[Missing <param name="numberOfBits"/> documentation for
]

Return Value

See Also

NetBuffer Class
PeekByte Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetBuffer....PeekBytes Method

NetBuffer Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekBytes(Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td>PeekBytes(array[Byte][][][], Int32, Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads the specified number of bytes without advancing the read pointer

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public byte[] PeekBytes(
    int numberOfBytes
)
```

**Visual Basic**

```vbnet
Public Function PeekBytes (
    numberOfBytes As Integer
) As Byte()
```

**Visual C++**

```cpp
public:
array<unsigned char>^ PeekBytes(
    int numberOfBytes
)
```

**Parameters**

`numberOfBytes`  
Type: `System::Int32`  

**Return Value**

See Also

NetBuffer Class
PeekBytes Overload
Lidgren.Network Namespace
Reads the specified number of bytes without advancing the read pointer

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
### Syntax

**C#**

```csharp
public void PeekBytes(
    byte[] into,
    int offset,
    int numberOfBytes
)
```

**Visual Basic**

```vbnet
Public Sub PeekBytes (_
    into As Byte(), _
    offset As Integer, _
    numberOfBytes As Integer _
)
```

**Visual C++**

```cpp
public:
void PeekBytes(
    array<unsigned char>^ into,
    int offset,
    int numberOfBytes
)
```

### Parameters

**into**

Type: array<System::Byte>[][]


**offset**

Type: System.Int32

numberOfBytes
Type: System..::.Int32

[Missing <param name="numberOfBytes"/> documentation for
See Also

NetBuffer Class
PeekBytes Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer.PeekDataBuffer Method

NetBuffer Class See Also Send Feedback

Gets the internal data buffer

Namespace: Lidgren.Network
Syntax

C#

public byte[] PeekDataBuffer();

Visual Basic

Public Function PeekDataBuffer As Byte()

Visual C++

public:
array<unsigned char>^ PeekDataBuffer();

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer::<..PeekDouble Method

NetBuffer Class See Also Send Feedback

Reads a 64-bit Double without advancing the read pointer

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public double PeekDouble()
```

**Visual Basic**

```vbnet
Public Function PeekDouble As Double
```

**Visual C++**

```cpp
public:
    double PeekDouble()
```

**Return Value**

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer:::.:PeekFloat Method

NetBuffer Class  See Also  Send Feedback

Reads a 32-bit Single without advancing the read pointer

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public float PeekFloat()
```

**Visual Basic**

```vbnet
Public Function PeekFloat As Single
```

**Visual C++**

```cpp
public:
float PeekFloat()
```

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Reads an Int16 without advancing the read pointer

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public short PeekInt16()
```

Visual Basic

```vbnet
Public Function PeekInt16 As Short
```

Visual C++

```cpp
public:
short PeekInt16()
```

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekInt32()</td>
<td>Reads an Int32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads the specified number of bits into an Int32 without advancing the read pointer</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer...:..:PeekInt32 Method

NetBuffer Class  See Also  Send Feedback

Reads an Int32 without advancing the read pointer

Namespace: Lidgren.Network
Syntax

C#

public int PeekInt32()

Visual Basic

Public Function PeekInt32 As Integer

Visual C++

public:
  int PeekInt32()

Return Value

See Also

NetBuffer Class
PeekInt32 Overload
Lidgren.Network Namespace
Reads the specified number of bits into an Int32 without advancing the read pointer

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public int PeekInt32(
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Function PeekInt32 ( _
    numberOfBits As Integer _
) As Integer
```

Visual C++

```cpp
public:
    int PeekInt32(
        int numberOfBits
    )
```

Parameters

`numberOfBits`  
Type: System::Int32  


Return Value

See Also

NetBuffer Class
PeekInt32 Overload
Lidgren.Network Namespace
NetBuffer Class
See Also
Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekInt64()()</td>
<td>Reads an Int64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads an Int64 without advancing the read pointer

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public long PeekInt64()
```

**Visual Basic**

```vbnet
Public Function PeekInt64 As Long
```

**Visual C++**

```cpp
public:
    long long PeekInt64()
```

## Return Value

See Also

NetBuffer Class
PeekInt64 Overload
Lidgren.Network Namespace
 Reads the specified number of bits into an Int64 without advancing the read pointer

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public long PeekInt64(
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Function PeekInt64 ( _
    numberOfBits As Integer _
) As Long
```

Visual C++

```cpp
public:
    long long PeekInt64(
        int numberOfBits
    )
```

Parameters

numberOfBits
Type: System::Int32


Return Value

See Also

NetBuffer Class
PeekInt64 Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Lidgren Network Library documentation
NetBuffer..::..PeekSByte Method
NetBuffer Class See Also Send Feedback

Reads an SByte without advancing the read pointer

Namespace: Lidgren.Network
Syntax

C#

public sbyte PeekSByte()

Visual Basic

Public Function PeekSByte As SByte

Visual C++

public:
    signed char PeekSByte()

Return Value

See Also

- NetBuffer Class
- Lidgren.Network Namespace
NetBuffer.PeekSingle Method

Reads a 32-bit Single without advancing the read pointer

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public float PeekSingle()
```

**Visual Basic**

```vbnet
Public Function PeekSingle As Single
```

**Visual C++**

```c++
public:
float PeekSingle()
```

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer..::..PeekString Method

NetBuffer Class  See Also  Send Feedback

Reads a string without advancing the read pointer

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public string PeekString()
```

**Visual Basic**

```vbnet
Public Function PeekString As String
```

**Visual C++**

```cpp
public: String^ PeekString()
```

**Return Value**

See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a UInt16 without advancing the read pointer

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public ushort PeekUInt16()

Visual Basic

Public Function PeekUInt16 As UShort

Visual C++

public:
unsigned short PeekUInt16()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer::PeekUInt32 Method

NetBuffer Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekUInt32</td>
<td>Reads a UInt32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt32(Int32)</td>
<td>Reads the specified number of bits into a UInt32 without advancing the read pointer</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer::PeekUInt32 Method

Reads a UInt32 without advancing the read pointer

Namespace: Lidgren.Network
Syntax

C#
public uint PeekUInt32()

Visual Basic
Public Function PeekUInt32 As UInteger

Visual C++

public:
unsigned int PeekUInt32()

Return Value

[Missing <returns> documentation for "M:Lidgren_NETWORK.NETBuffer.PeekUInt32"]
See Also

NetBuffer Class
PeekUInt32 Overload
Lidgren.Network Namespace
Reads the specified number of bits into a UInt32 without advancing the read pointer

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public uint PeekUInt32(
    int numberOfBits
)
```

**Visual Basic**

```vbnet
Public Function PeekUInt32 ( _
    numberOfBits As Integer _
) As UInteger
```

**Visual C++**

```cpp
public:
    unsigned int PeekUInt32(
        int numberOfBits
    )
```

Parameters

**numberOfBits**

Type: System::Int32


Return Value

See Also

NetBuffer Class
peekUInt32 Overload
Lidgren.Network Namespace
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekUInt64()</td>
<td>Reads a UInt64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekUInt64(Int32)</td>
<td>Reads the specified number of bits into an UInt64 without advancing the read pointer</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
NetBuffer PeekUInt64 Method

**NetBuffer Class** See Also Send Feedback

Reads a UInt64 without advancing the read pointer

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public ulong PeekUInt64()
```

### Visual Basic

```visualbasic
Public Function PeekUInt64 As ULong
```

### Visual C++

```cpp
public:
unsigned long long PeekUInt64()
```

## Return Value

See Also

NetBuffer Class
 PeekUInt64 Overload
 Lidgren.Network Namespace
Reads the specified number of bits into an UInt64 without advancing the read pointer

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public ulong PeekUInt64(
    int numberOfBits
)
```

**Visual Basic**

```vbnet
Public Function PeekUInt64 ( _
    numberOfBits As Integer _
) As ULong
```

**Visual C++**

```cpp
public:
    unsigned long long PeekUInt64(
        int numberOfBits
    )
```

### Parameters

**numberOfBits**

Type: `System::Int32`


### Return Value


See Also

NetBuffer Class
PeekUInt64 Overload
Lidgren.Network Namespace
C# Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetBuffer...ReadAllFields Method
NetBuffer Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadAllFields(Object)</td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td>ReadAllFields(Object, BindingFlags)</td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads all public and private declared instance fields of the object in alphabetical order using reflection

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
(2012.1.7.0)
Syntax

C#

```csharp
public void ReadAllFields(
    Object target
)
```

Visual Basic

```vbnet
Public Sub ReadAllFields ( _
    target As Object _
)
```

Visual C++

```cpp
public:
void ReadAllFields(
    Object^ target
)
```

Parameters

target
  Type: System::Object

See Also

NetBuffer Class
ReadAllFields Overload
Lidgren.Network Namespace
Reads all fields with the specified binding of the object in alphabetical order using reflection

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public void ReadAllFields(
    Object target,
    BindingFlags flags
)
```

### Visual Basic

```vbnet
Public Sub ReadAllFields (_, _
    target As Object, _
    flags As BindingFlags _
)
```

### Visual C++

```cpp
public:
void ReadAllFields(
    Object^ target,
    BindingFlags flags
)
```

## Parameters

**target**

Type: `System::Object`


**flags**

Type: `System.Reflection::BindingFlags`

See Also

NetBuffer Class
ReadAllFields Overload
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetBuffer:::ReadAllProperties Method
NetBuffer Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadAllProperties(Object)</strong></td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td><strong>ReadAllProperties(Object, BindingFlags)</strong></td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer...:::ReadStreamProperties Method (Object)

Reads all public and private declared instance fields of the object in alphabetical order using reflection

Namespace: Lidgren.Network
## Syntax

**C#**

```csharp
public void ReadAllProperties(
    Object target
)
```

**Visual Basic**

```vbscript
Public Sub ReadAllProperties (_
    target As Object _
)
```

**Visual C++**

```cpp
public:
void ReadAllProperties(
    Object^ target
)
```

### Parameters

target

Type: `System::Object`

See Also

NetBuffer Class
ReadAllProperties Overload
Lidgren.Network Namespace
Reads all fields with the specified binding of the object in alphabetical order using reflection

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public void ReadAllProperties(
    Object target,
    BindingFlags flags
)
```

### Visual Basic

```vbnet
Public Sub ReadAllProperties (_
    target As Object,
    flags As BindingFlags _
)
```

### Visual C++

```cpp
public:
void ReadAllProperties(
    Object^ target,
    BindingFlags flags
)
```

## Parameters

**target**

Type: `System::Object`

[Missing <param name="target"/> documentation for

**flags**

Type: `System.Reflection::BindingFlags`

[Missing <param name="flags"/> documentation for
See Also

NetBuffer Class
ReadAllProperties Overload
Lidgren.Network Namespace
Reads the specified number of bits into a preallocated array

Namespace: Lidgren.Network
Syntax

C#

public void ReadBits(
    byte[] into,
    int offset,
    int numberOfBits
)

Visual Basic

Public Sub ReadBits (
    into As Byte(),
    offset As Integer, _
    numberOfBits As Integer _
)

Visual C++

public:
    void ReadBits(
        array<unsigned char>^ into,
        int offset,
        int numberOfBits
    )

Parameters

into
    Type: array<System::::Byte>[][]
    The destination array

offset
    Type: System::::Int32
    The offset where to start writing in the destination array

numberOfBits
    Type: System::::Int32
The number of bits to read
See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a boolean value (stored as a single bit) written using Write(bool)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public bool ReadBoolean()
```

**Visual Basic**

```vbnet
Public Function ReadBoolean As Boolean
```

**Visual C++**

```cpp
public:
bool ReadBoolean()
```

**Return Value**

See Also

NetBuffer Class
Lidgren.Network Namespace
NetBuffer Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadByte()</code></td>
<td>Reads a byte</td>
</tr>
<tr>
<td><code>ReadByte(Byte%)</code></td>
<td>Reads a byte and returns true or false for success</td>
</tr>
<tr>
<td><code>ReadByte(Int32)</code></td>
<td>Reads 1 to 8 bits into a byte</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads a byte

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

public byte ReadByte()

**Visual Basic**

Public Function ReadByte As Byte

**Visual C++**

public:

unsigned char ReadByte()

Return Value

See Also

NetBuffer Class
ReadByte Overload
Lidgren.Network Namespace
Reads a byte and returns true or false for success

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public bool ReadByte(
    out byte result
)
```

### Visual Basic

```vbnet
Public Function ReadByte (_
    <OutAttribute> ByRef result As Byte _
) As Boolean
```

### Visual C++

```cpp
public:
bool ReadByte(
    [OutAttribute] unsigned char% result
)
```

### Parameters

result

Type: `System::Byte`


### Return Value

See Also

NetBuffer Class
ReadByte Overload
Lidgren.Network Namespace
Reads 1 to 8 bits into a byte

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public byte ReadByte(
    int numberOfBits
)

Visual Basic

Public Function ReadByte ( _
    numberOfBits As Integer _
) As Byte

Visual C++

public:
    unsigned char ReadByte(
    int numberOfBits
)

Parameters

numberOfBits
Type: System::::Int32


Return Value

See Also

NetBuffer Class
ReadStream Overload
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadBytes(Int32)</code></td>
<td>Reads the specified number of bytes</td>
</tr>
<tr>
<td><code>ReadBytes(Int32, array&lt;Byte&gt;[][])</code></td>
<td>Reads the specified number of bytes and returns true for success</td>
</tr>
<tr>
<td><code>ReadBytes(array&lt;Byte&gt;[][], Int32, Int32)</code></td>
<td>Reads the specified number of bytes into a preallocated array</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads the specified number of bytes

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

### C#

```csharp
public byte[] ReadBytes(
    int numberOfBytes
)
```

### Visual Basic

```vbnet
Public Function ReadBytes ( _
    numberOfBytes As Integer _
) As Byte()
```

### Visual C++

```cpp
public:
    array<unsigned char>^ ReadBytes(
        int numberOfBytes
    )
```

Parameters

**numberOfBytes**

Type: System::Int32

Return Value

See Also

NetBuffer Class
ReadBytes Overload
Lidgren.Network Namespace
Reads the specified number of bytes and returns true for success

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool ReadBytes(
    int numberOfBytes,
    out byte[] result
)
```

Visual Basic

```vbnet
Public Function ReadBytes (_
    numberOfBytes As Integer, _
    <OutAttribute> ByRef result As Byte() _
) As Boolean
```

Visual C++

```cpp
public: 
    bool ReadBytes(
        int numberOfBytes,
        [OutAttribute] array<unsigned char>^% result
    )
```

Parameters

numberOfBytes
Type: System::...::Int32

Result
Type: array<System::...::Byte>[]()[[]%

Return Value
[Missing <returns> documentation for
See Also

NetBuffer Class
ReadStream Overload
Lidgren.Network Namespace
Reads the specified number of bytes into a preallocated array

Namespace: Lidgren.Network
Syntax

C#

public void ReadBytes(
    byte[] into,
    int offset,
    int numberOfBytes
)

Visual Basic

Public Sub ReadBytes (_
    into As Byte(), _
    offset As Integer, _
    numberOfBytes As Integer _
)

Visual C++

public:
void ReadBytes(
    array<unsigned char>^ into,
    int offset,
    int numberOfBytes
)

Parameters

into
    Type: array<System::::Byte>[][]
The destination array

offset
    Type: System::::Int32
    The offset where to start writing in the destination array

numberOfBytes
    Type: System::::Int32
The number of bytes to read
See Also

NetBuffer Class
ReadBytes Overload
Lidgren.Network Namespace
Reads a 64 bit floating point value written using Write(Double)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public double ReadDouble()

Visual Basic

Public Function ReadDouble As Double

Visual C++

public:
double ReadDouble()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a 32 bit floating point value written using Write(Single)

Namespace: Lidgren.Network
Syntax

C#

public float ReadFloat()

Visual Basic

Public Function ReadFloat As Single

Visual C++

public:
float ReadFloat()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a 16 bit signed integer written using Write(Int16)

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public short ReadInt16()

Visual Basic

Public Function ReadInt16 As Short

Visual C++

public:
short ReadInt16()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetBuffer:::ReadInt32 Method

NetBuffer Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadInt32(0)</code></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 32 bits, written using Write(Int32, Int32)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32%)</code></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads a 32 bit signed integer written using Write(Int32)

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int ReadInt32()

Visual Basic

Public Function ReadInt32 As Integer

Visual C++

public:
int ReadInt32()

Return Value

See Also

NetBuffer Class
ReadInt32 Overload
Lidgren.Network Namespace
Reads a signed integer stored in 1 to 32 bits, written using Write(Int32, Int32)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public int ReadInt32(
    int numberOfBits
)
```

**Visual Basic**

```vbnet
Public Function ReadInt32 ( _
    numberOfBits As Integer _
) As Integer
```

**Visual C++**

```cpp
public:
    int ReadInt32(
        int numberOfBits
    )
```

## Parameters

`numberOfBits`  
Type: `[System::Int32](https://docs.microsoft.com/en-us/dotnet/api/system.int32)


## Return Value

See Also

NetBuffer Class
ReadInt32 Overload
Lidgren.Network Namespace
Reads a 32 bit signed integer written using Write(Int32)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool ReadInt32(
    out int result
)

Visual Basic

Public Function ReadInt32 (_
    <OutAttribute> ByRef result As Integer _
) As Boolean

Visual C++

public:
bool ReadInt32(
    [OutAttribute] int% result
)

Parameters

result
Type: System::Int32%


Return Value

See Also

NetBuffer Class
ReadInt32 Overload
Lidgren.Network Namespace
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadInt64()</code></td>
<td>Reads a 64 bit signed integer written using <code>Write(Int64)</code></td>
</tr>
<tr>
<td><code>ReadInt64(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 64 bits, written using <code>Write(Int64, Int32)</code></td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads a 64 bit signed integer written using Write(Int64)

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public long ReadInt64()
```

Visual Basic

```vbnet
Public Function ReadInt64 As Long
```

Visual C++

```cpp
public:
long long ReadInt64()
```

Return Value

See Also

NetBuffer Class
ReadInt64 Overload
Lidgren.Network Namespace
Reads a signed integer stored in 1 to 64 bits, written using Write(Int64, Int32)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public long ReadInt64(
    int numberOfBits
)

Visual Basic

Public Function ReadInt64 ( _
    numberOfBits As Integer _
) As Long

Visual C++

public:
    long long ReadInt64(
        int numberOfBits
    )

Parameters

numberOfBits
Type: System::::Int32


Return Value

See Also

NetBuffer Class
ReadInt64 Overload
Lidgren.Network Namespace
Reads a stored IPv4 endpoint description

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public IPEndPoint ReadIPEndPoint()
```

Visual Basic

```vbnet
Public Function ReadIPEndPoint As IPEndPoint
```

Visual C++

```cpp
public:
    IPEndPoint^ ReadIPEndPoint()
```

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void ReadPadBits()

Visual Basic

Public Sub ReadPadBits

Visual C++

public:

void ReadPadBits()
See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer:::ReadRangedInteger Method

NetBuffer Class See Also Send Feedback

Reads a 32 bit integer value written using WriteRangedInteger()

Namespace: Lidgren.Network
Syntax

C#

public int ReadRangedInteger(
    int min,
    int max
)

Visual Basic

Public Function ReadRangedInteger ( _
    min As Integer, _
    max As Integer _
) As Integer

Visual C++

public:
    int ReadRangedInteger(
    int min,
    int max
)

Parameters

min
    Type: System::Int32
    The minimum value used when writing the value

max
    Type: System::Int32
    The maximum value used when writing the value

Return Value

A signed integer value larger or equal to MIN and smaller or equal to MAX
See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a 32 bit floating point value written using WriteRangedSingle()

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public float ReadRangedSingle(
    float min,
    float max,
    int numberOfBits
)

Visual Basic

Public Function ReadRangedSingle ( _
    min As Single, _
    max As Single, _
    numberOfBits As Integer _
) As Single

Visual C++

public:
    float ReadRangedSingle(
        float min,
        float max,
        int numberOfBits
    )

Parameters

min
    Type: System::Single
    The minimum value used when writing the value

max
    Type: System::Single
    The maximum value used when writing the value

numberOfBits
    Type: System::Int32
The number of bits used when writing the value

**Return Value**

A floating point value larger or equal to MIN and smaller or equal to MAX
See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer:::ReadSByte Method

NetBuffer Class See Also Send Feedback

Reads a signed byte

Namespace: Lidgren.Network
Syntax

C#

public sbyte ReadSByte()

Visual Basic

Public Function ReadSByte As SByte

Visual C++

public:
signed char ReadSByte()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a 32 bit floating point value written using WriteSignedSingle()

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public float ReadSignedSingle(
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Function ReadSignedSingle ( _
    numberOfBits As Integer _
) As Single
```

Visual C++

```cpp
public:
float ReadSignedSingle(
    int numberOfBits
)
```

Parameters

numberOfBits
Type: System::Int32
The number of bits used when writing the value

Return Value

A floating point value larger or equal to -1 and smaller or equal to 1
See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer::ReadSingle Method

NetBuffer Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadSingle()()()</code></td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td><code>ReadSingle(Single%)</code></td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads a 32 bit floating point value written using Write(Single)

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public float ReadSingle()

Visual Basic

Public Function ReadSingle As Single

Visual C++

public:
  float ReadSingle()

Return Value

See Also

NetBuffer Class
ReadSingle Overload
Lidgren.Network Namespace
Reads a 32 bit floating point value written using Write(Single)

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public bool ReadSingle(
    out float result
)
```

### Visual Basic

```vbnet
Public Function ReadSingle (_
    <OutAttribute> ByRef result As Single _
) As Boolean
```

### Visual C++

```cpp
public:
    bool ReadSingle(_
        [OutAttribute] float% result
    )
```

## Parameters

**result**

Type: `System::Single%`


## Return Value

See Also

NetBuffer Class
ReadSingle Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer....ReadString Method
NetBuffer Class See Also Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadString()()()()</td>
<td>Reads a string written using Write(string)</td>
</tr>
<tr>
<td>ReadString(String%)</td>
<td>Reads a string written using Write(string) and returns true for success</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer..:..ReadStream Method
NetBuffer Class See Also Send Feedback

Reads a string written using Write(string)

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public string ReadString()
```

**Visual Basic**

```vbnet
Public Function ReadString As String
```

**Visual C++**

```cpp
public:
String^ ReadString()
```

**Return Value**

See Also

NetBuffer Class
ReadString Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer..::..ReadString Method (String %)
NetBuffer Class See Also Send Feedback

Reads a string written using Write(string) and returns true for success

Namespace: Lidgren.Network
Syntax

C#

```csharp
public bool ReadString(
    out string result
)
```

Visual Basic

```vbnet
Public Function ReadString (_
    <OutAttribute> ByRef result As String _
) As Boolean
```

Visual C++

```cpp
public:
bool ReadString(
    [OutAttribute] String^% result
)
```

Parameters

result

Type: `System::::String`


Return Value

See Also

NetBuffer Class
ReadString Overload
Lidgren.Network Namespace
C# Visual Basic Visual C++

Lidgren Network Library documentation

NetBuffer.READTIME.ReadTime Method

NetBuffer Class See Also Send Feedback

Reads a value, in local time comparable to NetTime.Now, written using WriteTime() for the connection supplied

Namespace: Lidgren.Net
## Syntax

### C#

```csharp
public double ReadTime(
    NetConnection connection,
    bool highPrecision
)
```

### Visual Basic

```vbnet
Public Function ReadTime ( _
    connection As NetConnection, _
    highPrecision As Boolean _
) As Double
```

### Visual C++

```cpp
public: double ReadTime(
    NetConnection^ connection,
    bool highPrecision
)
```

## Parameters

**connection**
- Type: `Lidgren.Network.NetConnection`


**highPrecision**
- Type: `System.Boolean`


## Return Value
[Missing <returns> documentation for
See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a 16 bit unsigned integer written using Write(UInt16)

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public ushort ReadUInt16()

Visual Basic

Public Function ReadUInt16 As UShort

Visual C++

public:
unsigned short ReadUInt16()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer...:..ReadUInt32 Method

NetBuffer Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadUInt32()()()</td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32)</td>
</tr>
<tr>
<td>ReadUInt32(Int32)</td>
<td>Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32)</td>
</tr>
<tr>
<td>ReadUInt32(UInt32%)</td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer::ReadUInt32 Method

NetBuffer Class See Also Send Feedback

Reads an 32 bit unsigned integer written using Write(UInt32)

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public uint ReadUInt32()
```

**Visual Basic**

```vbnet
Public Function ReadUInt32 As UInteger
```

**Visual C++**

```c++
public:
unsigned int ReadUInt32()
```

**Return Value**

See Also

NetBuffer Class
ReadUInt32 Overload
Lidgren.Network Namespace
NetBuffer..::.ReadUInt32 Method (Int32)

Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32)

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public uint ReadUInt32(
    int numberOfBits
)

Visual Basic

Public Function ReadUInt32 ( _
    numberOfBits As Integer _
) As UInteger

Visual C++

public:
    unsigned int ReadUInt32(
        int numberOfBits
    )

Parameters

numberOfBits
Type: System::::Int32


Return Value

See Also

NetBuffer Class
ReadUInt32 Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer...:...ReadUInt32 Method (UInt32%)

**NetBuffer Class**  **See Also**  **Send Feedback**

Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success

**Namespace:** [Lidgren.Network](#)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool ReadUInt32(
    out uint result
)

Visual Basic

Public Function ReadUInt32 (_
    <OutAttribute> ByRef result AsUInteger _
) As Boolean

Visual C++

public:
    bool ReadUInt32(
        [OutAttribute] unsigned int% result
    )

Parameters

result
    Type: System::::<UInt32%

[Missing <param name="result"/> documentation for "M:Lidgren.Network.NetBuffer.ReadUInt32(System.UInt32@)"

Return Value

See Also

NetBuffer Class
ReadUInt32 Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Include Protected Members  Include Inherited Members
Lidgren Network Library documentation
NetBuffer:::ReadUInt64 Method
NetBuffer Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadUInt64()()()</td>
<td>Reads a 64 bit unsigned integer written using Write(UInt64)</td>
</tr>
<tr>
<td>ReadUInt64(Int32)</td>
<td>Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64, Int32)</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads a 64 bit unsigned integer written using Write(UInt64)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public ulong ReadUInt64()

Visual Basic

Public Function ReadUInt64 As ULong

Visual C++

public:
unsigned long long ReadUInt64()

Return Value

See Also

NetBuffer Class
ReadUInt64 Overload
Lidgren.Network Namespace
Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64, Int32)

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public ulong ReadUInt64(
    int numberOfBits
)
```

### Visual Basic

```vbnet
Public Function ReadUInt64 ( _
    numberOfBits As Integer _
) As ULong
```

### Visual C++

```cpp
public:
    unsigned long long ReadUInt64(
        int numberOfBits
    )
```

## Parameters

**numberOfBits**
- Type: `System::Int32`


## Return Value

See Also

NetBuffer Class
ReadUInt64 Overload
Lidgren.Network Namespace
Reads a 32 bit floating point value written using WriteUnitSingle()

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public float ReadUnitSingle(
    int numberOfBits
)

Visual Basic

Public Function ReadUnitSingle ( _
    numberOfBits As Integer _
) As Single

Visual C++

public:
float ReadUnitSingle(
    int numberOfBits
)

Parameters

numberOfBits

Type: System::...Int32

The number of bits used when writing the value

Return Value

A floating point value larger or equal to 0 and smaller or equal to 1
See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer.::.::.ReadVariableInt32 Method

**NetBuffer Class** See Also Send Feedback

Reads a variable sized Int32 written using WriteVariableInt32()

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public int ReadVariableInt32()
```

**Visual Basic**

```vbnet
Public Function ReadVariableInt32 As Integer
```

**Visual C++**

```cpp
public:
    int ReadVariableInt32()
```

## Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Reads a variable sized Int64 written using WriteVariableInt64()

Namespace: Lidgren.Network
Syntax

C#

public long ReadVariableInt64()

Visual Basic

Public Function ReadVariableInt64 As Long

Visual C++

public:
long long ReadVariableInt64()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer...:..ReadVariableUInt32 Method
NetBuffer Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadVariableUInt32()</td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32()</td>
</tr>
<tr>
<td>ReadVariableUInt32(UInt32%)</td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Reads a variable sized UInt32 written using WriteVariableUInt32()
Syntax

C#

public uint ReadVariableUInt32()

Visual Basic

Public Function ReadVariableUInt32 As UInteger

Visual C++

public:
unsigned int ReadVariableUInt32()

Return Value

See Also

NetBuffer Class
ReadVariableUInt32 Overload
Lidgren.Network Namespace
Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool ReadVariableUInt32(
    out uint result
)

Visual Basic

Public Function ReadVariableUInt32 ( _
    <OutAttribute> ByRef result As UInteger _
) As Boolean

Visual C++

public:
bool ReadVariableUInt32(
    [OutAttribute] unsigned int% result
)

Parameters

result
Type: System::...::UInt32%

[Missing <param name="result"/> documentation for "M:Lidgren.Network.NetBuffer.ReadVariableUInt32(System.UInt32@)"

Return Value

See Also

NetBuffer Class
ReadVariableUInt32 Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer...:::ReadVariableUInt64 Method

NetBuffer Class  See Also  Send Feedback

Reads a variable sized UInt32 written using WriteVariableInt64()

Namespace: Lidgren.Network
Syntax

C#

public ulong ReadVariableUInt64()

Visual Basic

Public Function ReadVariableUInt64 As ULong

Visual C++

public:
unsigned long long ReadVariableUInt64()

Return Value

See Also

NetBuffer Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer::SkipPadBits Method

NetBuffer Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SkipPadBits()</td>
<td>Pads data with enough bits to reach a full byte.</td>
</tr>
<tr>
<td>SkipPadBits(Int32)</td>
<td>Decreases cpu usage for subsequent byte writes.</td>
</tr>
<tr>
<td></td>
<td>Pads data with the specified number of bits.</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void SkipPadBits()

Visual Basic

Public Sub SkipPadBits

Visual C++

public:
void SkipPadBits()
See Also

NetBuffer Class
SkipPadBits Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer..::.SkipPadBits Method (Int32)

**NetBuffer Class**  **See Also**  **Send Feedback**

Pads data with the specified number of bits.

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
(2012.1.7.0)
Syntax

C#

public void SkipPadBits(
        int numberOfBits
    )

Visual Basic

Public Sub SkipPadBits ( _
        numberOfBits As Integer _
    )

Visual C++

public:  
void SkipPadBits(
        int numberOfBits
    )

Parameters

numberOfBits
Type: System::Int32

See Also

NetBuffer Class
SkipPadBits Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer...Write Method

NetBuffer Class See Also Send Feedback
Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write(Boolean)</td>
<td>Writes a boolean value using 1 bit</td>
</tr>
<tr>
<td>Write(Byte)</td>
<td>Write a byte</td>
</tr>
<tr>
<td>Write(array&lt;Byte&gt;[][])</td>
<td>Writes all bytes in an array</td>
</tr>
<tr>
<td>Write(Double)</td>
<td>Writes a 64 bit floating point value</td>
</tr>
<tr>
<td>Write(Int16)</td>
<td>Writes a signed 16 bit integer</td>
</tr>
<tr>
<td>Write(Int32)</td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td>Write(Int64)</td>
<td>Writes a 64 bit signed integer</td>
</tr>
<tr>
<td>Write(IPEndPoint)</td>
<td>Writes an endpoint description</td>
</tr>
<tr>
<td>Write(SByte)</td>
<td>Writes a signed byte</td>
</tr>
<tr>
<td>Write(Single)</td>
<td>Writes a 32 bit floating point value</td>
</tr>
<tr>
<td>Write(String)</td>
<td>Write a string</td>
</tr>
<tr>
<td>Write(UInt16)</td>
<td>Writes an unsigned 16 bit integer</td>
</tr>
<tr>
<td>Write(UInt32)</td>
<td>Writes a 32 bit unsigned integer</td>
</tr>
<tr>
<td>Write(UInt64)</td>
<td>Writes a 64 bit unsigned integer</td>
</tr>
<tr>
<td>Write(NetIncomingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(NetOutgoingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(Byte, Int32)</td>
<td>Writes 1 to 8 bits of a byte</td>
</tr>
<tr>
<td>Write(Int32, Int32)</td>
<td>Writes a signed integer using 1 to 32 bits</td>
</tr>
<tr>
<td>Write(Int64, Int32)</td>
<td>Writes a signed integer using 1 to 64 bits</td>
</tr>
<tr>
<td>Write(UInt16, Int32)</td>
<td>Writes an unsigned integer using 1 to 16 bits</td>
</tr>
<tr>
<td>Write(UInt32, Int32)</td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td>Write(UInt64, Int32)</td>
<td>Writes an unsigned integer using 1 to 64 bits</td>
</tr>
<tr>
<td>Write(array&lt;Byte&gt;[][])</td>
<td>Writes the specified number of bytes</td>
</tr>
</tbody>
</table>
(Int32, Int32) from an array
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
NetBuffer Class See Also Send Feedback

Writes a boolean value using 1 bit

Namespace: Lidgren.Network
Syntax

C#

public void Write(
    bool value
)

Visual Basic

Public Sub Write ( _
    value As Boolean _
)

Visual C++

public:
void Write(
    bool value
)

Parameters

documentation for

value
Type: System::Boolean

[Missing <param name="value" /> documentation for
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Write a byte

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Write(byte source)

Visual Basic

Public Sub Write(_
    source As Byte _
)

Visual C++

public:
    void Write(
        unsigned char source
    )

Parameters

source

Type: System.Byte

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
NetBuffer class documentation

NetBuffer Write Method (array<Byte>[][]){}

writes all bytes in an array

Namespace: Lidgren.Network
Syntax

C#

public void Write(
    byte[] source
)

Visual Basic

Public Sub Write ( _
    source As Byte() _
)

Visual C++

public:
    void Write(
        array<unsigned char>^ source
    )

Parameters

source
    Type: array<System..::..Byte>[]>()[]

[Missing <param name="source"/> documentation for
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
C#  Visual Basic
Visual C++

Lidgren Network Library documentation
NetBuffer....Write Method (Double)

NetBuffer Class See Also Send Feedback

Writes a 64 bit floating point value

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public void Write(
    double source
)
```

### Visual Basic

```vbnet
Public Sub Write (_
    source As Double _
)
```

### Visual C++

```cpp
public:
    void Write(
        double source
    )
```

## Parameters

- **source**
  - Type: `System::Double`

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer:::..Write Method (Int16)

NetBuffer Class  See Also  Send Feedback

Writes a signed 16 bit integer

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public void Write(
    short source
)
```

### Visual Basic

```vbnet
Public Sub Write (_
    source As Short _
)
```

### Visual C++

```cpp
public:
void Write(
    short source
)
```

## Parameters

### source

Type: **System::::Int16**

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer...::Write Method (Int32)

NetBuffer Class See Also Send Feedback

Writes a 32 bit signed integer

Namespace: Lidgren.Network
Syntax

C#

public void Write(
    int source
)

Visual Basic

Public Sub Write (
    source As Integer _
)

Visual C++

public:
void Write(
    int source
)

Parameters

source
Type: System::::Int32

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
NetBuffer Class: See Also: Send Feedback

Writes a 64 bit signed integer

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void Write(
    long source
)
```

Visual Basic

```vbnet
Public Sub Write ( _
    source As Long _
)
```

Visual C++

```cpp
public:
void Write(
    long long source
)
```

Parameters

source

Type: System::Int64

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer::Write Method (IPEndPoint)

NetBuffer Class See Also Send Feedback

Writes an endpoint description

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Write(
    IPEndPoint endPoint
)

Visual Basic

Public Sub Write (_
    endPoint As IPEndPoint _
)

Visual C++

public:
void Write(
    IPEndPoint^ endPoint
)

Parameters

d endPoint
Type: System.Net.IP.EndPoint

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer...::Write Method (SByte)

NetBuffer Class See Also Send Feedback

Wrote a signed byte

Namespace: Lidgren.Network
Syntax

C#

public void Write(
    sbyte source
)

Visual Basic

Public Sub Write ( _
    source As SByte _
)

Visual C++

public:
    void Write(
        signed_char source
    )

Parameters

source
    Type: System::::SByte

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Writes a 32 bit floating point value

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
# Syntax

**C#**

```csharp
public void Write(
    float source
)
```

**Visual Basic**

```vbnet
Public Sub Write ( _
    source As Single _
)
```

**Visual C++**

```cpp
public:
void Write(
    float source
)
```

## Parameters

**source**

Type: `System::Single`

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Write a string

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0

(2012.1.7.0)
Syntax

C#

public void Write(
    string source
)

Visual Basic

Public Sub Write ( _
    source As String _
)

Visual C++

public:
    void Write(
        String^ source
    )

Parameters

source
    Type: System:::String

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Exports an unsigned 16 bit integer

*Namespace:* Lidgren.Network
Syntax

C#

```csharp
public void Write(
    ushort source
)
```

Visual Basic

```vbnet
Public Sub Write ( 
    source As UShort 
)
```

Visual C++

```cpp
public:
void Write(
    unsigned short source
)
```

Parameters

source
Type: `System::::::UInt16`

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer:::Write Method (UInt32)

NetBuffer Class  See Also  Send Feedback

Writes a 32 bit unsigned integer

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
# Syntax

**C#**

```csharp
public void Write(
    uint source
)
```

**Visual Basic**

```vbnet
Public Sub Write (_, source As UInteger _)
```

**Visual C++**

```c++
public:
void Write(
    unsigned int source
)
```

## Parameters

**source**

Type: [System::::UInt32](#)

[Missing <param name="source"/> documentation for "M:Lidgren.Network.NetBuffer.Write(System(UInt32)"")]
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
C# Visual Basic
Visual C++
Lidgren Network Library documentation
NetBuffer...:..:Write Method (UInt64)

NetBuffer Class See Also Send Feedback

Writes a 64 bit unsigned integer

Namespace: Lidgren.Network
Syntax

C#

public void Write(
    ulong source
)

Visual Basic

Public Sub Write (_
    source As ULong _
)

Visual C++

public:
void Write(
    unsigned long long source
)

Parameters

source
Type: System::::UInt64

[Missing <param name="source"/> documentation for "M:Lidgren.Network.NetBuffer.Write(System.UInt64)" ]
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Append all the bits of message to this message

**Namespace:** [Lidgren.Network](http://Lidgren.Network)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Write(
    NetIncomingMessage message
)

Visual Basic

Public Sub Write ( _
    message As NetIncomingMessage _
)

Visual C++

public:
    void Write(
        NetIncomingMessage^ message
    )

Parameters

message
    Type: Lidgren.Network..NetIncomingMessage

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Append all the bits of message to this message

Namespace: Lidgren.Network  
Syntax

C#

public void Write(
    NetOutgoingMessage message
)

Visual Basic

Public Sub Write ( _
    message As NetOutgoingMessage _
)

Visual C++

public:
void Write(
    NetOutgoingMessage^ message
)

Parameters

message
Type: Lidgren.NetworkNetMessage

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
NetBuffer Class

Write Method (Byte, Int32)

Writers 1 to 8 bits of a byte

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public void Write(
    byte source,
    int numberOfBits
)
```

**Visual Basic**

```vbnet
Public Sub Write ( _
    source As Byte, _
    numberOfBits As Integer _
)
```

**Visual C++**

```c++
publish:
void Write(
    unsigned char source,
    int numberOfBits
)
```

**Parameters**

**source**

Type: **System..::.Byte**

[Missing <param name="source"/> documentation for
]

**numberOfBits**

Type: **System..::.Int32**

[Missing <param name="numberOfBits"/> documentation for
]
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Lidgren Network Library documentation
NetBuffer:::Write Method (Int32, Int32)
NetBuffer Class  See Also  Send Feedback

Writes a signed integer using 1 to 32 bits

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public void Write(
    int source,
    int numberOfBits
)
```

**Visual Basic**

```vbnet
Public Sub Write ( _
    source As Integer, _
    numberOfBits As Integer _
)
```

**Visual C++**

```cpp
public:
void Write(
    int source,
    int numberOfBits
)
```

Parameters

**source**
Type: System::::Int32


**numberOfBits**
Type: System::::Int32

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++

Lidgren Network Library documentation
NetBuffer...:..Write Method (Int64, Int32)

NetBuffer Class  See Also  Send Feedback

Writes a signed integer using 1 to 64 bits

Namespace:  Lidgren.Network
Syntax

C#

```csharp
public void Write(
    long source,
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Sub Write (_
    source As Long,
    numberOfBits As Integer _
)
```

Visual C++

```cpp
public:
void Write(
    long long source,
    int numberOfBits
)
```

Parameters

source
Type: System::..::Int64

numberOfBits
Type: System::..::Int32
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
C#  Visual Basic
 iframe
 Visual C++

Lidgren Network Library documentation
NetBuffer...:::Write Method (UInt16, Int32)
 NetBuffer Class  See Also  Send Feedback
 Writes an unsigned integer using 1 to 16 bits

Namespace: Lidgren.Network
(2012.1.7.0)
Syntax

C#

public void Write(
    ushort source,
    int numberOfBits
)

Visual Basic

Public Sub Write ( _
    source As UShort, _
    numberOfBits As Integer _
)

Visual C++

public:
void Write(
    unsigned short source,
    int numberOfBits
)

Parameters

source
Type: System::::UInt16


numberOfBits
Type: System::::Int32

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer:::Write Method (UInt32, Int32)

NetBuffer Class See Also Send Feedback

Writes a 32 bit signed integer

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public void Write(
    uint source,
    int numberOfBits
)
```

**Visual Basic**

```vbasic
Public Sub Write (_
    source As UInteger, _
    numberOfBits As Integer _
)
```

**Visual C++**

```c++
public:
void Write(
    unsigned int source,
    int numberOfBits
)
```

**Parameters**

**source**
Type: `System::::UInt32`


**numberOfBits**
Type: `System::::Int32`

See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
NetBuffer::Write Method (UInt64, Int32)

Namespaces: Lidgren.Network

Writes an unsigned integer using 1 to 64 bits.
Syntax

C#

public void Write(
    ulong source,
    int numberOfBits
)

Visual Basic

Public Sub Write ( _
    source As ULong, _
    numberOfBits As Integer _
)

Visual C++

public:
void Write(
    unsigned long long source,
    int numberOfBits
)

Parameters

source
Type: System::UInt64

numberOfBits
Type: System::Int32
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
C# Visual Basic Visual C++

Lidgren Network Library documentation
NetBuffer::Write Method (array<Byte>[0][0], Int32, Int32)

NetBuffer Class See Also Send Feedback

Writes the specified number of bytes from an array

Namespace: Lidgren.Network
### Syntax

#### C#

```csharp
public void Write(
    byte[] source,
    int offsetInBytes,
    int numberOfBytes
)
```

#### Visual Basic

```vbnet
Public Sub Write (_
    source As Byte(), _
    offsetInBytes As Integer, _
    numberOfBytes As Integer _
)
```

#### Visual C++

```cpp
public:
void Write(
    array<unsigned char>^ source, 
    int offsetInBytes, 
    int numberOfBytes
)
```

### Parameters

**source**
- Type: `array<System:::Byte>[]`[]


**offsetInBytes**
- Type: `System:::Int32`

numberOfBytes
Type: System..::.Int32

[Missing <param name="numberOfBytes"/> documentation for
See Also

NetBuffer Class
Write Overload
Lidgren.Network Namespace
NetBuffer Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>WriteAllFields(Object)</code></td>
<td>Writes all public and private declared instance fields of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td><code>WriteAllFields(Object, BindingFlags)</code></td>
<td>Writes all fields with specified binding in alphabetical order using reflection</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetBuffer..::.WriteAllFields Method (Object)

NetBuffer Class See Also Send Feedback

Writes all public and private declared instance fields of the object in alphabetical order using reflection

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public void WriteAllFields(
    Object ob
)
```

### Visual Basic

```vbnet
Public Sub WriteAllFields ( _
    ob As Object _
)
```

### Visual C++

```cpp
public:
void WriteAllFields(
    Object^ ob
)
```

### Parameters

**ob**

Type: System..::.Object

See Also

NetBuffer Class
WriteAllFields Overload
Lidgren.Network Namespace
NetBuffer Class

WriteAllFields Method (Object, BindingFlags)

Namespace: Lidgren.Network

Writes all fields with specified binding in alphabetical order using reflection
Syntax

C#

```csharp
public void WriteAllFields(
    Object ob,
    BindingFlags flags
)
```

Visual Basic

```vbnet
Public Sub WriteAllFields ( _
    ob As Object, _
    flags As BindingFlags _
)
```

Visual C++

```cpp
public:
void WriteAllFields(
    Object^ ob,
    BindingFlags flags
)
```

Parameters

**ob**

Type: **System::Object**


**flags**

Type: **System::Reflection:: BindingFlags**

See Also

NetBuffer Class
WriteAllFields Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer::WriteAllProperties Method

NetBuffer Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteAllProperties(Object)</td>
<td>Writes all public and private declared instance properties of the object in alphabetical order using reflection</td>
</tr>
<tr>
<td>WriteAllProperties(Object, BindingFlags)</td>
<td>Writes all properties with specified binding in alphabetical order using reflection</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
NetBuffer Class

See Also: Send Feedback

Writes all public and private declared instance properties of the object in alphabetical order using reflection

Namespace: Lidgren.Network
Syntax

C#

public void WriteAllProperties(
    Object ob
)

Visual Basic

Public Sub WriteAllProperties (_
    ob As Object _
)

Visual C++

public:
void WriteAllProperties(
    Object^ ob
)

Parameters

ob

Type: System::Object

See Also

NetBuffer Class
WriteAllProperties Overload
Lidgren.Network Namespace
NetBuffer.WriteAllProperties Method (Object, BindingFlags)

Writes all properties with specified binding in alphabetical order using reflection

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0

(2012.1.7.0)
Syntax

C#

```csharp
public void WriteAllProperties(
    Object ob,
    BindingFlags flags
)
```

Visual Basic

```vbnet
Public Sub WriteAllProperties (_
    ob As Object, _
    flags As BindingFlags _
)
```

Visual C++

```cpp
public:
void WriteAllProperties(
    Object^ ob,
    BindingFlags flags
)
```

Parameters

ob

Type: `System::Object`


flags

Type: `System.Reflection::BindingFlags`

See Also

NetBuffer Class
WriteAllProperties Overload
Lidgren.Network Namespace
NetBuffer Class
See Also
Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WritePadBits()()()</td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.</td>
</tr>
<tr>
<td>WritePadBits(Int32)</td>
<td>Pads data with the specified number of bits.</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public void WritePadBits()
```

**Visual Basic**

```vbnet
Public Sub WritePadBits
```

**Visual C++**

```cpp
public:
    void WritePadBits()
```
See Also

NetBuffer Class
WritePadBits Overload
Lidgren.Network Namespace
Pads data with the specified number of bits.

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void WritePadBits(
    int numberOfBits
)

Visual Basic

Public Sub WritePadBits (_
    numberOfBits As Integer _
)

Visual C++

public:
void WritePadBits(
    int numberOfBits
)

Parameters

numberOfBits
Type: System::Int32

See Also

NetBuffer Class
WritePadBits Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++

Lidgren Network Library documentation

NetBuffer..::.WriteRangedInteger Method

NetBuffer Class  See Also  Send Feedback

Writes an integer with the least amount of bits need for the specified range
Returns number of bits written

Namespace:  Lidgren.Network
### Syntax

**C#**

```csharp
public int WriteRangedInteger(
    int min,
    int max,
    int value
)
```

**Visual Basic**

```vbnet
Public Function WriteRangedInteger (_
    min As Integer, _
    max As Integer, _
    value As Integer _
) As Integer
```

**Visual C++**

```cpp
public:
int WriteRangedInteger(
    int min,
    int max,
    int value
)
```

### Parameters

**min**  
Type: [System:::Int32](#)


**max**  
Type: [System:::Int32](#)

value

Type: System..::.Int32

[Missing <param name="value"/> documentation for

Return Value

[Missing <returns> documentation for
See Also

NetBuffer Class
Lidgren.Network Namespace
Compress a float within a specified range using a certain number of bits

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void WriteRangedSingle(
    float value,
    float min,
    float max,
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Sub WriteRangedSingle ( _
    value As Single, _
    min As Single, _
    max As Single, _
    numberOfBits As Integer _
)
```

Visual C++

```cpp
public:
void WriteRangedSingle(
    float value,
    float min,
    float max,
    int numberOfBits
)
```

Parameters

value

Type: `System::::Single`

min

Type: `System::::Single`
max
Type: **System..::.Single**

numberOfBits
Type: **System..::.Int32**
See Also

NetBuffer Class
Lidgren.Network Namespace
Compress (lossy) a float in the range -1..1 using `numberOfBits` bits

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void WriteSignedSingle(
    float value,
    int numberOfBits
)

Visual Basic

Public Sub WriteSignedSingle ( _
    value As Single, _
    numberOfBits As Integer _
)

Visual C++

public:
void WriteSignedSingle(
    float value,
    int numberOfBits
)

Parameters

value
Type: System::Single

numberOfBits
Type: System::Int32
See Also

NetBuffer Class
Lidgren.Network Namespace
NetBuffer Class

See Also

Send Feedback

NetBuffer...WriteTime Method

Include Inherited Members

Include Protected Members

Lidgren Network Library documentation

Visual Basic

Visual C++
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>WriteTime(Boolean)</code></td>
<td>Writes the current local time to a message; readable (and convertable to local time) by the remote host using <code>ReadTime()</code></td>
</tr>
<tr>
<td><code>WriteTime(Double, Boolean)</code></td>
<td>Writes a local timestamp to a message; readable (and convertable to local time) by the remote host using <code>ReadTime()</code></td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
NetBuffer Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer.WriteTime Method (Boolean)

NetBuffer Class  See Also  Send Feedback

Writes the current local time to a message; readable (and convertible to local time) by the remote host using ReadTime()

Namespace: Lidgren.Network
Syntax

C#

public void WriteTime(
    bool highPrecision
)

Visual Basic

Public Sub WriteTime ( _
    highPrecision As Boolean _
)

Visual C++

public:
void WriteTime(
    bool highPrecision
)

Parameters

highPrecision
Type: System::Boolean

See Also

NetBuffer Class
WriteTime Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetBuffer..::.WriteTime Method (Double, Boolean)

NetBuffer Class  See Also  Send Feedback

Writes a local timestamp to a message; readable (and convertable to local time) by the remote host using ReadTime()

Namespace:  Lidgren.Network
Syntax

C#

public void WriteTime(
double localTime,
bool highPrecision
)

Visual Basic

Public Sub WriteTime ( _
localTime As Double, _
highPrecision As Boolean _
)

Visual C++

public:
void WriteTime(
double localTime,
bool highPrecision
)

Parameters

localTime
Type: System::Double


highPrecision
Type: System::Boolean

See Also

NetBuffer Class
WriteTime Overload
Lidgren.Network Namespace
Compress (lossy) a float in the range 0..1 using `numberOfBits` bits.

Namespace: `Lidgren.Network`
Syntax

C#

```csharp
public void WriteUnitSingle(
    float value,
    int numberOfBits
)
```

Visual Basic

```vbnet
Public Sub WriteUnitSingle ( _
    value As Single, _
    numberOfBits As Integer _
)
```

Visual C++

```cpp
public:
void WriteUnitSingle(
    float value,
    int numberOfBits
)
```

Parameters

value
Type: System:::Single

[Missing <param name="value"/> documentation for
]

numberOfBits
Type: System:::Int32

[Missing <param name="numberOfBits"/> documentation for
]
See Also

NetBuffer Class
Lidgren.Network Namespace
Write Base128 encoded variable sized signed integer of up to 32 bits

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public int WriteVariableInt32(
    int value
)
```

### Visual Basic

```vbnet
Public Function WriteVariableInt32 ( _
    value As Integer _
) As Integer
```

### Visual C++

```cpp
public:
    int WriteVariableInt32(
    int value
)
```

## Parameters

**value**  
Type: `System::Int32`


## Return Value

number of bytes written
See Also

NetBuffer Class
Lidgren.Network Namespace
Write Base128 encoded variable sized signed integer of up to 64 bits

Namespace: Lidgrenetwork
Syntax

C#

public int WriteVariableInt64(
    long value
)

Visual Basic

Public Function WriteVariableInt64 ( _
    value As Long _
) As Integer

Visual C++

public:
    int WriteVariableInt64(
        long long value
    )

Parameters

value
    Type: System::Int64

[Missing <param name="value"/> documentation for
]

Return Value

number of bytes written
See Also

NetBuffer Class
Lidgren.Network Namespace
Write Base128 encoded variable sized unsigned integer of up to 32 bits.

Namespace: Lidgren.Network
Syntax

C#

public int WriteVariableUInt32(
    uint value
)

Visual Basic

Public Function WriteVariableUInt32 ( _
    value As UInteger _
) As Integer

Visual C++

public:
int WriteVariableUInt32(
    unsigned_int value
)

Parameters

value
Type: System:::UInt32

[Missing <param name="value"/> documentation for "M:Lidgren.Network.NetBuffer.WriteVariableUInt32(System.UInt32)"]

Return Value

number of bytes written
See Also

NetBuffer Class
Lidgren.Network Namespace
Write Base128 encoded variable sized unsigned integer of up to 64 bits

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public int WriteVariableUInt64(
    ulong value
)
```

### Visual Basic

```vbnet
Public Function WriteVariableUInt64 (_
    value As ULong _
) As Integer
```

### Visual C++

```cpp
public:
    int WriteVariableUInt64(
        unsigned long long value
    )
```

## Parameters

**value**

Type: `System::::::UInt64`

[Missing <param name="value"/> documentation for "M:Lidgren.Network.NetBuffer.WriteVariableUInt64(System.UInt64)""]

## Return Value

number of bytes written
See Also

NetBuffer Class
Lidgren.Network Namespace
The **NetBuffer** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>Gets or sets the internal data buffer</td>
</tr>
<tr>
<td><strong>LengthBits</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bits</td>
</tr>
<tr>
<td><strong>LengthBytes</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bytes</td>
</tr>
<tr>
<td><strong>Position</strong></td>
<td>Gets or sets the read position in the buffer, in bits (not bytes)</td>
</tr>
<tr>
<td><strong>PositionInBytes</strong></td>
<td>Gets the position in the buffer in bytes; note that the bits of the first returned byte may already have been read - check the Position property to make sure.</td>
</tr>
</tbody>
</table>
See Also

NetBuffer Class
Lidgren.Network Namespace
NetBuffer...:::..Data Property

NetBuffer Class See Also Send Feedback

Gets or sets the internal data buffer

Namespace: Lidgren.Network
Syntax

C#

public byte[] Data { get; set; }

Visual Basic

Public Property Data As Byte()
Get
Set

Visual C++

public:
property array<unsigned char>^ Data {
array<unsigned char>^ get ();
void set (array<unsigned char>^ value);
}
See Also

NetBuffer Class
Lidgren.Network Namespace
Gets or sets the length of the used portion of the buffer in bits

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
### Syntax

**C#**

```csharp
public int LengthBits { get; set; }
```

**Visual Basic**

```vbnet
Public Property LengthBits As Integer
    Get
    Set
```

**Visual C++**

```cpp
public:
property int LengthBits {
    int get ();
    void set (int value);
}
```
See Also

NetBuffer Class
Lidgren.Network Namespace
Gets or sets the length of the used portion of the buffer in bytes

**Namespace:** [Lidgren.Network](#)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
---Syntax---

**C#**

```csharp
public int LengthBytes { get; set; }
```

**Visual Basic**

```vbnet
Public Property LengthBytes As Integer
    Get
    Set
```

**Visual C++**

```cpp
public:
    property int LengthBytes {
        int get ();
        void set (int value);
    }
```
See Also

NetBuffer Class
Lidgren.Network Namespace
Gets or sets the read position in the buffer, in bits (not bytes)

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public long Position { get; set; }

Visual Basic

Public Property Position As Long
    Get
    Set

Visual C++

public:
    property long long Position {
        long long get ();
        void set (long long value);
    }

See Also

NetBuffer Class
Lidgren.Network Namespace
NetBuffer..::..PositionInBytes Property

Gets the position in the buffer in bytes; note that the bits of the first returned byte may already have been read - check the Position property to make sure.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int PositionInBytes { get; }

Visual Basic

Public ReadOnly Property PositionInBytes As Integer
Get

Visual C++

public:
property int PositionInBytes {
    int get ();
}

See Also

NetBuffer Class
Lidgren.Network Namespace
NetClient Class

Specialized version of NetPeer used for a "client" connection. It does not accept any incoming connections and maintains a ServerConnection property.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public class NetClient : NetPeer

Visual Basic

Public Class NetClient
    Inherits NetPeer

Visual C++

public ref class NetClient : public NetPeer
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetPeer
Lidgren.Network..::..NetClient
See Also

NetClient Members
Lidgren.Network Namespace
The `NetClient` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetClient</td>
<td>NetClient constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect(IPEndPoint)</td>
<td>Create a connection to a remote endpoint (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(IPEndPoint, NetOutgoingMessage)</td>
<td>Connect to a remote server (Overrides NetPeer::Connect(IPEndPoint, NetOutgoingMessage).)</td>
</tr>
<tr>
<td>Connect(String, Int32)</td>
<td>Create a connection to a remote endpoint (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(String, Int32, NetOutgoingMessage)</td>
<td>Create a connection to a remote endpoint (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage()()()()</td>
<td>Creates a new message for sending (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(Int32)</td>
<td>Creates a new message for sending (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(String)</td>
<td>Creates a new message for sending and writes the provided string to it (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Disconnect</td>
<td>Disconnect from server</td>
</tr>
<tr>
<td>DiscoverKnownPeer(IPEndPoint)</td>
<td>Emit a discovery signal to a single known host (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>DiscoverKnownPeer(String, Int32)</td>
<td>Emit a discovery signal to a single host (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>DiscoverLocalPeers</td>
<td>Emit a discovery signal to all hosts on your subnet (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object (Inherited from Object.)</td>
</tr>
<tr>
<td></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before disposal.</td>
</tr>
</tbody>
</table>
- **Finalize**
  - the **Object** is reclaimed by the garbage collector.
  (Inherited from **Object**.)

- **FlushSendQueue**
  - If **NetPeerConfiguration.AutoFlushSendQueue** is false; you need to call this to flush all messages queued using **SendMessage**.
  (Inherited from **NetPeer**.)

- **GetConnection**
  - Get the connection, if any, for a remote endpoint.
  (Inherited from **NetPeer**.)

- **GetHashCode**
  - Serves as a hash function for a particular type.
  (Inherited from **Object**.)

- **GetType**
  - Gets the **Type** of the current instance.
  (Inherited from **Object**.)

- **Introduce**
  - Send **NetIntroduction** message to the host and client; introducing client to host.
  (Inherited from **NetPeer**.)

- **MemberwiseClone**
  - Creates a shallow copy of the current object.
  (Inherited from **Object**.)

- **RawSend**
  - Send raw bytes; only used for debugging.
  (Inherited from **NetPeer**.)

- **ReadMessage**
  - Read a pending message from any connection, if any.
  (Inherited from **NetPeer**.)

- **ReadMessages**
  - Read a pending message from any connection, if any.
  (Inherited from **NetPeer**.)

- **Recycle**(NetIncomingMessage)
  - Recycles a **NetIncomingMessage** instance for reuse; taking pressure off the garbage collector.
  (Inherited from **NetPeer**.)

- **Recycle**(IEnumerable<NetIncomingMessage>)
  - Recycles a list of **NetIncomingMessage** instances for reuse; taking pressure off the garbage collector.
  (Inherited from **NetPeer**.)

Call this to register a callback.
- **RegisterReceivedCallback**
  message arrives
  (Inherited from [NetPeer](#).)

- **SendDiscoveryResponse**
  Send a discovery response
  (Inherited from [NetPeer](#).)

- **SendMessage(NetOutgoingMessage, NetDeliveryMethod)**
  Sends message to server

- **SendMessage(NetOutgoingMessage, NetDeliveryMethod, Int32)**
  Sends message to server

- **SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)**
  Send a message to a specific connection
  (Inherited from [NetPeer](#).)

- **SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)**
  Send a message to a specific connection
  (Inherited from [NetPeer](#).)

- **SendMessage(NetOutgoingMessage, List<Of <'(NetConnection)>>>), NetDeliveryMethod, Int32)**
  Send a message to a list of connections
  (Inherited from [NetPeer](#).)

- **SendUnconnectedMessage(NetOutgoingMessage, IList<Of <='(IPEndPoint)>)>>)**
  Send a message to an unconnected host
  (Inherited from [NetPeer](#).)

- **SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)**
  Send a message to an unconnected host
  (Inherited from [NetPeer](#).)

- **SendUnconnectedMessage(NetOutgoingMessage, String, Int32)**
  Send a message to this exact same netpeer
  (loopback)
  (Inherited from [NetPeer](#).)

- **SendUnconnectedToSelf**
  Disconnects all active connections
  (Inherited from [NetPeer](#).)

- **Shutdown**
  Binds to socket and spawns
  (Inherited from [NetPeer](#).)

- **Start**
  Read a pending message from
  (Inherited from [NetPeer](#).)

- **ToString**
  Returns a string that represents
  (Overrides [Object.:::ToString](#).)

- **WaitMessage**
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>Gets the configuration used to instantiate this NetPeer</td>
</tr>
<tr>
<td>Connections</td>
<td>Gets a copy of the list of connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>ConnectionsCount</td>
<td>Gets the number of active connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>ConnectionStatus</td>
<td>Gets the connection status of the server connection (or NetConnectionStatus.Disconnected if no connection)</td>
</tr>
<tr>
<td>MessageReceivedEvent</td>
<td>Signalling event which can be waited on to determine when a message is queued for reading. Note that there is no guarantee that after the event is signaled the blocked thread will find the message in the queue. Other user created threads could be preempted and dequeue the message before the waiting thread wakes up. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Port</td>
<td>Gets the port number this NetPeer is listening and sending on, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>ServerConnection</td>
<td>Gets the connection to the server, if any</td>
</tr>
<tr>
<td>Socket</td>
<td>Gets the socket, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Statistics</td>
<td>Statistics on this NetPeer since it was initialized (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Status</td>
<td>Gets the NetPeerStatus of the NetPeer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Tag</strong></td>
<td>Gets or sets the application defined object containing data about the peer. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>UniqueIdentifier</strong></td>
<td>Gets a unique identifier for this NetPeer based on Mac address and ip/port. Note! Not available until Start() has been called! (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td><strong>UPnP</strong></td>
<td>Returns an UPnP object if enabled in the NetPeerConfiguration. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetClient Class
Lidgren.Network Namespace
NetClient constructor

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
(2012.1.7.0)
### Syntax

**C#**

```csharp
public NetClient(
    NetPeerConfiguration config
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    config As NetPeerConfiguration _
)
```

**Visual C++**

```cpp
public:
NetClient(
    NetPeerConfiguration^ config
)
```

### Parameters

config

Type: Lidgren.Network..NetPeerConfiguration

See Also

NetClient Class
Lidgren.Network Namespace
The `NetClient` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="Method" alt="Connect(IPEndPoint)" /></td>
<td>Create a connection to a remote endpoint. (Inherited from NetPeer.) Connect to a remote server. (Overrides NetPeer:::Con NetOutgoingMessage.)</td>
</tr>
<tr>
<td><img src="Method" alt="Connect(IPEndPoint, NetOutgoingMessage)" /></td>
<td>Connect to a remote server. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="Connect(String, Int32)" /></td>
<td>Create a connection to a remote endpoint. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="Connect(String, Int32, NetOutgoingMessage)" /></td>
<td>Create a connection to a remote endpoint. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="CreateMessage()" /></td>
<td>Creates a new message for sending. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="CreateMessage(Int32)" /></td>
<td>Creates a new message for sending. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="CreateMessage(String)" /></td>
<td>Creates a new message for sending and writes the provided string to it. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="Disconnect" /></td>
<td>Disconnect from server.</td>
</tr>
<tr>
<td><img src="Method" alt="DiscoverKnownPeer(IPEndPoint)" /></td>
<td>Emit a discovery signal to a known host. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="DiscoverKnownPeer(String, Int32)" /></td>
<td>Emit a discovery signal to a known host. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="DiscoverLocalPeers" /></td>
<td>Emit a discovery signal to all hosts on your subnet. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><img src="Method" alt="Equals" /></td>
<td>Determines whether the specified object is equal to the current Object. (Inherited from Object.) Allows an Object to attempt and perform other cleanup operations.</td>
</tr>
</tbody>
</table>
Finalize

FlushSendQueue

GetConnection

GetHashCode

GetType

Introduce

MemberwiseClone

RawSend

ReadMessage

ReadMessages

Recycle(NetIncomingMessage)

Recycle(IEnumerable<NetIncomingMessage>)

Recycle(IEnumerable<NetIncomingMessage>)

The Object is reclaimed by the garbage collector (Inherited from Object.)

If NetPeerConfiguration.AutoFlushSendQueue() is false; you need to call this function to send all queued messages using SendMessage() (Inherited from NetPeer.)

Get the connection, if any, for a remote endpoint (Inherited from NetPeer.)

Serves as a hash function for the type (Inherited from Object.)

Gets the Type of the current instance (Inherited from Object.)

Send NetIntroduction to hostExternal and clientExternal; introducing client to host (Inherited from NetPeer.)

Creates a shallow copy of the current object (Inherited from Object.)

Send raw bytes; only used for debugging (Inherited from NetPeer.)

Read a pending message from the connection, if any (Inherited from NetPeer.)

Read a pending message from the connection, if any (Inherited from NetPeer.)

Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)

Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)

Call this to register a callba
- **RegisterReceivedCallback**
  
  message arrives
  (Inherited from **NetPeer**.)

- **SendDiscoveryResponse**
  
  Send a discovery response message
  (Inherited from **NetPeer**.)

- **SendMessage(NetOutgoingMessage, NetDeliveryMethod)**
  
  Sends message to server

- **SendMessage(NetOutgoingMessage, NetDeliveryMethod, Int32)**
  
  Sends message to server

- **SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)**
  
  Send a message to a specific connection
  (Inherited from **NetPeer**.)

- **SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)**
  
  Send a message to a specific connection
  (Inherited from **NetPeer**.)

- **SendMessage(NetOutgoingMessage, List<Of('<(NetConnection>>)>, NetDeliveryMethod, Int32)**
  
  Send a message to a list of connections
  (Inherited from **NetPeer**.)

- **SendUnconnectedMessage(NetOutgoingMessage, IList<Of('<(IPEndPoint>>)>)**
  
  Send a message to an unconnected host
  (Inherited from **NetPeer**.)

- **SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)**
  
  Send a message to an unconnected host
  (Inherited from **NetPeer**.)

- **SendUnconnectedMessage(NetOutgoingMessage, String, Int32)**
  
  Send a message to this exact unconnected host
  (loopback)
  (Inherited from **NetPeer**.)

- **SendUnconnectedToSelf**
  
  Send a message to this exact same netpeer
  (loopback)
  (Inherited from **NetPeer**.)

- **Shutdown**
  
  Disconnects all active connections and closes the socket
  (Inherited from **NetPeer**.)

- **Start**
  
  Binds to socket and spawns the networking thread
  (Inherited from **NetPeer**.)

- **ToString**
  
  Returns a string that represents this object
  (Overrides **Object**:::**ToString**)

- **WaitMessage**
  
  Reads a pending message from a connection, blocking up to maxMillis
  (Inherited from **NetPeer**.)
See Also

NetClient Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetClient:::Connect Method

NetClient Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect(IPEndPoint)</td>
<td>Create a connection to a remote endpoint (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(IPEndPoint,</td>
<td>Connect to a remote server (Overrides NetPeer:::.Connect(IPEndPoint,</td>
</tr>
<tr>
<td>Connect(String, Int32)</td>
<td>Create a connection to a remote endpoint (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(String, Int32,</td>
<td>Create a connection to a remote endpoint (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>NetOutgoingMessage)</td>
<td></td>
</tr>
</tbody>
</table>
See Also

NetClient Class
NetClient Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetClient..::..Connect Method (IPEndPoint, NetOutgoingMessage)

NetClient Class See Also Send Feedback

Connect to a remote server

Namespace: Lidgren.Network
Syntax

C#

```csharp
public override NetConnection Connect(
    IPEndPoint remoteEndpoint,
    NetOutgoingMessage hailMessage
)
```

Visual Basic

```vbnet
Public Overrides Function Connect ( _
    remoteEndpoint As IPEndPoint, _
    hailMessage As NetOutgoingMessage _
) As NetConnection
```

Visual C++

```cpp
public:
virtual NetConnection^ Connect(
    IPEndPoint^ remoteEndpoint,
    NetOutgoingMessage^ hailMessage
) override
```

Parameters

remoteEndpoint

Type: `System.Net.IPEndPoint`  
The remote endpoint to connect to

hailMessage

Type: `Lidgren.Network.NetOutgoingMessage`  
The hail message to pass

Return Value

server connection, or null if already connected
See Also

NetClient Class
Connect Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetClient...::CreateMessage Method

NetClient Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateMessage()()()()</td>
<td>Creates a new message for sending (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(Int32)</td>
<td>Creates a new message for sending (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(String)</td>
<td>Creates a new message for sending and writes the provided string to it (Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetClient Class
NetClient Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetClient Disconnect Method

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public void Disconnect(
    string byeMessage
)
```

**Visual Basic**

```vbnet
Public Sub Disconnect (_
    byeMessage As String _
)
```

**Visual C++**

```cpp
public:
void Disconnect(
    String^ byeMessage
)
```

**Parameters**

`byeMessage`

Type: `System::::String`

reason for disconnect
See Also

NetClient Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetClient...DiscoverKnownPeer Method

NetClient Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiscoverKnownPeer(IPEndPoint)</td>
<td>Emit a discovery signal to a single known host</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetPeer.)</td>
</tr>
<tr>
<td>DiscoverKnownPeer(String, Int32)</td>
<td>Emit a discovery signal to a single known host</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetClient Class
NetClient Members
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetClient:::Recycle Method
NetClient Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recycle(NetIncomingMessage)</td>
<td>Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Recycle(IEnumerable&lt;Of &lt;&lt;(NetIncomingMessage)&gt;&gt;))</td>
<td>Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetClient Class
NetClient Members
Lidgren.Network Namespace
C#  Visual Basic
   Visual C++
   Include Protected Members
   Include Inherited Members

Lidgren Network Library documentation
NetClient...:SendMessage Method

NetClient Class  See Also  Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetDeliveryMethod)</code></td>
<td>Sends message to server</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetDeliveryMethod, Int32)</code></td>
<td>Sends message to server</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)</code></td>
<td>Send a message to a specific connection (Inherited from <code>NetPeer</code>.)</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)</code></td>
<td>Send a message to a specific connection (Inherited from <code>NetPeer</code>.)</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, List&lt;Of &lt;=(NetConnection)&gt;), NetDeliveryMethod, Int32)</code></td>
<td>Send a message to a list of connections (Inherited from <code>NetPeer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetClient Class
NetClient Members
Lidgren.Network Namespace
C# Visual Basic Visual C++

Lidgren Network Library documentation

NetClient..:.:SendMessage Method (NetOutgoingMessage, NetDeliveryMethod)

NetClient Class See Also Send Feedback

Sends message to server

Namespace: Lidgren.Network
### Syntax

**C#**

```csharp
public NetSendResult SendMessage(
    NetOutgoingMessage msg,
    NetDeliveryMethod method
)
```

**Visual Basic**

```vbnet
Public Function SendMessage (_
    msg As NetOutgoingMessage, _
    method As NetDeliveryMethod _
) As NetSendResult
```

**Visual C++**

```cpp
public:
NetSendResult SendMessage(
    NetOutgoingMessage^ msg,
    NetDeliveryMethod method
)
```

### Parameters

**msg**

Type: `Lidgren.Network.NetOutgoingMessage`


**method**

Type: `Lidgren.Network.NetDeliveryMethod`


### Return Value
See Also

NetClient Class
SendMessage Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetClient.SendMessage Method (NetOutgoingMessage, NetDeliveryMethod, Int32)

Sends message to server

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetSendResult SendMessage(
    NetOutgoingMessage msg,
    NetDeliveryMethod method,
    int sequenceChannel
)

Visual Basic

Public Function SendMessage (_
    msg As NetOutgoingMessage, _
    method As NetDeliveryMethod, _
    sequenceChannel As Integer _
) As NetSendResult

Visual C++

public:
NetSendResult SendMessage(
    NetOutgoingMessage^ msg,
    NetDeliveryMethod method,
    int sequenceChannel
)

Parameters

msg
  Type: Lidgren.Network.NetOutgoingMessage

[Missing <param name="msg"/> documentation for

method
  Type: Lidgren.Network.NetDeliveryMethod

[Missing <param name="method"/> documentation for
sequenceChannel
Type: System::..Int32

[Missing <param name="sequenceChannel"/> documentation for

Return Value

[Missing <returns> documentation for
See Also

NetClient Class
SendMessage Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetClient::SendUnconnectedMessage Method

NetClient Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SendUnconnectedMessage(NetOutgoingMessage, IList&lt;Of &lt;&lt;(IPEndPoint)&gt;&gt;))</td>
<td>Send a message to an unconnected host (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)</td>
<td>Send a message to an unconnected host (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>SendUnconnectedMessage(NetOutgoingMessage, String, Int32)</td>
<td>Send a message to an unconnected host (Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetClient Class
NetClient Members
Lidgren.Network Namespace
Returns a string that represents this object

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public override string ToString()

Visual Basic

Public Overrides Function ToString As String

Visual C++

public:
virtual String^ ToString() override

Return Value

See Also

NetClient Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetClient Properties

The `NetClient` type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration</strong></td>
<td>Gets the configuration used to instanciate this NetPeer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Connections</strong></td>
<td>Gets a copy of the list of connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>ConnectionsCount</strong></td>
<td>Gets the number of active connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>ConnectionStatus</strong></td>
<td>Gets the connection status of the server connection (or NetConnectionStatus.Disconnected if no connection)</td>
</tr>
<tr>
<td><strong>MessageReceivedEvent</strong></td>
<td>Signalling event which can be waited on to determine when a message is queued for reading. Note that there is no guarantee that after the event is signaled the blocked thread will find the message in the queue. Other user created threads could be preempted and dequeue the message before the waiting thread wakes up. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Gets the port number this NetPeer is listening and sending on, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>ServerConnection</strong></td>
<td>Gets the connection to the server, if any (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Socket</strong></td>
<td>Gets the socket, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Statistics</strong></td>
<td>Statistics on this NetPeer since it was initialized (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Gets the NetPeerStatus of the NetPeer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Tag</strong></td>
<td>Gets or sets the application defined object containing data about the peer (Inherited from <a href="#">NetPeer</a>).</td>
</tr>
<tr>
<td><strong>UniqueIdentifier</strong></td>
<td>Gets a unique identifier for this NetPeer based on Mac address and ip/port. Note! Not available until Start() has been called! (Inherited from <a href="#">NetPeer</a>).</td>
</tr>
<tr>
<td><strong>UPnP</strong></td>
<td>Returns an UPnP object if enabled in the NetPeerConfiguration (Inherited from <a href="#">NetPeer</a>).</td>
</tr>
</tbody>
</table>
See Also

NetClient Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetClient:::ConnectionStatus Property

NetClient Class  See Also  Send Feedback

Gets the connection status of the server connection (or NetConnectionStatus.Disconnected if no connection)

Namespace: Lidgren.Network
Syntax

C#

public NetConnectionStatus ConnectionStatus { get; }

Visual Basic

Public ReadOnly Property ConnectionStatus As NetConnectionStatus
Get

Visual C++

public:
property NetConnectionStatus ConnectionStatus {
    NetConnectionStatus get ();
}
See Also

NetClient Class
Lidgren.Network Namespace
Gets the connection to the server, if any

**Namespace:** [Lidgren.Network](https://www.lidgren-network.com)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
# Syntax

**C#**

```csharp
public NetConnection ServerConnection { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property ServerConnection As NetConnection
    Get
```

**Visual C++**

```cpp
public:
    property NetConnection^ ServerConnection { NetConnection^ get (); }
```
See Also

NetClient Class
Lidgren.Network Namespace
Represents a connection to a remote peer

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public class NetConnection

Visual Basic

Public Class NetConnection

Visual C++

public ref class NetConnection
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetConnection
See Also

NetConnection Members
Lidgren.Network Namespace
The `NetConnection` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Approve()</strong></td>
<td>Approves this connection; sending a connection response to the remote host</td>
</tr>
<tr>
<td><strong>Approve(NetOutgoingMessage)</strong></td>
<td>Approves this connection; sending a connection response to the remote host</td>
</tr>
<tr>
<td><strong>Deny()</strong></td>
<td>Denies this connection; disconnecting it</td>
</tr>
<tr>
<td><strong>Deny(String)</strong></td>
<td>Denies this connection; disconnecting it</td>
</tr>
<tr>
<td><strong>Disconnect</strong></td>
<td>Disconnect from the remote peer</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetLocalTime</strong></td>
<td>Gets local time value comparable to NetTime.Now from a remote value</td>
</tr>
<tr>
<td><strong>GetRemoteTime</strong></td>
<td>Gets the remote time value for a local time value produced by NetTime.Now</td>
</tr>
<tr>
<td></td>
<td>Zero windowSize indicates that the channel is not yet instantiated (used)</td>
</tr>
<tr>
<td></td>
<td>Negative freeWindowSlots means this amount of messages are currently queued</td>
</tr>
<tr>
<td></td>
<td>but delayed due to closed window.</td>
</tr>
</tbody>
</table>
- **Get Type**
  Gets the **Type** of the current instance.
  (Inherited from **Object**.)

- **MemberwiseClone**
  Creates a shallow copy of the current **Object**.
  (Inherited from **Object**.)

- **SendMessage**
  Sends a message to this remote connection.

- **ToString**
  Returns a string that represents this object.
  (Overrides **Object**... **ToString()**.)
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AverageRoundtripTime</td>
<td>Gets the current average roundtrip time in seconds</td>
</tr>
<tr>
<td>LocalHailMessage</td>
<td>Gets the local hail message that was sent as part of the handshake</td>
</tr>
<tr>
<td>Peer</td>
<td>Gets the peer which holds this connection</td>
</tr>
<tr>
<td>RemoteEndpoint</td>
<td>Gets the remote endpoint for the connection</td>
</tr>
<tr>
<td>RemoteHailMessage</td>
<td>The message that the remote part specified via Connect() or Approve() - can be null.</td>
</tr>
<tr>
<td>RemoteTimeOffset</td>
<td>Time offset between this peer and the remote peer</td>
</tr>
<tr>
<td>RemoteUniqueIdentifier</td>
<td>Gets the unique identifier of the remote NetPeer for this connection</td>
</tr>
<tr>
<td>Statistics</td>
<td>Gets various statistics for this connection</td>
</tr>
<tr>
<td>Status</td>
<td>Gets the current status of the connection (synced to the last status message read)</td>
</tr>
<tr>
<td>Tag</td>
<td>Gets or sets the application defined object containing data about the connection</td>
</tr>
</tbody>
</table>
See Also

NetConnection Class
Lidgren.Network Namespace
The `NetConnection` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approve()()()</td>
<td>Approves this connection; sending a connection response to the remote host.</td>
</tr>
<tr>
<td>Approve(NetOutgoingMessage)</td>
<td>Approves this connection; sending a connection response to the remote host.</td>
</tr>
<tr>
<td>Deny()()()()</td>
<td>Denies this connection; disconnecting it.</td>
</tr>
<tr>
<td>Deny(String)</td>
<td>Denies this connection; disconnecting it.</td>
</tr>
<tr>
<td>Disconnect</td>
<td>Disconnect from the remote peer.</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetLocalTime</td>
<td>Gets local time value comparable to NetTime.Now from a remote value.</td>
</tr>
<tr>
<td>GetRemoteTime</td>
<td>Gets the remote time value for a local time value produced by NetTime.Now. Zero windowSize indicates that the channel is not yet instantiated (used) Negative freeWindowSlots means this amount of messages are currently queued but delayed due to closed window</td>
</tr>
</tbody>
</table>
- **GetType**: Gets the **Type** of the current instance. (Inherited from **Object**.)

- **MemberwiseClone**: Creates a shallow copy of the current **Object**. (Inherited from **Object**.)

- **SendMessage**: Sends a message to this remote connection

- **ToString**: Returns a string that represents this object (Overrides **Object**.::.ToString()).
See Also

NetConnection Class
Lidgren.Network Namespace
C#  Visual Basic  Visual C++  Include Protected Members  Include Inherited Members
Lidgren Network Library documentation
NetConnection...::Approve Method
NetConnection Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approve()()()()</td>
<td>Approves this connection; sending a connection response to the remote host</td>
</tr>
<tr>
<td>Approve(NetOutgoingMessage)</td>
<td>Approves this connection; sending a connection response to the remote host</td>
</tr>
</tbody>
</table>
See Also

NetConnection Class
NetConnection Members
Lidgren.Network Namespace
Approves this connection; sending a connection response to the remote host

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Approve()

Visual Basic

Public Sub Approve

Visual C++

public:
void Approve()
See Also

NetConnection Class
Approve Overload
Lidgren.Network Namespace
Approves this connection; sending a connection response to the remote host

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void Approve(
    NetOutgoingMessage localHail
)
```

Visual Basic

```vbnet
Public Sub Approve ( _
    localHail As NetOutgoingMessage _
)
```

Visual C++

```cpp
public:
void Approve(
    NetOutgoingMessage^ localHail
)
```

Parameters

localHail

Type: Lidgren.NetworkNetMessage
The local hail message that will be set as RemoteHailMessage on the remote host
See Also

NetConnection Class
Approve Overload
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deny()</td>
<td>Denies this connection; disconnecting it</td>
</tr>
<tr>
<td>Deny(String)</td>
<td>Denies this connection; disconnecting it</td>
</tr>
</tbody>
</table>
See Also

NetConnection Class
NetConnection Members
Lidgren.Network Namespace
Denies this connection; disconnecting it

Namespace: Lidgren.Network
Syntax

C#

public void Deny()

Visual Basic

Public Sub Deny

Visual C++

public:
    void Deny()
See Also

NetConnection Class
Deny Overload
Lidgren.Network Namespace
Denies this connection; disconnecting it

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Deny(
    string reason
)

Visual Basic

Public Sub Deny ( _
    reason As String _
)

Visual C++

public:
void Deny(
    String^ reason
)

Parameters

reason
Type: System::String
The stated reason for the disconnect, readable as a string in the StatusChanged message on the remote host
See Also

NetConnection Class
Deny Overload
Lidgren.Network Namespace
Disconnect from the remote peer

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Disconnect(
    string byeMessage
)

Visual Basic

Public Sub Disconnect ( _
    byeMessage As String _
)

Visual C++

public:
void Disconnect(
    String^ byeMessage
)

Parameters

byeMessage
    Type: System::::String
    the message to send with the disconnect message
See Also

NetConnection Class
Lidgren.Network Namespace
GetLocalTime Method

NetConnection Class See Also  Send Feedback

Gets local time value comparable to NetTime.Now from a remote value

Namespace: Lidgren.Network
Syntax

C#

public double GetLocalTime(
    double remoteTimestamp
)

Visual Basic

Public Function GetLocalTime (_
    remoteTimestamp As Double _
) As Double

Visual C++

public:
    double GetLocalTime(
        double remoteTimestamp
    )

Parameters

remoteTimestamp
Type: System::Double

[Missing <param name="remoteTimestamp"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetConnection Class
Lidgren.Network Namespace
GetRemoteTime Method

Gets the remote time value for a local time value produced by NetTime.Now

Namespace: Lidgren.Network
Syntax

C#

public double GetRemoteTime(
    double localTimestamp
)

Visual Basic

Public Function GetRemoteTime (_
    localTimestamp As Double _
) As Double

Visual C++

public:
    double GetRemoteTime(
        double localTimestamp
    )

Parameters

localTimestamp
    Type: System::Double

    [Missing <param name="localTimestamp"/> documentation for
    ]

Return Value

    [Missing <returns> documentation for
    ]
See Also

NetConnection Class
Lidgren.Network Namespace
Zero windowSize indicates that the channel is not yet instantiated (used)
Negative freeWindowSlots means this amount of messages are currently queued
but delayed due to closed window

Namespace: Lidgren.Network
(2012.1.7.0)
Syntax

C#

```csharp
public void GetSendQueueInfo(
    NetDeliveryMethod method,
    int sequenceChannel,
    out int windowSize,
    out int freeWindowSlots
)
```

Visual Basic

```vbnet
Public Sub GetSendQueueInfo ( _
    method As NetDeliveryMethod, _
    sequenceChannel As Integer, _
    <OutAttribute> ByRef windowSize As Integer, _
    <OutAttribute> ByRef freeWindowSlots As Integer _
)
```

Visual C++

```cpp
public:
void GetSendQueueInfo(
    NetDeliveryMethod method,
    int sequenceChannel,
    [OutAttribute] int% windowSize,
    [OutAttribute] int% freeWindowSlots
)
```

Parameters

method
Type: Lidgren.Network..::..NetDeliveryMethod

[Missing <param name="method"/> documentation for

sequenceChannel
Type: System..::..Int32
windowSize
Type: `System.Int32`

freeWindowSlots
Type: `System.Int32`
See Also

NetConnection Class
Lidgren.Network Namespace
Send a message to this remote connection

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetSendResult SendMessage(
    NetOutgoingMessage msg,
    NetDeliveryMethod method,
    int sequenceChannel
)

Visual Basic

Public Function SendMessage ( 
    msg As NetOutgoingMessage, _
    method As NetDeliveryMethod, _
    sequenceChannel As Integer _
) As NetSendResult

Visual C++

public:
NetSendResult SendMessage(
    NetOutgoingMessage^ msg,
    NetDeliveryMethod method,
    int sequenceChannel
)

Parameters

msg
    Type: Lidgren.Network::..::NetOutgoingMessage
    The message to send

method
    Type: Lidgren.Network::..::NetDeliveryMethod
    How to deliver the message

sequenceChannel
    Type: System::..::Int32
Sequence channel within the delivery method

Return Value

See Also

NetConnection Class
Lidgren.Network Namespace
Returns a string that represents this object

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public override string ToString()
```

**Visual Basic**

```vbnet
PublicOverrides Function ToString As String
```

**Visual C++**

```cpp
public:
virtual String^ ToString() override
```

**Return Value**

See Also

NetConnection Class
Lidgren.Network Namespace
The `NetConnection` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AverageRoundtripTime</td>
<td>Gets the current average roundtrip time in seconds</td>
</tr>
<tr>
<td>LocalHailMessage</td>
<td>Gets the local hail message that was sent as part of the handshake</td>
</tr>
<tr>
<td>Peer</td>
<td>Gets the peer which holds this connection</td>
</tr>
<tr>
<td>RemoteEndpoint</td>
<td>Gets the remote endpoint for the connection</td>
</tr>
<tr>
<td>RemoteHailMessage</td>
<td>The message that the remote part specified via Connect() or Approve() - can be null.</td>
</tr>
<tr>
<td>RemoteTimeOffset</td>
<td>Time offset between this peer and the remote peer</td>
</tr>
<tr>
<td>RemoteUniqueIdentifier</td>
<td>Gets the unique identifier of the remote NetPeer for this connection</td>
</tr>
<tr>
<td>Statistics</td>
<td>Gets various statistics for this connection</td>
</tr>
<tr>
<td>Status</td>
<td>Gets the current status of the connection</td>
</tr>
<tr>
<td>(synced to the last status message read)</td>
<td></td>
</tr>
<tr>
<td>Tag</td>
<td>Gets or sets the application defined object containing data about the connection</td>
</tr>
</tbody>
</table>
See Also

NetConnection Class
Lidgren.Network Namespace
NetConnection.AverageRoundtripTime Property

Gets the current average roundtrip time in seconds

**Namespace:** [Lidgren.Network](Lidgren.Network)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public float AverageRoundtripTime { get; }

Visual Basic

Public ReadOnly Property AverageRoundtripTime As Single
Get

Visual C++

public:
property float AverageRoundtripTime {
    float get ();
}

See Also

NetConnection Class
Lidgren.Network Namespace
Gets the local hail message that was sent as part of the handshake

**Namespace:**  [Lidgren.Network](#)  
**Assembly:**  Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public NetOutgoingMessage LocalHailMessage { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property LocalHailMessage As NetOutgoingMessage
Get
```

**Visual C++**

```cpp
public:
property NetOutgoingMessage^ LocalHailMessage { 
    NetOutgoingMessage^ get ();
}
See Also

NetConnection Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetConnection::Peer Property

NetConnection Class See Also Send Feedback

Gets the peer which holds this connection

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public NetPeer Peer { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property Peer As NetPeer
    Get
```

**Visual C++**

```cpp
public:
    property NetPeer^ Peer {
        NetPeer^ get ();
    }
```
See Also

NetConnection Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetConnection...:.RemoteEndpoint Property

NetConnection Class  See Also  Send Feedback

Gets the remote endpoint for the connection

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public IPEndPoint RemoteEndpoint { get; }

Visual Basic

Public ReadOnly Property RemoteEndpoint As IPEndPoint
Get

Visual C++

public:
property IPEndPoint^ RemoteEndpoint {
    IPEndPoint^ get ();
}
See Also

NetConnection Class
Lidgren.Network Namespace
The message that the remote part specified via Connect() or Approve() - can be null.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetIncomingMessage RemoteHailMessage { get; }

Visual Basic

Public ReadOnly Property RemoteHailMessage As NetIncomingMessage
Get

Visual C++

public:
property NetIncomingMessage^ RemoteHailMessage {
    NetIncomingMessage^ get ();
}
See Also

NetConnection Class
Lidgren.Network Namespace
NetConnection..::.RemoteTimeOffset Property

NetConnection Class

See Also

Send Feedback

Time offset between this peer and the remote peer

Namespace: Lidgren.Network

### Syntax

**C#**

```csharp
public float RemoteTimeOffset { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property RemoteTimeOffset As Single
Get
```

**Visual C++**

```cpp
public:
property float RemoteTimeOffset {
    float get ()
}
```
See Also

NetConnection Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetConnection.RemoteUniqueIdentifier Property

NetConnection Class See Also Send Feedback

Gets the unique identifier of the remote NetPeer for this connection

Namespace: Lidgren.Network
Syntax

C#

public long RemoteUniqueIdentifier { get; }

Visual Basic

Public ReadOnly Property RemoteUniqueIdentifier As Long
    Get

Visual C++

public:
    property long long RemoteUniqueIdentifier {
        long long get ();
    }
}
See Also

NetConnection Class
Lidgren.Network Namespace
Gets various statistics for this connection

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public NetConnectionStatistics Statistics { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property Statistics As NetConnectionStatistics
Get
```

**Visual C++**

```cpp
public:
property NetConnectionStatistics^ Statistics { NetConnectionStatistics^ get (); }
```
See Also

NetConnection Class
Lidgren.Network Namespace
Gets the current status of the connection (synced to the last status message read)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetConnectionStatus Status { get; }

Visual Basic

Public ReadOnly Property Status As NetConnectionStatus
Get

Visual C++

public:
property NetConnectionStatus Status {
    NetConnectionStatus get ();
}
See Also

NetConnection Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetConnection..:..Tag Property

NetConnection Class  See Also  Send Feedback

Gets or sets the application defined object containing data about the connection

Namespace: Lidgren.Network
Syntax

C#

public Object Tag { get; set; }

Visual Basic

Public Property Tag As Object
    Get
    Set

Visual C++

public:
property Object^ Tag {
    Object^ get ();
    void set (Object^ value);
}
See Also

NetConnection Class
Lidgren.Network Namespace
Statistics for a NetConnection instance

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public sealed class NetConnectionStatistics

Visual Basic

Public NotInheritable Class NetConnectionStatistics

Visual C++

public ref class NetConnectionStatistics sealed
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetConnectionStatistics
See Also

NetConnectionStatistics Members
Lidgren.Network Namespace
The `NetConnectionStatistics` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a string that represents this object (Overrides Object::ToString().)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReceivedBytes</td>
<td>Gets the number of received bytes for this connection</td>
</tr>
<tr>
<td>ReceivedPackets</td>
<td>Gets the number of received packets for this connection</td>
</tr>
<tr>
<td>ResentMessages</td>
<td>Gets the number of resent reliable messages for this connection</td>
</tr>
<tr>
<td>SentBytes</td>
<td>Gets the number of sent bytes for this connection</td>
</tr>
<tr>
<td>SentPackets</td>
<td>Gets the number of sent packets for this connection</td>
</tr>
</tbody>
</table>
See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
The **NetConnectionStatistics** type exposes the following members.
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <code>Object</code> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a string that represents this object</td>
</tr>
<tr>
<td></td>
<td>(Overrides <code>Object::ToString</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetConnectionStatistics...::ToString Method

NetConnectionStatistics Class See Also Send Feedback

Returns a string that represents this object

Namespace: Lidgren.Network
**Syntax**

**C#**

public override string ToString()

**Visual Basic**

Public Overrides Function ToString As String

**Visual C++**

public:
virtual String^ ToString() override

**Return Value**

See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
The `NetConnectionStatistics` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReceivedBytes</strong></td>
<td>Gets the number of received bytes for this connection</td>
</tr>
<tr>
<td><strong>ReceivedPackets</strong></td>
<td>Gets the number of received packets for this connection</td>
</tr>
<tr>
<td><strong>ResentMessages</strong></td>
<td>Gets the number of resent reliable messages for this connection</td>
</tr>
<tr>
<td><strong>SentBytes</strong></td>
<td>Gets the number of sent bytes for this connection</td>
</tr>
<tr>
<td><strong>SentPackets</strong></td>
<td>Gets the number of sent packets for this connection</td>
</tr>
</tbody>
</table>
See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetConnectionStatistics..:..ReceivedBytes Property

NetConnectionStatistics Class See Also Send Feedback

Gets the number of received bytes for this connection

Namespace: Lidgren.Network
Syntax

C#

public int ReceivedBytes { get; }

Visual Basic

Public ReadOnly Property ReceivedBytes As Integer
    Get
        'Visual C++

public:
    property int ReceivedBytes {
        int get ();
    }

See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetConnectionStatistics..:.:ReceivedPackets Property
NetConnectionStatistics Class See Also Send Feedback

Gets the number of received packets for this connection

Namespace: Lidgren.Network
### Syntax

**C#**

```csharp
public int ReceivedPackets { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property ReceivedPackets As Integer
    Get
```

**Visual C++**

```cpp
public:
    property int ReceivedPackets {
        int get ();
    }
```
See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
NetConnectionStatistics ResentMessages Property

Gets the number of resent reliable messages for this connection

Namespace: Lidgren.Network
Syntax

C#

public int ResentMessages { get; }

Visual Basic

Public ReadOnly Property ResentMessages As Integer
    Get

Visual C++

public:
property int ResentMessages {
    int get ();
}
}
See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetConnectionStatistics::SentBytes Property

NetConnectionStatistics Class See Also Send Feedback

 Gets the number of sent bytes for this connection

Namespace: Lidgren.Network
Syntax

C#

public int SentBytes { get; }

Visual Basic

Public ReadOnly Property SentBytes As Integer
Get

Visual C++

public:
property int SentBytes {
    int get ()
}
See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetConnectionStatistics::SentPackets Property

Gets the number of sent packets for this connection

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int SentPackets { get; }  

Visual Basic

Public ReadOnly Property SentPackets As Integer  
Get

Visual C++

public:
property int SentPackets {
    int get ();
}


See Also

NetConnectionStatistics Class
Lidgren.Network Namespace
Status for a NetConnection instance

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
(2012.1.7.0)
Syntax

**C#**

```csharp
public enum NetConnectionStatus
```

**Visual Basic**

```vbnet
Public Enumeration NetConnectionStatus
```

**Visual C++**

```cpp
public enum class NetConnectionStatus
```
### Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
<tr>
<td>InitiatedConnect</td>
<td></td>
</tr>
<tr>
<td>RespondedAwaitingApproval</td>
<td></td>
</tr>
<tr>
<td>RespondedConnect</td>
<td></td>
</tr>
<tr>
<td>Connected</td>
<td></td>
</tr>
<tr>
<td>Disconnecting</td>
<td></td>
</tr>
<tr>
<td>Disconnected</td>
<td></td>
</tr>
</tbody>
</table>
See Also

[Lidgren.Network Namespace]
How the library deals with resends and handling of late messages

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public enum NetDeliveryMethod
```

**Visual Basic**

```vbnet
Public Enumeration NetDeliveryMethod
```

**Visual C++**

```cpp
public enum class NetDeliveryMethod
```
<table>
<thead>
<tr>
<th>Member name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td>Unreliable</td>
<td></td>
</tr>
<tr>
<td>UnreliableSequenced</td>
<td></td>
</tr>
<tr>
<td>ReliableUnordered</td>
<td></td>
</tr>
<tr>
<td>ReliableSequenced</td>
<td></td>
</tr>
<tr>
<td>ReliableOrdered</td>
<td></td>
</tr>
</tbody>
</table>
See Also

Lidgren.Network Namespace
Namespace: Lidgren.Network
Syntax

C#

public class NetDESEncryption : INetEncryption

Visual Basic

Public Class NetDESEncryption
    Implements INetEncryption

Visual C++

public ref class NetDESEncryption : INetEncryption
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetDESEncryption
See Also

NetDES Encryption Members
Lidgren.Network Namespace
The `NetDESEncryption` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetDESEncryption(String)</td>
<td>NetDESEncryption constructor</td>
</tr>
<tr>
<td>NetDESEncryption(array&lt;Byte&gt;[][], array&lt;Byte&gt;[][])</td>
<td>NetDESEncryption constructor</td>
</tr>
<tr>
<td>NetDESEncryption(String, Int32)</td>
<td>NetDESEncryption constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>)</td>
</tr>
</tbody>
</table>
See Also

NetDESEncryption Class
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetDESEncryption Constructor

NetDESEncryption Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetDESEncryption(String)</td>
<td>NetDESEncryption constructor</td>
</tr>
<tr>
<td>NetDESEncryption(array&lt;Byte&gt;[], array&lt;Byte&gt;[])</td>
<td>NetDESEncryption constructor</td>
</tr>
<tr>
<td>NetDESEncryption(String, Int32)</td>
<td>NetDESEncryption constructor</td>
</tr>
</tbody>
</table>
See Also

NetDESEncryption Class
NetDESEncryption Members
Lidgren.Network Namespace
NetDESEncryption Constructor (String)

Namespace: Lidgren.Network
Syntax

C#

```csharp
public NetDESEncryption(
    string key
)
```

Visual Basic

```vbnet
Public Sub New (_
    key As String _
)
```

Visual C++

```cpp
public:
NetDESEncryption(
    String^ key
)
```

Parameters

key

Type: System.String

See Also

NetDESEncryption Class
NetDESEncryption Overload
Lidgren.Network Namespace
NetDESEncryption Constructor (array<Byte>[][], array<Byte>[][])
## Syntax

### C#

```csharp
public NetDESEncryption(
    byte[] key,
    byte[] iv
)
```

### Visual Basic

```vbnet
Public Sub New (_
    key As Byte(), _
    iv As Byte() _
)
```

### Visual C++

```cpp
public:
NetDESEncryption(
    array<unsigned_char>^ key,
    array<unsigned_char>^ iv
)
```

## Parameters

**key**

Type: array`<System::::::Byte>[]()[[]`


**iv**

Type: array`<System::::::Byte>[]()[[]`

See Also

NetDESEncryption Class
NetDESEncryption Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetDESEncryption Constructor (String, Int32)

**NetDESEncryption Class**

**See Also**

**Send Feedback**

NetDESEncryption constructor

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public NetDESEncryption(
    string key,
    int bitsize
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    key As String, _
    bitsize As Integer _
)
```

**Visual C++**

```cpp
public:
NetDESEncryption(
    String^ key,
    int bitsize
)
```

### Parameters

**key**

Type: `System::String`


**bitsize**

Type: `System::Int32`

See Also

NetDESEncryption Class
NetDESEncryption Overload
Lidgren.Network Namespace
The **NetDESEncryption** type exposes the following members.
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetDESEncryption Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetDESEncryption...:::...Decrypt Method

**NetDESEncryption Class**  [See Also](#)  [Send Feedback](#)

Decrypt incoming message

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool Decrypt(
    NetIncomingMessage msg
)

Visual Basic

Public Function Decrypt (_
    msg As NetIncomingMessage _
) As Boolean

Visual C++

public: bool Decrypt(
    NetIncomingMessage^ msg
) sealed

Parameters

msg
Type: Lidgren.Network.NetIncomingMessage


Return Value


Implements

INetEncryption.NetDESEncryption.Decrypt(NetIncomingMessage)
See Also

NetDES Encryption Class
Lidgren.Network Namespace
Encrypt outgoing message

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool Encrypt(
    NetOutgoingMessage msg
)
```

Visual Basic

```vbnet
Public Function Encrypt ( _
    msg As NetOutgoingMessage _
) As Boolean
```

Visual C++

```cpp
public:
    virtual bool Encrypt( _
        NetOutgoingMessage^ msg
    ) sealed
```

Parameters

msg


Return Value


Implements

INetEncryption.NetDESEncryption.Encrypt(NetOutgoingMessage)
See Also

NetDESEncryption Class
Lidgren.Network Namespace
Exception thrown in the Lidgren Network Library

Namespace: Lidgren.Network
Syntax

C#

[SerializableAttribute]
public sealed class NetException : Exception

Visual Basic

<SerializableAttribute> _
Public NotInheritable Class NetException _
Inherits Exception

Visual C++

[SerializableAttribute]
public ref class NetException sealed : public Exception
Inheritance Hierarchy

System...Object
System...Exception
Lidgren.Network...NetException
See Also

NetException Members
Lidgren.Network Namespace
The `NetException` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetException()()()</td>
<td>NetException constructor</td>
</tr>
<tr>
<td>NetException(String)</td>
<td>NetException constructor</td>
</tr>
<tr>
<td>NetException(String, Exception)</td>
<td>NetException constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assert(Boolean)</td>
<td>Throws an exception, in DEBUG only, if first parameter is false</td>
</tr>
<tr>
<td>Assert(Boolean, String)</td>
<td>Throws an exception, in DEBUG only, if first parameter is false</td>
</tr>
<tr>
<td></td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetBaseException</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Exception.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>When overridden in a derived class, sets the SerializationInfo with information about the exception.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetObjectData</td>
<td>Gets the runtime type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Exception.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates and returns a string representation of the current exception.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Exception.)</td>
</tr>
<tr>
<td>ToString</td>
<td></td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from <em>Exception.</em>)</td>
</tr>
<tr>
<td><strong>HelpLink</strong></td>
<td>Gets or sets a link to the help file associated with this exception. (Inherited from <em>Exception.</em>)</td>
</tr>
<tr>
<td><strong>HResult</strong></td>
<td>Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from <em>Exception.</em>)</td>
</tr>
<tr>
<td><strong>InnerException</strong></td>
<td>Gets the <em>Exception</em> instance that caused the current exception. (Inherited from <em>Exception.</em>)</td>
</tr>
<tr>
<td><strong>Message</strong></td>
<td>Gets a message that describes the current exception. (Inherited from <em>Exception.</em>)</td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td>Gets or sets the name of the application or the object that causes the error. (Inherited from <em>Exception.</em>)</td>
</tr>
<tr>
<td><strong>StackTrace</strong></td>
<td>Gets a string representation of the frames on the call stack at the time the current exception was thrown. (Inherited from <em>Exception.</em>)</td>
</tr>
<tr>
<td><strong>TargetSite</strong></td>
<td>Gets the method that throws the current exception. (Inherited from <em>Exception.</em>)</td>
</tr>
</tbody>
</table>
See Also

NetException Class
Lidgren.Network Namespace
NetException Constructor

NetException Class  See Also  Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetException()()()</td>
<td>NetException constructor</td>
</tr>
<tr>
<td>NetException(String)</td>
<td>NetException constructor</td>
</tr>
<tr>
<td>NetException(String, Exception)</td>
<td>NetException constructor</td>
</tr>
</tbody>
</table>
See Also

NetException Class
NetException Members
Lidgren.Network Namespace
NetException constructor

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetException()

Visual Basic

Public Sub New

Visual C++

public:
NetException()
See Also

NetException Class
NetException Overload
Lidgren.Network Namespace
NetException constructor

Namespace: Lidgren.Network
Syntax

C#

public NetException(
  string message
)

Visual Basic

Public Sub New ( _
  message As String _
)

Visual C++

public:
NetException(
  String^ message
)

Parameters

message
  Type: System:::String

See Also

NetException Class
NetException Overload
Lidgren.Network Namespace
NetException Constructor (String, Exception)

Namespace: Lidgren.Network
Syntax

C#

public NetException(
    string message,
    Exception inner
)

Visual Basic

Public Sub New (_
    message As String, _
    inner As Exception _
)

Visual C++

public:
NetException(
    String^ message,
    Exception^ inner
)

Parameters

message
  Type: System..::.String

inner
  Type: System..::.Exception
See Also

NetException Class
NetException Overload
Lidgren.Network Namespace
The `NetException` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Assert(Boolean)</code></td>
<td>Throws an exception, in DEBUG only, if first parameter is false.</td>
</tr>
<tr>
<td><code>Assert(Boolean, String)</code></td>
<td>Throws an exception, in DEBUG only, if first parameter is false. Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><code>Equals</code></td>
<td>Calculates whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><code>Finalize</code></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><code>GetBaseException</code></td>
<td>When overridden in a derived class, returns the Exception that is the root cause of one or more subsequent exceptions. (Inherited from Exception.)</td>
</tr>
<tr>
<td><code>GetHashCode</code></td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td><code>GetObjectData</code></td>
<td>When overridden in a derived class, sets the SerializationInfo with information about the exception. (Inherited from Exception.)</td>
</tr>
<tr>
<td><code>GetType</code></td>
<td>Gets the runtime type of the current instance. (Inherited from Exception.)</td>
</tr>
<tr>
<td><code>MemberwiseClone</code></td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td><code>ToString</code></td>
<td>Creates and returns a string representation of the current exception. (Inherited from Exception.)</td>
</tr>
</tbody>
</table>
See Also

NetException Class
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetException...:::Assert Method

NetException Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assert(Boolean)</td>
<td>Throws an exception, in DEBUG only, if first parameter is false</td>
</tr>
<tr>
<td>Assert(Boolean, String)</td>
<td>Throws an exception, in DEBUG only, if first parameter is false</td>
</tr>
</tbody>
</table>
See Also

NetException Class
NetException Members
Lidgren.Network Namespace
Throws an exception, in DEBUG only, if first parameter is false.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public static void Assert(bool isOk)
```

Visual Basic

```vbnet
Public Shared Sub Assert(_
isOk As Boolean _
)
```

Visual C++

```csharp
public:
static void Assert( 
    bool isOk
)
```

Parameters

isOk

Type: `System.Boolean`

See Also

NetException Class
Assert Overload
Lidgren.Network Namespace
NetException..::.Assert Method (Boolean, String)

Throws an exception, in DEBUG only, if first parameter is false

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
### Syntax

**C#**

```csharp
public static void Assert(
    bool isOk,
    string message
)
```

**Visual Basic**

```vbnet
Public Shared Sub Assert (_
    isOk As Boolean, _
    message As String _
)
```

**Visual C++**

```cpp
public:
static void Assert(
    bool isOk,
    String^ message
)
```

### Parameters

**isOk**

Type: **System:::Boolean**


**message**

Type: **System:::String**

See Also

NetException Class
Assert Overload
Lidgren.Network Namespace
The `NetException` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Gets a collection of key/value pairs that provide additional user-defined information about the exception. (Inherited from Exception.)</td>
</tr>
<tr>
<td>HelpLink</td>
<td>Gets or sets a link to the help file associated with this exception. (Inherited from Exception.)</td>
</tr>
<tr>
<td>HResult</td>
<td>Gets or sets HRESULT, a coded numerical value that is assigned to a specific exception. (Inherited from Exception.)</td>
</tr>
<tr>
<td>InnerException</td>
<td>Gets the Exception instance that caused the current exception. (Inherited from Exception.)</td>
</tr>
<tr>
<td>Message</td>
<td>Gets a message that describes the current exception. (Inherited from Exception.)</td>
</tr>
<tr>
<td>Source</td>
<td>Gets or sets the name of the application or the object that causes the error. (Inherited from Exception.)</td>
</tr>
<tr>
<td>StackTrace</td>
<td>Gets a string representation of the frames on the call stack at the time the current exception was thrown. (Inherited from Exception.)</td>
</tr>
<tr>
<td>TargetSite</td>
<td>Gets the method that throws the current exception. (Inherited from Exception.)</td>
</tr>
</tbody>
</table>
See Also

NetException Class
Lidgren.Network Namespace
Incoming message either sent from a remote peer or generated within the library

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
# Syntax

**C#**

```csharp
public sealed class NetIncomingMessage : NetBuffer
```

**Visual Basic**

```vbnet
Public NotInheritable Class NetIncomingMessage
    Inherits NetBuffer
```

**Visual C++**

```cpp
public ref class NetIncomingMessage sealed : public NetBuffer
```
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetBuffer
Lidgren.Network..::..NetIncomingMessage
See Also

NetIncomingMessage Members
Lidgren.Network Namespace
The NetIncomingMessage type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt a message</td>
</tr>
<tr>
<td>EnsureBufferSize</td>
<td>Ensures the buffer can hold this number of bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>PeekBoolean</td>
<td>Reads a 1-bit Boolean without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte()()()()</td>
<td>Reads a Byte without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte(Int32)</td>
<td>Reads the specified number of bits into a Byte without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>PeekBytes(Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekBytes(array[Byte][][], Int32, Int32)</td>
<td>Reads the specified number of bytes into an Int32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekDataBuffer</td>
<td>Gets the internal data buffer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekDouble</td>
<td>Reads a 64-bit Double without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekFloat</td>
<td>Reads a 32-bit Single without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt16</td>
<td>Reads an Int16 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt32()()()</td>
<td>Reads an Int32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt64()()()</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekSByte</td>
<td>Reads an SByte without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekSingle</td>
<td>Reads a 32-bit Single without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekSingle()()()()</td>
<td>Reads a string without advancing the read pointer</td>
</tr>
</tbody>
</table>
- **PeekString**
  Reads the specified number of bits into a read pointer
  (Inherited from NetBuffer.)

- **PeekUInt16**
  Reads a UInt16 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt32(0)**
  Reads a UInt32 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt32(Int32)**
  Reads the specified number of bits into a UInt32 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt64(0)**
  Reads a UInt64 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt64(Int32)**
  Reads the specified number of bits into an UInt64 without advancing the read pointer
  (Inherited from NetBuffer.)

- **ReadAllFields(Object)**
  Reads all public and private declared instance fields of the object in alphabetical order using reflection
  (Inherited from NetBuffer.)

- **ReadAllFields(Object, BindingFlags)**
  Reads all fields with the specified binding of the object in alphabetical order using reflection
  (Inherited from NetBuffer.)

- **ReadAllProperties(Object)**
  Reads all public and private declared instance fields of the object in alphabetical order using reflection
  (Inherited from NetBuffer.)

- **ReadAllProperties(Object, BindingFlags)**
  Reads all fields with the specified binding of the object in alphabetical order using reflection
  (Inherited from NetBuffer.)

- **Reads the specified number of bits into**
- **ReadBits**
  a preallocated array
  (Inherited from [NetBuffer](#).)

- **ReadBoolean**
  Reads a boolean value (stored as a single bit) written using Write(bool)
  (Inherited from [NetBuffer](#).)

- **.ReadByte]**
  Reads a byte
  (Inherited from [NetBuffer](#).)

- **ReadStream(Byte%)**
  Reads a byte and returns true or false for success
  (Inherited from [NetBuffer](#).)

- **ReadStream(Byte] Int32)**
  Reads 1 to 8 bits into a byte
  (Inherited from [NetBuffer](#).)

- **ReadStream(Int32)**
  Reads the specified number of bytes
  (Inherited from [NetBuffer](#).)

- **ReadStream(array<Byte> Int32, Int32)**
  Reads the specified number of bytes into a preallocated array
  (Inherited from [NetBuffer](#).)

- **ReadStream(ReadBytes(Int32, array<Byte> Int32)] Int32)**
  Reads a 64 bit floating point value written using Write(Double)
  (Inherited from [NetBuffer](#).)

- **ReadStream(ReadFloat)**
  Reads a 32 bit floating point value written using Write(Single)
  (Inherited from [NetBuffer](#).)

- **ReadStream(ReadInt16)**
  Reads a 16 bit signed integer written using Write(Int16)
  (Inherited from [NetBuffer](#).)

- **ReadStream(ReadInt32)**
  Reads a 32 bit signed integer written using Write(Int32)
  (Inherited from [NetBuffer](#).)

- **ReadStream(ReadInt32(Int32)**
  Reads a signed integer stored in 1 to 32 bits, written using Write(Int32, Int32)
  (Inherited from [NetBuffer](#).)

- **ReadStream(ReadInt32(Int32)**
  Reads a 32 bit signed integer written
- **ReadInt32(Int32%)**
  Using Write(Int32)
  (Inherited from NetBuffer.)

- **ReadInt64()**
  Reads a 64 bit signed integer written using Write(Int64)
  (Inherited from NetBuffer.)

- **ReadInt64(Int32)**
  Reads a signed integer stored in 1 to 64 bits, written using Write(Int64, Int32)
  (Inherited from NetBuffer.)

- **ReadIPEndpoint**
  Reads a stored IPv4 endpoint description
  (Inherited from NetBuffer.)

- **ReadPadBits**
  Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.
  (Inherited from NetBuffer.)

- **ReadRangedInteger**
  Reads a 32 bit integer value written using WriteRangedInteger()
  (Inherited from NetBuffer.)

- **ReadRangedSingle**
  Reads a 32 bit floating point value written using WriteRangedSingle()
  (Inherited from NetBuffer.)

- **ReadSByte**
  Reads a signed byte
  (Inherited from NetBuffer.)

- **ReadSignedSingle**
  Reads a 32 bit floating point value written using WriteSignedSingle()
  (Inherited from NetBuffer.)

- **ReadSingle()**
  Reads a 32 bit floating point value written using Write(Single)
  (Inherited from NetBuffer.)

- **ReadSingle(Single%)**
  Reads a 32 bit floating point value written using Write(Single)
  (Inherited from NetBuffer.)

- **ReadString()**
  Reads a string written using Write(string)
  (Inherited from NetBuffer.)

- **ReadString()**
  Reads a string written using...
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadString(String%)</code></td>
<td>Write(string) and returns true for success (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadTime(Boolean)</code></td>
<td>Reads a value, in local time comparable to NetTime.Now, written using WriteTime() Must have a connected sender</td>
</tr>
<tr>
<td><code>ReadTime(NetConnection, Boolean)</code></td>
<td>Reads a value, in local time comparable to NetTime.Now, written using WriteTime() for the connection supplied (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadUInt16</code></td>
<td>Reads a 16 bit unsigned integer written using Write(UInt16) (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadUInt32(UInt32%)</code></td>
<td>Reads an unsigned integer written using Write(UInt32) and returns true for success (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadUInt32(Int32)</code></td>
<td>Reads a 32 bit unsigned integer written using Write(UInt32, Int32) (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadUInt64(UInt64%)</code></td>
<td>Reads a 64 bit unsigned integer written using Write(UInt64) (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadUInt64(Int32)</code></td>
<td>Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64, Int32) (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadUnitSingle</code></td>
<td>Reads a 32 bit floating point value written using WriteUnitSingle() (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>Reads a variable sized Int32 written</code></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ReadVariableInt32</strong></td>
<td>using WriteVariableInt32() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadVariableInt64</strong></td>
<td>Reads a variable sized Int64 written using WriteVariableInt64() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadVariableUInt32()</strong></td>
<td>using WriteVariableUInt32() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadVariableUInt32(UInt32%)</strong></td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadVariableUInt64</strong></td>
<td>using WriteVariableInt64() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>SkipPadBits()</strong></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>SkipPadBits(Int32)</strong></td>
<td>Pads data with the specified number of bits. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a string that represents this object (Overrives Object.....ToString()())</td>
</tr>
<tr>
<td><strong>Write(Boolean)</strong></td>
<td>Writes a boolean value using 1 bit (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Byte)</strong></td>
<td>Write a byte (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(array&lt;Byte&gt;[][])</strong></td>
<td>Writes all bytes in an array (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Double)</strong></td>
<td>Writes a 64 bit floating point value (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Int16)</strong></td>
<td>Writes a signed 16 bit integer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Int32)</strong></td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>Write(Int64)</td>
<td>Writes a 64 bit signed integer</td>
</tr>
<tr>
<td>Write(IPEndPoint)</td>
<td>Writes an endpoint description</td>
</tr>
<tr>
<td>Write(SByte)</td>
<td>Writes a signed byte</td>
</tr>
<tr>
<td>Write(Single)</td>
<td>Writes a 32 bit floating point value</td>
</tr>
<tr>
<td>Write(String)</td>
<td>Write a string</td>
</tr>
<tr>
<td>Write(UInt16)</td>
<td>Writes an unsigned 16 bit integer</td>
</tr>
<tr>
<td>Write(UInt32)</td>
<td>Writes a 32 bit unsigned integer</td>
</tr>
<tr>
<td>Write(UInt64)</td>
<td>Writes a 64 bit unsigned integer</td>
</tr>
<tr>
<td>Write(NetIncomingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(NetOutgoingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(Byte, Int32)</td>
<td>Writes 1 to 8 bits of a byte</td>
</tr>
<tr>
<td>Write(Int32, Int32)</td>
<td>Writes a signed integer using 1 to 32 bits</td>
</tr>
<tr>
<td>Write(Int64, Int32)</td>
<td>Writes a signed integer using 1 to 64 bits</td>
</tr>
<tr>
<td>Write(UInt16, Int32)</td>
<td>Writes an unsigned integer using 1 to 16 bits</td>
</tr>
<tr>
<td>Write(UInt32, Int32)</td>
<td>Writes a 32 bit signed integer</td>
</tr>
</tbody>
</table>
- **Write(UInt64, Int32)**: Writes an unsigned integer using 1 to 64 bits (Inherited from NetBuffer.)

- **Write(array<Byte>[], Int32, Int32)**: Writes the specified number of bytes from an array (Inherited from NetBuffer.)

- **WriteAllFields(Object)**: Writes all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from NetBuffer.)

- **WriteAllFields(Object, BindingFlags)**: Writes all fields with specified binding in alphabetical order using reflection (Inherited from NetBuffer.)

- **WriteAllProperties(Object)**: Writes all public and private declared instance properties of the object in alphabetical order using reflection (Inherited from NetBuffer.)

- **WriteAllProperties(Object, BindingFlags)**: Writes all properties with specified binding in alphabetical order using reflection (Inherited from NetBuffer.)

- **WritePadBits()** Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)

- **WritePadBits(Int32)**: Pads data with the specified number of bits. (Inherited from NetBuffer.)

- **WriteRangedInteger**: Writes an integer with the least amount of bits need for the specified range

  Returns number of bits written (Inherited from NetBuffer.)

- **WriteRangedSingle**: Compress a float within a specified range using a certain number of bits (Inherited from NetBuffer.)
<table>
<thead>
<tr>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteSignedSingle</td>
<td>Compress (lossy) a float in the range -1..1 using numberOfBits bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteTime(Boolean)</td>
<td>Writes the current local time to a message; readable (and convertable to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteTime(Double, Boolean)</td>
<td>Writes a local timestamp to a message; readable (and convertable to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteUnitSingle</td>
<td>Compress (lossy) a float in the range 0..1 using numberOfBits bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableInt32</td>
<td>Write Base128 encoded variable sized signed integer of up to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableInt64</td>
<td>Write Base128 encoded variable sized signed integer of up to 64 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableUInt32</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableUInt64</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 64 bits (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>Gets or sets the internal data buffer</td>
</tr>
<tr>
<td></td>
<td><em>(Inherited from <a href="#">NetBuffer</a>).</em></td>
</tr>
<tr>
<td><strong>DeliveryMethod</strong></td>
<td>Gets the delivery method this message was sent with</td>
</tr>
<tr>
<td></td>
<td><em>(if user data)</em></td>
</tr>
<tr>
<td><strong>LengthBits</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bits</td>
</tr>
<tr>
<td></td>
<td><em>(Inherited from <a href="#">NetBuffer</a>).</em></td>
</tr>
<tr>
<td><strong>LengthBytes</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bytes</td>
</tr>
<tr>
<td></td>
<td><em>(Inherited from <a href="#">NetBuffer</a>).</em></td>
</tr>
<tr>
<td><strong>MessageType</strong></td>
<td>Gets the type of this incoming message</td>
</tr>
<tr>
<td><strong>Position</strong></td>
<td>Gets or sets the read position in the buffer, in bits</td>
</tr>
<tr>
<td></td>
<td><em>(not bytes)</em></td>
</tr>
<tr>
<td></td>
<td><em>(Inherited from <a href="#">NetBuffer</a>).</em></td>
</tr>
<tr>
<td><strong>PositionInBytes</strong></td>
<td>Gets the position in the buffer in bytes; note that the bits of the first returned byte may already have been read - check the Position property to make sure.</td>
</tr>
<tr>
<td></td>
<td><em>(Inherited from <a href="#">NetBuffer</a>).</em></td>
</tr>
<tr>
<td><strong>ReceiveTime</strong></td>
<td>What local time the message was received from the network</td>
</tr>
<tr>
<td><strong>SenderConnection</strong></td>
<td>NetConnection of sender, if any</td>
</tr>
<tr>
<td><strong>SenderEndpoint</strong></td>
<td>IPEndPoint of sender, if any</td>
</tr>
<tr>
<td><strong>SequenceChannel</strong></td>
<td>Gets the sequence channel this message was sent with <em>(if user data)</em></td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
The **NetIncomingMessage** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt a message</td>
</tr>
<tr>
<td>EnsureBufferSize</td>
<td>Ensures the buffer can hold this number of bits</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>PeekBoolean</td>
<td>Reads a 1-bit Boolean without advancing the read pointer</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte()()()()()()</td>
<td>Reads a Byte without advancing the read pointer</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte(Int32)</td>
<td>Reads the specified number of bits into a Byte without advancing the read</td>
</tr>
<tr>
<td></td>
<td>pointer</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td></td>
<td>Reads the specified number of bytes</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>PeekBytes(Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekBytes(array&lt;Byte&gt;[][], Int32, Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekDataBuffer</td>
<td>Gets the internal data buffer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekDouble</td>
<td>Reads a 64-bit Double without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekFloat</td>
<td>Reads a 32-bit Single without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt16</td>
<td>Reads an Int16 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt32()()()()</td>
<td>Reads an Int32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads the specified number of bits into an Int32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt64()()()()</td>
<td>Reads an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekSByte</td>
<td>Reads an SByte without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekSingle</td>
<td>Reads a 32-bit Single without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
| PeekSingle | Reads a string without advancing the
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekString</td>
<td>Reads a UInt16 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt16</td>
<td>Reads a UInt32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt32(0)</td>
<td>Reads the specified number of bits into a UInt32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt32(Int32)</td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt64(0)</td>
<td>Reads the specified number of bits into an UInt64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt64(Int32)</td>
<td>Reads the specified number of bits into an UInt64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadAllFields(Object)</td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadAllFields(Object, BindingFlags)</td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadAllProperties(Object)</td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadAllProperties(Object, BindingFlags)</td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ReadBits</td>
<td>a preallocated array</td>
</tr>
<tr>
<td>ReadBoolean</td>
<td>Reads a boolean value (stored as a single bit) written using Write(bool)</td>
</tr>
<tr>
<td>ReadByte()()()()()()</td>
<td>Reads a byte</td>
</tr>
<tr>
<td>ReadByte(Byte%)</td>
<td>Reads a byte and returns true or false for success</td>
</tr>
<tr>
<td>ReadByte(Int32)</td>
<td>Reads 1 to 8 bits into a byte</td>
</tr>
<tr>
<td>ReadBytes(Int32)</td>
<td>Reads the specified number of bytes</td>
</tr>
<tr>
<td>ReadBytes(Int32, array&lt;Byte&gt;[]Q[])[]%</td>
<td>Reads the specified number of bytes and returns true for success</td>
</tr>
<tr>
<td>ReadBytes(array&lt;Byte&gt;[]Q[], Int32, Int32)</td>
<td>Reads the specified number of bytes into a preallocated array</td>
</tr>
<tr>
<td>ReadDouble</td>
<td>Reads a 64 bit floating point value written using Write(Double)</td>
</tr>
<tr>
<td>ReadFloat</td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td>ReadInt16</td>
<td>Reads a 16 bit signed integer written using Write(Int16)</td>
</tr>
<tr>
<td>ReadInt32QQ()Q()()()Q</td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td>ReadInt32(Int32)</td>
<td>Reads a signed integer stored in 1 to 32 bits, written using Write(Int32, Int32)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>ReadInt32(Int32%)</code></td>
<td>Reads a 32 bit integer value written using <code>Write(Int32)</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadInt64()</code></td>
<td>Reads a 64 bit signed integer written using <code>Write(Int64)</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadInt64(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 64 bits, written using <code>Write(Int64, Int32)</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadIPEndpoint</code></td>
<td>Reads a stored IPv4 endpoint description (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadPadBits</code></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadRangedInteger</code></td>
<td>Reads a 32 bit integer value written using <code>WriteRangedInteger()</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadRangedSingle</code></td>
<td>Reads a 32 bit floating point value written using <code>WriteRangedSingle()</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadSByte</code></td>
<td>Reads a signed byte (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadSignedSingle</code></td>
<td>Reads a 32 bit floating point value written using <code>WriteSignedSingle()</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadSingle()</code></td>
<td>Reads a 32 bit floating point value written using <code>Write(Single)</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadSingle(Single%)</code></td>
<td>Reads a 32 bit floating point value written using <code>Write(Single)</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadString()</code></td>
<td>Reads a string written using <code>Write(string)</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td><code>ReadString(String%)</code></td>
<td>Reads a string written using <code>Write(string)</code> (Inherited from <code>NetBuffer</code>).</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>ReadString(String%)</strong></td>
<td>Write(string) and returns true for success</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadTime(Boolean)</strong></td>
<td>Reads a value, in local time comparable to NetTime.Now, written using</td>
</tr>
<tr>
<td></td>
<td>WRITETime() Must have a connected sender</td>
</tr>
<tr>
<td><strong>ReadTime(NetConnection, Boolean)</strong></td>
<td>Reads a value, in local time comparable to NetTime.Now, written using</td>
</tr>
<tr>
<td></td>
<td>WRITETime() for the connection supplied</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadUInt16</strong></td>
<td>Reads a 16 bit unsigned integer written using WRITE(UInt16)</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadUInt32()</strong></td>
<td>Reads an 32 bit unsigned integer written using WRITE(UInt32)</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadUInt32(Int32)</strong></td>
<td>Reads an unsigned integer stored in 1 to 32 bits, written using WRITE(UInt32,</td>
</tr>
<tr>
<td></td>
<td>Int32)</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadUInt32(UInt32%)</strong></td>
<td>Reads an 32 bit unsigned integer written using WRITE(UInt32) and returns</td>
</tr>
<tr>
<td></td>
<td>true for success</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadUInt64()</strong></td>
<td>Reads a 64 bit unsigned integer written using WRITE(UInt64)</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadUInt64(Int32)</strong></td>
<td>Reads an unsigned integer stored in 1 to 64 bits, written using WRITE(UInt64,</td>
</tr>
<tr>
<td></td>
<td>Int32)</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>ReadUnitSingle</strong></td>
<td>Reads a 32 bit floating point value written using WRITEUnitSingle()</td>
</tr>
<tr>
<td><em>(Inherited from NetBuffer.)</em></td>
<td></td>
</tr>
<tr>
<td><strong>Reads a variable sized Int32 written</strong></td>
<td></td>
</tr>
</tbody>
</table>
- **ReadVariableInt32**
  using WriteVariableInt32()
  (Inherited from NetBuffer.)

- **ReadVariableInt64**
  Reads a variable sized Int64 written
  using WriteVariableInt64()
  (Inherited from NetBuffer.)

- **ReadVariableUInt32**

- **ReadVariableUInt32(UInt32%)**
  Reads a variable sized UInt32 written
  using WriteVariableUInt32() and
  returns true for success
  (Inherited from NetBuffer.)

- **ReadVariableUInt64**
  Reads a variable sized UInt32 written
  using WriteVariableInt64()
  (Inherited from NetBuffer.)

- **SkipPadBits()**
  Pads data with enough bits to reach a
  full byte. Decreases cpu usage for
  subsequent byte writes.
  (Inherited from NetBuffer.)

- **SkipPadBits(Int32)**
  Pads data with the specified number of
  bits.
  (Inherited from NetBuffer.)

- **ToString**
  Returns a string that represents this
  object
  (Overrides Object::ToString().)

- **Write(Boolean)**
  Writes a boolean value using 1 bit
  (Inherited from NetBuffer.)

- **Write(Byte)**
  Write a byte
  (Inherited from NetBuffer.)

- **Write(array<Byte>[][])**
  Writes all bytes in an array
  (Inherited from NetBuffer.)

- **Write(Double)**
  Writes a 64 bit floating point value
  (Inherited from NetBuffer.)

- **Write(Int16)**
  Writes a signed 16 bit integer
  (Inherited from NetBuffer.)

- **Write(Int32)**
  Writes a 32 bit signed integer
- **Write(Int64)** - Writes a 64 bit signed integer  
  (Inherited from [NetBuffer](#).)

- **Write(IPEndPoint)** - Writes an endpoint description  
  (Inherited from [NetBuffer](#).)

- **Write(SByte)** - Writes a signed byte  
  (Inherited from [NetBuffer](#).)

- **Write(Single)** - Writes a 32 bit floating point value  
  (Inherited from [NetBuffer](#).)

- **Write(String)** - Write a string  
  (Inherited from [NetBuffer](#).)

- **Write(UInt16)** - Writes an unsigned 16 bit integer  
  (Inherited from [NetBuffer](#).)

- **Write(UInt32)** - Writes a 32 bit unsigned integer  
  (Inherited from [NetBuffer](#).)

- **Write(UInt64)** - Writes a 64 bit unsigned integer  
  (Inherited from [NetBuffer](#).)

- **Write(NetIncomingMessage)** - Append all the bits of message to this message  
  (Inherited from [NetBuffer](#).)

- **Write(NetOutgoingMessage)** - Append all the bits of message to this message  
  (Inherited from [NetBuffer](#).)

- **Write(Byte, Int32)** - Writes 1 to 8 bits of a byte  
  (Inherited from [NetBuffer](#).)

- **Write(Int32, Int32)** - Writes a signed integer using 1 to 32 bits  
  (Inherited from [NetBuffer](#).)

- **Write(Int64, Int32)** - Writes a signed integer using 1 to 64 bits  
  (Inherited from [NetBuffer](#).)

- **Write(UInt16, Int32)** - Writes an unsigned integer using 1 to 16 bits  
  (Inherited from [NetBuffer](#).)

- **Write(UInt32, Int32)** - Writes a 32 bit signed integer
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write(UInt64, Int32)</td>
<td>Writes an unsigned integer using 1 to 64 bits (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>Write(array&lt;Byte&gt;[I][I],[Int32, Int32])</td>
<td>Writes the specified number of bytes from an array (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>WriteAllFields(Object)</td>
<td>Writes all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>WriteAllFields(Object, BindingFlags)</td>
<td>Writes all fields with specified binding in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>WriteAllProperties(Object)</td>
<td>Writes all public and private declared instance properties of the object in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>WriteAllProperties(Object, BindingFlags)</td>
<td>Writes all properties with specified binding in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>WritePadBits()()()()</td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>WritePadBits(Int32)</td>
<td>Pads data with the specified number of bits. (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>WriteRangedInteger</td>
<td>Writes an integer with the least amount of bits need for the specified range.</td>
</tr>
<tr>
<td>WriteRangedSingle</td>
<td>Returns number of bits written (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td></td>
<td>Compress a float within a specified range using a certain number of bits (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WriteSignedSingle</td>
<td>Compress (lossy) a float in the range -1..1 using numberOfBits bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteTime(Boolean)</td>
<td>Writes the current local time to a message; readable (and convertible to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteTime(Double, Boolean)</td>
<td>Writes a local timestamp to a message; readable (and convertible to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteUnitSingle</td>
<td>Compress (lossy) a float in the range 0..1 using numberOfBits bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableInt32</td>
<td>Write Base128 encoded variable sized signed integer of up to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableInt64</td>
<td>Write Base128 encoded variable sized signed integer of up to 64 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableUInt32</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteVariableUInt64</td>
<td>Write Base128 encoded variable sized unsigned integer of up to 64 bits (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
Decrypt a message

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool Decrypt(
    INetEncryption encryption
)
```

Visual Basic

```vbnet
Public Function Decrypt (_
    encryption As INetEncryption _
) As Boolean
```

Visual C++

```c++
public:
bool Decrypt(
    INetEncryption^ encryption
)
```

Parameters

encryption
  Type: Lidgren.Network::INetEncryption
  The encryption algorithm used to encrypt the message

Return Value

true on success
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
C#  Visual Basic  Visual C++  Include Protected Members  Include Inherited Members  Lidgren Network Library documentation

NetIncomingMessage::PeekByte Method

NetIncomingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekByte()()()()</td>
<td>Reads a Byte without advancing the read pointer</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte(Int32)</td>
<td>Reads the specified number of bits into a Byte without advancing the read pointer</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage::PeekBytes Method

NetIncomingMessage Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>PeekBytes(Int32)</code></td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>PeekBytes(array&lt;Byte&gt;[][], Int32, Int32)</code></td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from <code>NetBuffer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage::PeekInt32 Method

NetIncomingMessage Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekInt32()()()()</td>
<td>Reads an Int32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads the specified number of bits into an Int32</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage...:PeekInt64 Method

NetIncomingMessage Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekInt64()()()</td>
<td>Reads an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
NetIncomingMessage::PeekUInt32 Method

NetIncomingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekUInt32()()</td>
<td>Reads a UInt32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt32(Int32)</td>
<td>Reads the specified number of bits into a UInt32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage::PeekUInt64 Method

NetIncomingMessage Class See Also Send Feedback
Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekUInt64()</td>
<td>Reads a UInt64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt64(Int32)</td>
<td>Reads the specified number of bits into an UInt64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage::ReadAllFields Method

NetIncomingMessage Class
See Also
Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadAllFields(Object)</td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadAllFields(Object, BindingFlags)</td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
C# Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetIncomingMessage...ReadAllProperties Method
NetIncomingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadAllProperties(Object)</code></td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadAllProperties(Object, BindingFlags)</code></td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage::.ReadByte Method

NetIncomingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadByte()()()</td>
<td>Reads a byte (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadByte(Byte%)</td>
<td>Reads a byte and returns true or false for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadByte(Int32)</td>
<td>Reads 1 to 8 bits into a byte (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
NetIncomingMessage...::...ReadBytes Method

NetIncomingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadBytes(Int32)</code></td>
<td>Reads the specified number of bytes (Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
<tr>
<td><code>ReadBytes(Int32, array&lt;Byte&gt;[][])</code></td>
<td>Reads the specified number of bytes and returns true for success</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
<tr>
<td><code>ReadBytes(array&lt;Byte&gt;[][], Int32, Int32)</code></td>
<td>Reads the specified number of bytes into a preallocated array</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetIncomingMessage
.NetIncomingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadInt32()()()</strong></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadInt32(Int32)</strong></td>
<td>Reads a signed integer stored in 1 to 32 bits, written using Write(Int32, Int32)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadInt32(Int32%)</strong></td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage::...ReadInt64 Method

NetIncomingMessage Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadInt64()()()()</td>
<td>Reads a 64 bit signed integer written using Write(Int64) [\text{Inherited from NetBuffer.})</td>
</tr>
<tr>
<td>ReadInt64(Int32)</td>
<td>Reads a signed integer stored in 1 to 64 bits, written using Write(Int64, Int32) [\text{Inherited from NetBuffer.})</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage...ReadSingle Method

See Also  Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadSingle()()()</td>
<td>Reads a 32 bit floating point value written using Write(Single) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadSingle(Single%)</td>
<td>Reads a 32 bit floating point value written using Write(Single) (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetIncomingMessage.....ReadString Method

NetIncomingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadString</td>
<td>Reads a string written using Write(string)</td>
</tr>
<tr>
<td>ReadString(String%)</td>
<td>Reads a string written using Write(string) and returns true for success</td>
</tr>
</tbody>
</table>

(Inherited from NetBuffer.)
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage::ReadTime Method

NetIncomingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadTime(Boolean)</code></td>
<td>Reads a value, in local time comparable to NetTime.Now, written using WriteTime() Must have a connected sender</td>
</tr>
<tr>
<td><code>ReadTime(NetConnection, Boolean)</code></td>
<td>Reads a value, in local time comparable to NetTime.Now, written using WriteTime() for the connection supplied (Inherited from <code>NetBuffer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage.ReadTime Method (Boolean)

NetIncomingMessage Class  See Also  Send Feedback

Reads a value, in local time comparable to NetTime.Now, written using WriteTime() Must have a connected sender

Namespace: Lidgren.Network
Syntax

C#

public double ReadTime(
    bool highPrecision
)

Visual Basic

Public Function ReadTime (_
    highPrecision As Boolean _
) As Double

Visual C++

public:
    double ReadTime(
        bool highPrecision
    )

Parameters

highPrecision
    Type: System::Boolean


Return Value

See Also

NetIncomingMessage Class
ReadTime Overload
Lidgren.Network Namespace
NetIncomingMessage.::ReadUInt32 Method

NetIncomingMessage Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadUInt32()</strong></td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadUInt32(Int32)</strong></td>
<td>Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadUInt32(UInt32%)</strong></td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage::ReadUInt64 Method

NetIncomingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadUInt64()()()</strong></td>
<td>Reads a 64 bit unsigned integer written using Write(UInt64)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td><strong>ReadUInt64(Int32)</strong></td>
<td>Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64, Int32)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage...:ReadVariableU32 Method
NetIncomingMessage Class See Also Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="ReadVariableUIInt32(QQQQ)" alt="ReadVariableUIInt32(QQQQ)" /></td>
<td>Reads a variable sized UIInt32 written using WriteVariableUIInt32() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><img src="ReadVariableUIInt32(UIInt32%25)" alt="ReadVariableUIInt32(UIInt32%)" /></td>
<td>Reads a variable sized UIInt32 written using WriteVariableUIInt32() and returns true for success (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="skip_pad_bits_400" alt="SkipPadBits()()()" /></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><img src="skip_pad_bits_int32" alt="SkipPadBits(Int32)" /></td>
<td>Pads data with the specified number of bits. (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage::ToString Method

**NetIncomingMessage Class**  **See Also**  **Send Feedback**

Returns a string that represents this object

**Namespace:**  [Lidgren.Network](#)
**Assembly:**  Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public override string ToString()
```

**Visual Basic**

```vbnet
Public Overrides Function ToString As String
```

**Visual C++**

```cpp
public:
virtual String^ ToString() override
```

**Return Value**

See Also

NetIncomingMessage Class
Lidgren.Network Namespace
NetIncomingMessage Class
See Also
Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Write(Boolean)</strong></td>
<td>Writes a boolean value using 1 bit (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(Byte)</strong></td>
<td>Write a byte (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(array[Byte][][[]])</strong></td>
<td>Writes all bytes in an array (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(Double)</strong></td>
<td>Writes a 64 bit floating point value (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(Int16)</strong></td>
<td>Writes a signed 16 bit integer (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(Int32)</strong></td>
<td>Writes a 32 bit signed integer (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(Int64)</strong></td>
<td>Writes a 64 bit signed integer (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(IPEndPoint)</strong></td>
<td>Writes an endpoint description (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(SByte)</strong></td>
<td>Writes a signed byte (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(Single)</strong></td>
<td>Writes a 32 bit floating point value (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(String)</strong></td>
<td>Write a string (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(UInt16)</strong></td>
<td>Writes an unsigned 16 bit integer (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(UInt32)</strong></td>
<td>Writes a 32 bit unsigned integer (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(UInt64)</strong></td>
<td>Writes a 64 bit unsigned integer (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Write(NetIncomingMessage)</strong></td>
<td>Append all the bits of message to this message</td>
</tr>
</tbody>
</table>
- **Write(NetOutgoingMessage)**
  Append all the bits of message to this message
  (Inherited from [NetBuffer](#).)

- **Write(Byte, Int32)**
  Writes 1 to 8 bits of a byte
  (Inherited from [NetBuffer](#).)

- **Write(Int32, Int32)**
  Writes a signed integer using 1 to 32 bits
  (Inherited from [NetBuffer](#).)

- **Write(Int64, Int32)**
  Writes a signed integer using 1 to 64 bits
  (Inherited from [NetBuffer](#).)

- **Write(UInt16, Int32)**
  Writes an unsigned integer using 1 to 16 bits
  (Inherited from [NetBuffer](#).)

- **Write(UInt32, Int32)**
  Writes a 32 bit signed integer
  (Inherited from [NetBuffer](#).)

- **Write(UInt64, Int32)**
  Writes an unsigned integer using 1 to 64 bits
  (Inherited from [NetBuffer](#).)

- **Write(array<Byte>[][], Int32, Int32)**
  Writes the specified number of bytes from an array
  (Inherited from [NetBuffer](#).)
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetIncomingMessage......WriteAllFields Method

NetIncomingMessage Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteAllFields(Object)</td>
<td>Writes all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteAllFields(Object, BindingFlags)</td>
<td>Writes all fields with specified binding in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage...WriteAllProperties Method

NetIncomingMessage Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteAllProperties(Object)</td>
<td>Writes all public and private declared instance properties of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteAllProperties(Object, BindingFlags)</td>
<td>Writes all properties with specified binding in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Include Protected Members  Include Inherited Members
Lidgren Network Library documentation
NetIncomingMessage::WritePadBits Method
NetIncomingMessage Class  See Also  Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from <a href="#">NetBuffer</a>.)</td>
<td>WritePadBits()()()</td>
</tr>
<tr>
<td>pads data with the specified number of bits. (Inherited from <a href="#">NetBuffer</a>.)</td>
<td>WritePadBits(Int32)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteTime(Boolean)</td>
<td>Writes the current local time to a message; readable (and convertable to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteTime(Double, Boolean)</td>
<td>Writes a local timestamp to a message; readable (and convertable to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
NetIncomingMessage Members
Lidgren.Network Namespace
The NetIncomingMessage type exposes the following members.
# Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>Gets or sets the internal data buffer (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><strong>DeliveryMethod</strong></td>
<td>Gets the delivery method this message was sent with (if user data)</td>
</tr>
<tr>
<td><strong>LengthBits</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bits (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><strong>LengthBytes</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bytes (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><strong>MessageType</strong></td>
<td>Gets the type of this incoming message</td>
</tr>
<tr>
<td><strong>Position</strong></td>
<td>Gets or sets the read position in the buffer, in bits (not bytes) (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><strong>PositionInBytes</strong></td>
<td>Gets the position in the buffer in bytes; note that the bits of the first returned byte may already have been read - check the Position property to make sure. (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><strong>ReceiveTime</strong></td>
<td>What local time the message was received from the network</td>
</tr>
<tr>
<td><strong>SenderConnection</strong></td>
<td>NetConnection of sender, if any</td>
</tr>
<tr>
<td><strong>SenderEndpoint</strong></td>
<td>IPEndPoint of sender, if any</td>
</tr>
<tr>
<td><strong>SequenceChannel</strong></td>
<td>Gets the sequence channel this message was sent with (if user data)</td>
</tr>
</tbody>
</table>
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
NetIncomingMessage::DeliveryMethod Property

Gets the delivery method this message was sent with (if user data)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetDeliveryMethod DeliveryMethod { get; }

Visual Basic

Public ReadOnly Property DeliveryMethod As NetDeliveryMethod
Get

Visual C++

public:
property NetDeliveryMethod DeliveryMethod {
NetDeliveryMethod get ();
}
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
Gets the type of this incoming message

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetIncomingMessageType MessageType { get; }

Visual Basic

Public ReadOnly Property MessageType As NetIncomingMessageType
    Get

Visual C++

public:
    property NetIncomingMessageType MessageType {
        NetIncomingMessageType get();
    }

See Also

NetIncomingMessage Class
Lidgren.Network Namespace
What local time the message was received from the network

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public double ReceiveTime { get; }

Visual Basic

Public ReadOnly Property ReceiveTime As Double
Get

Visual C++

public:
property double ReceiveTime {
    double get ();
}
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage...SenderConnection Property

NetIncomingMessage Class See Also Send Feedback

NetConnection of sender, if any

Namespace: Lidgren.Network
Syntax

C#

```csharp
public NetConnection SenderConnection { get; }
```

Visual Basic

```vbnet
Public ReadOnly Property SenderConnection As NetConnection
    Get
```

Visual C++

```cpp
public:
    property NetConnection^ SenderConnection { 
        NetConnection^ get ();
    }
```
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage::..::.SenderEndpoint Property

NetIncomingMessage Class  See Also  Send Feedback

IPEndPoint of sender, if any

**Namespace:**  Lidgren.Network

**Assembly:**  Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
C#

public IPEndPoint SenderEndpoint { get; }

Visual Basic

Public ReadOnly Property SenderEndpoint As IPEndPoint
Get

Visual C++

public:
property IPEndPoint^ SenderEndpoint {
    IPEndPoint^ get ();
}
See Also

NetIncomingMessage Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetIncomingMessage...SequenceChannel Property

NetIncomingMessage Class See Also Send Feedback

Gets the sequence channel this message was sent with (if user data)

Namespace: Lidgren.Network
## Syntax

**C#**

```csharp
public int SequenceChannel { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property SequenceChannel As Integer
    Get
        ...
    End Get
End Property
```

**Visual C++**

```cpp
public:
    property int SequenceChannel {
        int get ();
    }
```

See Also

NetIncomingMessage Class
Lidgren.Network Namespace
The type of a NetIncomingMessage

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public enum NetIncomingMessageType
```

**Visual Basic**

```vbnet
Public Enumeration NetIncomingMessageType
```

**Visual C++**

```cpp
public enum class NetIncomingMessageType
```
<table>
<thead>
<tr>
<th>Member name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td></td>
</tr>
<tr>
<td>StatusChanged</td>
<td></td>
</tr>
<tr>
<td>UnconnectedData</td>
<td></td>
</tr>
<tr>
<td>ConnectionApproval</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
</tr>
<tr>
<td>Receipt</td>
<td></td>
</tr>
<tr>
<td>DiscoveryRequest</td>
<td></td>
</tr>
<tr>
<td>DiscoveryResponse</td>
<td></td>
</tr>
<tr>
<td>VerboseDebugMessage</td>
<td></td>
</tr>
<tr>
<td>DebugMessage</td>
<td></td>
</tr>
<tr>
<td>WarningMessage</td>
<td></td>
</tr>
<tr>
<td>ErrorMessage</td>
<td></td>
</tr>
<tr>
<td>NatIntroductionSuccess</td>
<td></td>
</tr>
<tr>
<td>ConnectionLatencyUpdated</td>
<td></td>
</tr>
</tbody>
</table>
See Also

Lidgren.Network Namespace
Outgoing message used to send data to remote peer(s)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public sealed class NetOutgoingMessage : NetBuffer

Visual Basic

Public NotInheritable Class NetOutgoingMessage
    Inherits NetBuffer

Visual C++

public ref class NetOutgoingMessage sealed : public NetBuffer
Inheritance Hierarchy

System...Object
Lidgren.Network...NetBuffer
Lidgren.Network...NetOutgoingMessage
See Also

NetOutgoingMessage Members
Lidgren.Network Namespace
The **NetOutgoingMessage** type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encrypt</td>
<td>Encrypt this message using the provided algorithm; no more writing can be done before sending it or the message will be corrupt!</td>
</tr>
<tr>
<td>EnsureBufferSize</td>
<td>Ensures the buffer can hold this number of bits (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>PeekBoolean</td>
<td>Reads a 1-bit Boolean without advancing the read pointer  (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>PeekByte()0000</td>
<td>Reads a Byte without advancing the read pointer  (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td></td>
<td>Reads the specified number of bits into a Byte without advancing the read</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PeekByte(Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td>PeekBytes(Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td>PeekBytes(array[Byte][][], Int32, Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer</td>
</tr>
<tr>
<td>PeekDataBuffer</td>
<td>Gets the internal data buffer</td>
</tr>
<tr>
<td>PeekDouble</td>
<td>Reads a 64-bit Double without advancing the read pointer</td>
</tr>
<tr>
<td>PeekFloat</td>
<td>Reads a 32-bit Single without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt16</td>
<td>Reads an Int16 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt32(QQ)</td>
<td>Reads an Int32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads an Int32 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt64(QQ)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads an Int64 without advancing the read pointer</td>
</tr>
<tr>
<td>PeekSByte</td>
<td>Reads an SByte without advancing the read pointer</td>
</tr>
</tbody>
</table>
PeekSingle
Reads a 32-bit Single without advancing the read pointer
(Inherited from NetBuffer.)

PeekString
Reads a string without advancing the read pointer
(Inherited from NetBuffer.)

PeekUInt16
Reads a UInt16 without advancing the read pointer
(Inherited from NetBuffer.)

PeekUInt32
Reads a UInt32 without advancing the read pointer
(Inherited from NetBuffer.)

PeekUInt32(Int32)
Reads the specified number of bits into a UInt32 without advancing the read pointer
(Inherited from NetBuffer.)

PeekUInt64
Reads a UInt64 without advancing the read pointer
(Inherited from NetBuffer.)

PeekUInt64(Int32)
Reads the specified number of bits into an UInt64 without advancing the read pointer
(Inherited from NetBuffer.)

ReadAllFields(Object)
Reads all public and private declared instance fields of the object in alphabetical order using reflection
(Inherited from NetBuffer.)

ReadAllFields(Object, BindingFlags)
Reads all fields with the specified binding of the object in alphabetical order using reflection
(Inherited from NetBuffer.)

ReadAllProperties(Object)
Reads all public and private declared instance fields of the object in alphabetical order using reflection
(Inherited from NetBuffer.)
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadAllProperties</strong>(<em>Object, BindingFlags</em>)</td>
<td>binding of the object in alphabetical order using reflection (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadBits</strong></td>
<td>Reads the specified number of bits into a preallocated array (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadBoolean</strong></td>
<td>Reads a boolean value (stored as a single bit) written using Write(bool) (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadByte</strong>()()()()</td>
<td>Reads a byte (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadByte(Byte%)</strong></td>
<td>Reads a byte and returns true or false for success (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadByte(Int32)</strong></td>
<td>Reads 1 to 8 bits into a byte (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadBytes(Int32)</strong></td>
<td>Reads the specified number of bytes (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadBytes(Int32, array&lt;Byte&gt;[][])</strong></td>
<td>Reads the specified number of bytes and returns true for success (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadBytes(array&lt;Byte&gt;[][])[], Int32, Int32)</strong></td>
<td>Reads the specified number of bytes into a preallocated array (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadDouble</strong></td>
<td>Reads a 64 bit floating point value written using Write(Double) (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadFloat</strong></td>
<td>Reads a 32 bit floating point value written using Write(Single) (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadInt16</strong></td>
<td>Reads a 16 bit signed integer written using Write(Int16) (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td><strong>ReadInt32Q[]Q[]Q</strong></td>
<td>Reads a 32 bit signed integer written using Write(Int32) (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>ReadInt32(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 32 bits, written using <code>Write(Int32, Int32)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32%)</code></td>
<td>Reads a 32 bit signed integer written using <code>Write(Int32)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadInt64()</code></td>
<td>Reads a 64 bit signed integer written using <code>Write(Int64)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadInt64(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 64 bits, written using <code>Write(Int64, Int32)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadIPEndpoint</code></td>
<td>Reads a stored IPv4 endpoint description (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadPadBits</code></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadRangedInteger</code></td>
<td>Reads a 32 bit integer value written using <code>WriteRangedInteger()</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadRangedSingle</code></td>
<td>Reads a 32 bit floating point value written using <code>WriteRangedSingle()</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSByte</code></td>
<td>Reads a signed byte (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSignedSingle</code></td>
<td>Reads a 32 bit floating point value written using <code>WriteSignedSingle()</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSingle()</code></td>
<td>Reads a 32 bit floating point value written using <code>Write(Single)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSingle%</code></td>
<td>Reads a 32 bit floating point value written using <code>Write(Single)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
</tbody>
</table>
- **ReadString()**
  - Reads a string written using Write(string)
  - (Inherited from **NetBuffer**.)

- **ReadString(String%)**
  - Reads a string written using Write(string) and returns true for success
  - (Inherited from **NetBuffer**.)

- **ReadTime**
  - Reads a value, in local time comparable to NetTime.Now, written using WriteTime() for the connection supplied
  - (Inherited from **NetBuffer**.)

- **ReadUInt16**
  - Reads a 16 bit unsigned integer written using Write(UInt16)
  - (Inherited from **NetBuffer**.)

- **ReadUInt32()**
  - Reads an 32 bit unsigned integer written using Write(UInt32)
  - (Inherited from **NetBuffer**.)

- **ReadUInt32(Int32)**
  - Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32)
  - (Inherited from **NetBuffer**.)

- **ReadUInt32(UInt32%)**
  - Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success
  - (Inherited from **NetBuffer**.)

- **ReadUInt64()**
  - Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64)
  - (Inherited from **NetBuffer**.)

- **ReadUInt64(Int32)**
  - Reads a 64 bit unsigned integer written using Write(UInt64, Int32)
  - (Inherited from **NetBuffer**.)

- **ReadUnitSingle**
  - Reads a 32 bit floating point value written using WriteUnitSingle()
  - (Inherited from **NetBuffer**.)

- **Reads a variable sized Int32 written**
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadVariableInt32</td>
<td>using WriteVariableInt32() (Inherited from NetBuffer.) Reads a variable sized Int64 written using WriteVariableInt64() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadVariableInt64</td>
<td>using WriteVariableInt64() (Inherited from NetBuffer.) Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadVariableUInt32()</td>
<td>using WriteVariableUInt32() (Inherited from NetBuffer.) Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadVariableUInt64</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadVariableUInt32(UInt32%)</td>
<td>using WriteVariableUInt32() and returns true for success (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>SkipPadBits()</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>SkipPadBits(Int32)</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ToString</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Boolean)</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Byte)</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(array&lt;Byte&gt;[][])</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Double)</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Int16)</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Int32)</td>
<td>using WriteVariableUInt64() (Inherited from NetBuffer.) Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>Write(Int64)</td>
<td>Writes a 64 bit signed integer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(IPEndPoint)</td>
<td>Writes an endpoint description (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(SByte)</td>
<td>Writes a signed byte (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Single)</td>
<td>Writes a 32 bit floating point value (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(String)</td>
<td>Write a string (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt16)</td>
<td>Writes an unsigned 16 bit integer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt32)</td>
<td>Writes a 32 bit unsigned integer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt64)</td>
<td>Writes a 64 bit unsigned integer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(NetIncomingMessage)</td>
<td>Append all the bits of message to this message (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(NetOutgoingMessage)</td>
<td>Append all the bits of message to this message (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Byte, Int32)</td>
<td>Writes 1 to 8 bits of a byte (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Int32, Int32)</td>
<td>Writes a signed integer using 1 to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Int64, Int32)</td>
<td>Writes a signed integer using 1 to 64 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt16, Int32)</td>
<td>Writes an unsigned integer using 1 to 16 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt32, Int32)</td>
<td>Writes a 32 bit signed integer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>Write(UInt64, Int32)</code></td>
<td>Writes an unsigned integer using 1 to 64 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>Write(array&lt;Byte&gt;[][], Int32, Int32)</code></td>
<td>Writes the specified number of bytes from an array (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteAllFields(Object)</code></td>
<td>Writes all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteAllFields(Object, BindingFlags)</code></td>
<td>Writes all fields with specified binding in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteAllProperties(Object)</code></td>
<td>Writes all public and private declared instance properties of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteAllProperties(Object, BindingFlags)</code></td>
<td>Writes all properties with specified binding in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WritePadBits()</code></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WritePadBits(Int32)</code></td>
<td>Pads data with the specified number of bits. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteRangedInteger</code></td>
<td>Writes an integer with the least amount of bits need for the specified range</td>
</tr>
<tr>
<td><code>WriteRangedSingle</code></td>
<td>Compress (lossy) a float in the range</td>
</tr>
<tr>
<td><code>WriteRangedSingle</code></td>
<td>Compress a float within a specified range using a certain number of bits (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
- **WriteSignedSingle** -1..1 using `numberOfBits` bits
  (Inherited from [NetBuffer](#).)

  Writes the current local time to a message; readable (and convertable to local time) by the remote host using `ReadTime()`
  (Inherited from [NetBuffer](#).)

- **WriteTime(Boolean)**

  Writes a local timestamp to a message; readable (and convertable to local time) by the remote host using `ReadTime()`
  (Inherited from [NetBuffer](#).)

- **WriteTime(Double, Boolean)**

  Compress (lossy) a float in the range 0..1 using `numberOfBits` bits
  (Inherited from [NetBuffer](#).)

  Write Base128 encoded variable sized signed integer of up to 32 bits
  (Inherited from [NetBuffer](#).)

- **WriteVariableInt32**

  Write Base128 encoded variable sized signed integer of up to 64 bits
  (Inherited from [NetBuffer](#).)

- **WriteVariableInt64**

  Write Base128 encoded variable sized unsigned integer of up to 32 bits
  (Inherited from [NetBuffer](#).)

- **WriteVariableUInt32**

  Write Base128 encoded variable sized unsigned integer of up to 64 bits
  (Inherited from [NetBuffer](#).)
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>Gets or sets the internal data buffer (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>LengthBits</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bits (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>LengthBytes</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bytes (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>Position</strong></td>
<td>Gets or sets the read position in the buffer, in bits (not bytes) (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><strong>PositionInBytes</strong></td>
<td>Gets the position in the buffer in bytes; note that the bits of the first returned byte may already have been read - check the Position property to make sure. (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
Lidgren.Network Namespace
The `NetOutgoingMessage` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encrypt</td>
<td>Encrypt this message using the provided algorithm; no more writing can be done before sending it or the message will be corrupt!</td>
</tr>
<tr>
<td>EnsureBufferSize</td>
<td>Ensures the buffer can hold this number of bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>PeekBoolean</td>
<td>Reads a 1-bit Boolean without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte</td>
<td>Reads a Byte without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte(0000)</td>
<td>Reads the specified number of bits into a Byte without advancing the read</td>
</tr>
</tbody>
</table>

```csharp
// Example for PeekByte
byte[] data = new byte[4];

for (int i = 0; i < 4; i++)
{
    data[i] = PeekByte();
}
```
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekByte(Int32)</td>
<td>pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekBytes(Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekBytes(array&lt;Byte&gt;[][], Int32, Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekDataBuffer</td>
<td>Gets the internal data buffer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekDouble</td>
<td>Reads a 64-bit Double without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekFloat</td>
<td>Reads a 32-bit Single without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekInt16</td>
<td>Reads an Int16 without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekInt32([Int32])</td>
<td>Reads an Int32 without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads the specified number of bits into an Int32 without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekInt64([Int32])</td>
<td>Reads an Int64 without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>PeekSByte</td>
<td>Reads an SByte without advancing the read pointer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
</tbody>
</table>
- **PeekSingle**
  Reads a 32-bit Single without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekString**
  Reads a string without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt16**
  Reads a UInt16 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt32(0)**
  Reads a UInt32 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt32(Int32)**
  Reads the specified number of bits into a UInt32 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt64(0)**
  Reads a UInt64 without advancing the read pointer
  (Inherited from NetBuffer.)

- **PeekUInt64(Int32)**
  Reads the specified number of bits into an UInt64 without advancing the read pointer
  (Inherited from NetBuffer.)

- **ReadAllFields(Object)**
  Reads all public and private declared instance fields of the object in alphabetical order using reflection
  (Inherited from NetBuffer.)

- **ReadAllFields(Object, BindingFlags)**
  Reads all fields with the specified binding of the object in alphabetical order using reflection
  (Inherited from NetBuffer.)

- **ReadAllProperties(Object)**
  Reads all public and private declared instance fields of the object in alphabetical order using reflection
  (Inherited from NetBuffer.)

  Reads all fields with the specified
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadAllProperties(Object, BindingFlags)</strong></td>
<td>binding of the object in alphabetical order using reflection (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadBits</strong></td>
<td>Reads the specified number of bits into a preallocated array (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadBoolean</strong></td>
<td>Reads a boolean value (stored as a single bit) written using Write(bool) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadByte()</strong></td>
<td>Reads a byte (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadByte(Byte%)</strong></td>
<td>Reads a byte and returns true or false for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadByte(Int32)</strong></td>
<td>Reads 1 to 8 bits into a byte (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadBytes(Int32)</strong></td>
<td>Reads the specified number of bytes (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadBytes(Int32, array&lt;Byte&gt;[][])</strong></td>
<td>Reads the specified number of bytes and returns true for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadBytes(array&lt;Byte&gt;[][], Int32, Int32)</strong></td>
<td>Reads the specified number of bytes into a preallocated array (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadDouble</strong></td>
<td>Reads a 64 bit floating point value written using Write(Double) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadFloat</strong></td>
<td>Reads a 32 bit floating point value written using Write(Single) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadInt16</strong></td>
<td>Reads a 16 bit signed integer written using Write(Int16) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadInt32()</strong></td>
<td>Reads a 32 bit signed integer written using Write(Int32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>ReadInt32(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 32 bits, written using <code>Write(Int32, Int32)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadInt32(Int32%)</code></td>
<td>Reads a 32 bit signed integer written using <code>Write(Int32)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadInt64()</code></td>
<td>Reads a 64 bit signed integer written using <code>Write(Int64)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadInt64(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 64 bits, written using <code>Write(Int64, Int32)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadIPEndPoint</code></td>
<td>Reads a stored IPv4 endpoint description (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadPadBits</code></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadRangedInteger</code></td>
<td>Reads a 32 bit integer value written using <code>WriteRangedInteger()</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadRangedSingle</code></td>
<td>Reads a 32 bit floating point value written using <code>WriteRangedSingle()</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSByte</code></td>
<td>Reads a signed byte (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSignedSingle</code></td>
<td>Reads a 32 bit floating point value written using <code>WriteSignedSingle()</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSingle()</code></td>
<td>Reads a 32 bit floating point value written using <code>Write(Single)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadSingle(Single%)</code></td>
<td>Reads a 32 bit floating point value written using <code>Write(Single)</code> (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ReadString()()()</td>
<td>Reads a string written using Write(string) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadString(String%)</td>
<td>Reads a string written using Write(string) and returns true for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadTime</td>
<td>Reads a value, in local time comparable to NetTime.Now, written using WriteTime() for the connection supplied (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadUInt16</td>
<td>Reads a 16 bit unsigned integer written using Write(UInt16) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadUInt32()()()()</td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadUInt32(Int32)</td>
<td>Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadUInt32(UInt32%)</td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadUInt64()()()()()</td>
<td>Reads an unsigned integer stored in 1 to 64 bits, written using Write(UInt64, Int32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadUInt64(Int32)</td>
<td>Reads a 32 bit floating point value written using WriteUnitSingle() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadUnitSingle</td>
<td>Reads a variable sized Int32 written</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>ReadVariableInt32</strong></td>
<td>using WriteVariableInt32()  (Inherited from NetBuffer.) Reads a variable sized Int64 written using WriteVariableInt64()  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadVariableInt64</strong></td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32()  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadVariableUInt32()()()</strong></td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32() and returns true for success  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadVariableUInt32(UInt32%)</strong></td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt64()  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>SkipPadBits()()()</strong></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes.  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>SkipPadBits(Int32)</strong></td>
<td>Pads data with the specified number of bits.  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a string that represents this object  (Overrrides Object::ToString()()().)</td>
</tr>
<tr>
<td><strong>Write(Boolean)</strong></td>
<td>Writes a boolean value using 1 bit  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Byte)</strong></td>
<td>Write a byte  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(array&lt;Byte&gt;[][][])</strong></td>
<td>Writes all bytes in an array  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Double)</strong></td>
<td>Writes a 64 bit floating point value  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Int16)</strong></td>
<td>Writes a signed 16 bit integer  (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>Write(Int32)</strong></td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Write(Int64)</td>
<td>Writes a 64 bit signed integer</td>
</tr>
<tr>
<td>Write(IPEndPoint)</td>
<td>Writes an endpoint description</td>
</tr>
<tr>
<td>Write(SByte)</td>
<td>Writes a signed byte</td>
</tr>
<tr>
<td>Write(Single)</td>
<td>Writes a 32 bit floating point value</td>
</tr>
<tr>
<td>Write(String)</td>
<td>Write a string</td>
</tr>
<tr>
<td>Write(UInt16)</td>
<td>Writes an unsigned 16 bit integer</td>
</tr>
<tr>
<td>Write(UInt32)</td>
<td>Writes a 32 bit unsigned integer</td>
</tr>
<tr>
<td>Write(UInt64)</td>
<td>Writes a 64 bit unsigned integer</td>
</tr>
<tr>
<td>Write(NetIncomingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(NetOutgoingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Write(Byte, Int32)</td>
<td>Writes 1 to 8 bits of a byte</td>
</tr>
<tr>
<td>Write(Int32, Int32)</td>
<td>Writes a signed integer using 1 to 32 bits</td>
</tr>
<tr>
<td>Write(Int64, Int32)</td>
<td>Writes a signed integer using 1 to 64 bits</td>
</tr>
<tr>
<td>Write(UInt16, Int32)</td>
<td>Writes an unsigned integer using 1 to 16 bits</td>
</tr>
<tr>
<td>Write(UInt32, Int32)</td>
<td>Writes a 32 bit signed integer</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><code>Write(UInt64, Int32)</code></td>
<td>Writes an unsigned integer using 1 to 64 bits (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>Write(array&lt;Byte&gt;[][], Int32, Int32)</code></td>
<td>Writes the specified number of bytes from an array (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>WriteAllFields(Object)</code></td>
<td>Writes all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>WriteAllFields(Object, BindingFlags)</code></td>
<td>Writes all fields with specified binding in alphabetical order using reflection (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>WriteAllProperties(Object)</code></td>
<td>Writes all public and private declared instance properties of the object in alphabetical order using reflection (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>WriteAllProperties(Object, BindingFlags)</code></td>
<td>Writes all properties with specified binding in alphabetical order using reflection (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>WritePadBits()</code></td>
<td>Pads data with enough bits to reach a full byte. Decreases cpu usage for subsequent byte writes. (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>WritePadBits(Int32)</code></td>
<td>Pads data with the specified number of bits. (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>WriteRangedInteger</code></td>
<td>Writes an integer with the least amount of bits need for the specified range</td>
</tr>
<tr>
<td><code>WriteRangedSingle</code></td>
<td>Compress (lossy) a float in the range using a certain number of bits (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td><code>WriteSignedSingle</code></td>
<td>Writes a single number in the range -1..1 using <code>numberOfBits</code> bits. (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteTime(Boolean)</code></td>
<td>Writes the current local time to a message; readable (and convertible to local time) by the remote host using <code>ReadTime()</code> (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteTime(Double, Boolean)</code></td>
<td>Writes a local timestamp to a message; readable (and convertible to local time) by the remote host using <code>ReadTime()</code> (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteUnitSingle</code></td>
<td>Compress (lossy) a float in the range 0..1 using <code>numberOfBits</code> bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteVariableInt32</code></td>
<td>Write Base128 encoded variable sized signed integer of up to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteVariableInt64</code></td>
<td>Write Base128 encoded variable sized signed integer of up to 64 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteVariableUInt32</code></td>
<td>Write Base128 encoded variable sized unsigned integer of up to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><code>WriteVariableUInt64</code></td>
<td>Write Base128 encoded variable sized unsigned integer of up to 64 bits (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
Lidgren.Network Namespace
Encrypt this message using the provided algorithm; no more writing can be done before sending it or the message will be corrupt!

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool Encrypt(
    INetEncryption encryption
)
```

Visual Basic

```
Public Function Encrypt ( _
    encryption As INetEncryption _
) As Boolean
```

Visual C++

```cpp
public:
    bool Encrypt(
        INetEncryption^ encryption
    )
```

Parameters

encryption

Type: Lidgren.Network::INetEncryption

[Missing <param name="encryption"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetOutgoingMessage Class
Lidgren.Network Namespace
C# Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetOutgoingMessage:::PeekByte Method

NetOutgoingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekByte()()()</td>
<td>Reads a Byte without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekByte(Int32)</td>
<td>Reads the specified number of bits into a Byte without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekBytes(Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekBytes(array[Byte][][], Int32, Int32)</td>
<td>Reads the specified number of bytes without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetOutgoingMessage:::PeekInt32 Method

NetOutgoingMessage Class See Also Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekInt32</td>
<td>Reads an Int32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt32(Int32)</td>
<td>Reads the specified number of bits into an Int32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetOutgoingMessage::PeekInt64 Method

NetOutgoingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekInt64()()()</td>
<td>Reads an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekInt64(Int32)</td>
<td>Reads the specified number of bits into an Int64 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
NetOutgoingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekUInt32()()()</td>
<td>Reads a UInt32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>PeekUInt32(Int32)</td>
<td>Reads the specified number of bits into a UInt32 without advancing the read pointer (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetOutgoingMessage::PeekUInt64 Method
NetOutgoingMessage Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PeekUInt64()()()</td>
<td>Reads a UInt64 without advancing the read pointer</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td>PeekUInt64(Int32)</td>
<td>Reads the specified number of bits into an UInt64</td>
</tr>
<tr>
<td></td>
<td>without advancing the read pointer</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
C#  Visual Basic  Visual C++  Include Protected Members  Include Inherited Members  Lidgren Network Library documentation  NetOutgoingMessage:::ReadAllFields Method  NetOutgoingMessage Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| `ReadAllFields(Object)` | Reads all public and private declared instance fields of the object in alphabetical order using reflection  
(Inherited from `NetBuffer`.) |
| `ReadAllFields(Object, BindingFlags)` | Reads all fields with the specified binding of the object in alphabetical order using reflection  
(Inherited from `NetBuffer`.) |
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetOutgoingMessage::ReadAllProperties Method

NetOutgoingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadAllProperties(Object)</code></td>
<td>Reads all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadAllProperties(Object, BindingFlags)</code></td>
<td>Reads all fields with the specified binding of the object in alphabetical order using reflection (Inherited from <code>NetBuffer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
NetOutgoingMessage::ReadByte Method
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadByte()</code></td>
<td>Reads a byte (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>ReadByte(Byte%)</code></td>
<td>Reads a byte and returns true or false for success (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>ReadByte(Int32)</code></td>
<td>Reads 1 to 8 bits into a byte (Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadBytes(Int32)</td>
<td>Reads the specified number of bytes (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadBytes(Int32, array[Byte][])</td>
<td>Reads the specified number of bytes and returns true for success (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadBytes(array[Byte][][], Int32, Int32)</td>
<td>Reads the specified number of bytes into a preallocated array (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetOutgoingMessage::ReadInt32 Method

See Also

Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadInt32(000)</td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a> )</td>
</tr>
<tr>
<td>ReadInt32(Int32)</td>
<td>Reads a signed integer stored in 1 to 32 bits, written using Write(Int32, Int32)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a> )</td>
</tr>
<tr>
<td>ReadInt32(Int32%)</td>
<td>Reads a 32 bit signed integer written using Write(Int32)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a> )</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetOutgoingMessage

.NetOutgoingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadInt64()</code></td>
<td>Reads a 64 bit signed integer written using <code>Write(Int64)</code></td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td><code>ReadInt64(Int32)</code></td>
<td>Reads a signed integer stored in 1 to 64 bits, written using <code>Write(Int64, Int32)</code></td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>NetBuffer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadSingle()</td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>ReadSingle(S)</td>
<td>Reads a 32 bit floating point value written using Write(Single)</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadString()()()()</td>
<td>Reads a string written using <code>Write(string)</code></td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>NetBuffer</code>.)</td>
</tr>
<tr>
<td>ReadString(String%)</td>
<td>Reads a string written using <code>Write(string)</code> and returns true for success</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>NetBuffer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetOutgoingMessage::ReadUInt32 Method
NetOutgoingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ReadUInt32()</strong></td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadUInt32(Int32)</strong></td>
<td>Reads an unsigned integer stored in 1 to 32 bits, written using Write(UInt32, Int32) (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td><strong>ReadUInt32(UInt32%)</strong></td>
<td>Reads an 32 bit unsigned integer written using Write(UInt32) and returns true for success (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetOutgoingMessage::ReadUInt64 Method

NetOutgoingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ReadUInt64()</code></td>
<td>Reads a 64 bit unsigned integer written using <code>Write(UInt64)</code></td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
<tr>
<td><code>ReadUInt64(Int32)</code></td>
<td>Reads an unsigned integer stored in 1 to 64 bits, written using <code>Write(UInt64, Int32)</code></td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetBuffer</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
C# □ Visual Basic
□ Visual C++
□ Include Protected Members
□ Include Inherited Members
Lidgren Network Library documentation
NetOutgoingMessage:::ReadVariableUInt32 Method
NetOutgoingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReadVariableUInt32()</td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32()</td>
</tr>
<tr>
<td>(Inherited from NetBuffer.)</td>
<td></td>
</tr>
<tr>
<td>ReadVariableUInt32(UInt32)</td>
<td>Reads a variable sized UInt32 written using WriteVariableUInt32() and</td>
</tr>
<tr>
<td></td>
<td>returns true for success</td>
</tr>
<tr>
<td>(Inherited from NetBuffer.)</td>
<td></td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetOutgoingMessage::SkipPadBits Method

NetOutgoingMessage Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SkipPadBits()()()</td>
<td>Pads data with enough bits to reach a full byte.</td>
</tr>
<tr>
<td>(Inherited from NetBuffer.)</td>
<td></td>
</tr>
<tr>
<td>SkipPadBits(Int32)</td>
<td>Pads data with the specified number of bits.</td>
</tr>
<tr>
<td>(Inherited from NetBuffer.)</td>
<td></td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Returns a string that represents this object

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public override string ToString()
```

### Visual Basic

```vbnet
Public Overrides Function ToString As String
```

### Visual C++

```cpp
public:
virtual String^ ToString() override
```

## Return Value

See Also

NetOutgoingMessage Class
Lidgren.Network Namespace
NetOutgoingMessage Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write(Boolean)</td>
<td>Writes a boolean value using 1 bit (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(Byte)</td>
<td>Write a byte (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(array[Byte[]])</td>
<td>Writes all bytes in an array (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(Double)</td>
<td>Writes a 64 bit floating point value (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(Int16)</td>
<td>Writes a signed 16 bit integer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(Int32)</td>
<td>Writes a 32 bit signed integer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(Int64)</td>
<td>Writes a 64 bit signed integer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(IPEndPoint)</td>
<td>Writes an endpoint description (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(SByte)</td>
<td>Writes a signed byte (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(Single)</td>
<td>Writes a 32 bit floating point value (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(String)</td>
<td>Write a string (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(UInt16)</td>
<td>Writes an unsigned 16 bit integer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(UInt32)</td>
<td>Writes a 32 bit unsigned integer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(UInt64)</td>
<td>Writes a 64 bit unsigned integer (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td>Write(NetIncomingMessage)</td>
<td>Append all the bits of message to this message</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>Write(NetOutgoingMessage)</td>
<td>Append all the bits of message to this message (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Byte, Int32)</td>
<td>Writes 1 to 8 bits of a byte (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Int32, Int32)</td>
<td>Writes a signed integer using 1 to 32 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(Int64, Int32)</td>
<td>Writes a signed integer using 1 to 64 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt16, Int32)</td>
<td>Writes an unsigned integer using 1 to 16 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt32, Int32)</td>
<td>Writes a 32 bit signed integer (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(UInt64, Int32)</td>
<td>Writes an unsigned integer using 1 to 64 bits (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>Write(array&lt;Byte&gt;[][], Int32, Int32)</td>
<td>Writes the specified number of bytes from an array (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteAllFields(Object)</td>
<td>Writes all public and private declared instance fields of the object in alphabetical order using reflection (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
<tr>
<td>WriteAllFields(Object, BindingFlags)</td>
<td>Writes all fields with specified binding in alphabetical order using reflection (Inherited from <strong>NetBuffer</strong>.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WriteAllProperties(Object)</strong></td>
<td>Writes all public and private declared instance properties of the object in alphabetical order using reflection (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
<tr>
<td><strong>WriteAllProperties(Object, BindingFlags)</strong></td>
<td>Writes all properties with specified binding in alphabetical order using reflection (Inherited from <a href="#">NetBuffer</a>).</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetOutgoingMessage...WritePadBits Method

NetOutgoingMessage Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WritePadBits()()()()</td>
<td>Pads data with enough bits to reach a full byte.</td>
</tr>
<tr>
<td></td>
<td>Decreases cpu usage for subsequent byte writes.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WritePadBits(Int32)</td>
<td>Pads data with the specified number of bits.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetOutgoingMessage‥‥.WriteTime Method

NetOutgoingMessage Class See Also Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WriteTime(Boolean)</td>
<td>Writes the current local time to a message; readable (and convertible to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
<tr>
<td>WriteTime(Double, Boolean)</td>
<td>Writes a local timestamp to a message; readable (and convertible to local time) by the remote host using ReadTime() (Inherited from NetBuffer.)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
NetOutgoingMessage Members
Lidgren.Network Namespace
The `NetOutgoingMessage` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data</strong></td>
<td>Gets or sets the internal data buffer (Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
<tr>
<td><strong>LengthBits</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bits (Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
<tr>
<td><strong>LengthBytes</strong></td>
<td>Gets or sets the length of the used portion of the buffer in bytes (Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
<tr>
<td><strong>Position</strong></td>
<td>Gets or sets the read position in the buffer, in bits (not bytes) (Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
<tr>
<td><strong>PositionInBytes</strong></td>
<td>Gets the position in the buffer in bytes; note that the bits of the first returned byte may already have been read - check the Position property to make sure. (Inherited from <a href="#">NetBuffer</a>)</td>
</tr>
</tbody>
</table>
See Also

NetOutgoingMessage Class
Lidgren.Network Namespace
Represents a local peer capable of holding zero, one or more connections to remote peers

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public class NetPeer
```

**Visual Basic**

```vbnet
Public Class NetPeer
```

**Visual C++**

```cpp
public ref class NetPeer
```
Inheritance Hierarchy

System:::Object
Lidgren.Network:::NetPeer
  Lidgren.Network:::NetClient
  Lidgren.Network:::NetServer
See Also

NetPeer Members
Lidgren.Network Namespace
The `NetPeer` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetPeer</td>
<td>NetPeer constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Connect(IPEndPoint)</strong></td>
<td>Create a connection to a remote endpoint.</td>
</tr>
<tr>
<td><strong>Connect(IPEndPoint, NetOutgoingMessage)</strong></td>
<td>Create a connection to a remote endpoint.</td>
</tr>
<tr>
<td><strong>Connect(String, Int32)</strong></td>
<td>Create a connection to a remote endpoint.</td>
</tr>
<tr>
<td><strong>Connect(String, Int32, NetOutgoingMessage)</strong></td>
<td>Create a connection to a remote endpoint.</td>
</tr>
<tr>
<td><strong>CreateMessage()</strong></td>
<td>Creates a new message for sending.</td>
</tr>
<tr>
<td><strong>CreateMessage(Int32)</strong></td>
<td>Creates a new message for sending.</td>
</tr>
<tr>
<td><strong>CreateMessage(String)</strong></td>
<td>Creates a new message for sending and writes the provided string to it.</td>
</tr>
<tr>
<td><strong>DiscoverKnownPeer(IPEndPoint)</strong></td>
<td>Emit a discovery signal to a single known host.</td>
</tr>
<tr>
<td><strong>DiscoverKnownPeer(String, Int32)</strong></td>
<td>Emit a discovery signal to a single known host.</td>
</tr>
<tr>
<td><strong>DiscoverLocalPeers</strong></td>
<td>Emit a discovery signal to all hosts on your subnet.</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified object is equal to the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to perform cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>FlushSendQueue</strong></td>
<td>If <strong>NetPeerConfiguration.AutoFlushSendQueue()</strong> is false; you need to call this to send all messages queued using <strong>Send().</strong></td>
</tr>
<tr>
<td><strong>GetConnection</strong></td>
<td>Get the connection, if any, for the remote endpoint.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for the current <strong>Object</strong>.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current <strong>Object</strong>.</td>
</tr>
</tbody>
</table>
**Introduce**

**MemberwiseClone**

**RawSend**

**ReadMessage**

**ReadMessages**

**Recycle(NetIncomingMessage)**

**Recycle(IEnumerable<NetIncomingMessage>)**

**RegisterReceivedCallback**

**SendDiscoveryResponse**

**SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)**

**SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)**

**SendMessage(NetOutgoingMessage, List<NetConnection>, NetDeliveryMethod, Int32)**

**SendUnconnectedMessage(NetOutgoingMessage, IList<IPEndPoint>)**

**SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)**

**SendUnconnectedMessage(NetOutgoingMessage, String, Int32)**

**SendUnconnectedToSelf**

**Shutdown**

(Inherited from **Object**) Send NetIntroduction to host clientExternal; introducing (Inherited from **Object**) Creates a shallow copy of the (Inherited from **Object**) Send raw bytes; only used for Read a pending message from connection, if any Read a pending message from connection, if any Recycles a NetIncomingMessage instances for reuse; taking pressure off the garbage collector Call this to register a callback message arrives Send a discovery response Send a message to a specific Send a message to a specific Send a message to a list of connections Send a message to an unconnected Send a message to an unconnected Send a message to an unconnected host Send a message to this exact (loopback) Disconnects all active connections the socket
Start

ToString

WaitMessage

Binds to socket and spawns thread
Returns a String that represents the current Object.
(Inherited from Object.)
Read a pending message from connection, blocking up to maxMillis if needed
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration</strong></td>
<td>Gets the configuration used to instanciate this NetPeer</td>
</tr>
<tr>
<td><strong>Connections</strong></td>
<td>Gets a copy of the list of connections</td>
</tr>
<tr>
<td><strong>ConnectionsCount</strong></td>
<td>Gets the number of active connections</td>
</tr>
<tr>
<td><strong>MessageReceivedEvent</strong></td>
<td>Signalling event which can be waited on to determine when a message is queued for reading. Note that there is no guarantee that after the event is signaled the blocked thread will find the message in the queue. Other user created threads could be preempted and dequeue the message before the waiting thread wakes up.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Gets the port number this NetPeer is listening and sending on, if Start() has been called</td>
</tr>
<tr>
<td><strong>Socket</strong></td>
<td>Gets the socket, if Start() has been called</td>
</tr>
<tr>
<td><strong>Statistics</strong></td>
<td>Statistics on this NetPeer since it was initialized</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Gets the NetPeerStatus of the NetPeer</td>
</tr>
<tr>
<td><strong>Tag</strong></td>
<td>Gets or sets the application defined object containing data about the peer</td>
</tr>
<tr>
<td><strong>UniqueIdentifier</strong></td>
<td>Gets a unique identifier for this NetPeer based on Mac address and ip/port. Note! Not available until Start() has been called!</td>
</tr>
<tr>
<td><strong>UPnP</strong></td>
<td>Returns an UPnP object if enabled in the NetPeerConfiguration</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
Lidgren.Network Namespace
NetPeer constructor

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public NetPeer(
    NetPeerConfiguration config
)
```

Visual Basic

```vbnet
Public Sub New (_
    config As NetPeerConfiguration _
)
```

Visual C++

```cpp
public:
NetPeer(
    NetPeerConfiguration^ config
)
```

Parameters

config

Type: Lidgren.Network.NetPeerConfiguration

See Also

NetPeer Class
Lidgren.Network Namespace
The `NetPeer` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Connect(IPEndPoint)</strong></td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td><strong>Connect(IPEndPoint, NetOutgoingMessage)</strong></td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td><strong>Connect(String, Int32)</strong></td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td><strong>Connect(String, Int32, NetOutgoingMessage)</strong></td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td><strong>CreateMessage()</strong></td>
<td>Creates a new message for sending</td>
</tr>
<tr>
<td><strong>CreateMessage(Int32)</strong></td>
<td>Creates a new message for sending</td>
</tr>
<tr>
<td><strong>CreateMessage(String)</strong></td>
<td>Creates a new message for sending and writes the provided string to it</td>
</tr>
<tr>
<td><strong>DiscoverKnownPeer(IPEndPoint)</strong></td>
<td>Emit a discovery signal to a single known host</td>
</tr>
<tr>
<td><strong>DiscoverKnownPeer(String, Int32)</strong></td>
<td>Emit a discovery signal to a single known host</td>
</tr>
<tr>
<td><strong>DiscoverLocalPeers</strong></td>
<td>Emit a discovery signal to all hosts on your subnet</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified object is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to release any resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>FlushSendQueue</strong></td>
<td>If <code>NetPeerConfiguration.AutoFlushSendQueue()</code> is false; you need to call this to send all messages queued using <code>SendMessage()</code></td>
</tr>
<tr>
<td><strong>GetConnection</strong></td>
<td>Get the connection, if any, for the remote endpoint</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a <code>Type</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current object</td>
</tr>
</tbody>
</table>


**Introduce**  
Send NetIntroduction to host clientExternal; introducing

**MemberwiseClone**  
Creates a shallow copy of the current
(Inherited from **Object**.)

**RawSend**  
Send raw bytes; only used for debugging

**ReadMessage**  
Read a pending message from connection, if any

**ReadMessages**  
Read a pending message from connection, if any

**Recycle(NetIncomingMessage)**  
Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector

**Recycle(IEnumerable<NetIncomingMessage>)**  
Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector

**RegisterReceivedCallback**  
Call this to register a callback for when a new message arrives

**SendDiscoveryResponse**  
Send a discovery response message

**SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)**  
Send a message to a specific connection

**SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)**  
Send a message to a specific connection

**SendMessage(NetOutgoingMessage, List<NetConnection>, NetDeliveryMethod, Int32)**  
Send a message to a list of connections

**SendUnconnectedMessage(NetOutgoingMessage, IList<IPEndPoint>)**  
Send a message to an unconnected host (loopback)

**SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)**  
Send a message to an unconnected host

**SendUnconnectedMessage(NetOutgoingMessage, String, Int32)**  
Send a message to this exact host

**SendUnconnectedToSelf**  
Send a message to this exact netpeer (loopback)

**Shutdown**  
Disconnects all active connections and closes the socket
- **Start**

  Binds to socket and spawns thread

- **ToString**

  Returns a `String` that represents the current `Object`.
  (Inherited from `Object`.)

- **WaitMessage**

  Read a pending message from connection, blocking up to maxMillis if needed
See Also

NetPeer Class
Lidgren.Network Namespace
C# Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetPeer::Connect Method
NetPeer Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect(IPEndPoint)</td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td>Connect(IPEndPoint, NetOutgoingMessage)</td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td>Connect(String, Int32)</td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td>Connect(String, Int32, NetOutgoingMessage)</td>
<td>Create a connection to a remote endpoint</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
NetPeer Members
Lidgren.Network Namespace
Create a connection to a remote endpoint

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public NetConnection Connect(
    IPEndPoint remoteEndpoint
)
```

Visual Basic

```vbnet
Public Function Connect (_
    remoteEndpoint As IPEndPoint _
) As NetConnection
```

Visual C++

```cpp
public:
    NetConnection^ Connect(
        IPEndPoint^ remoteEndpoint
    )
```

Parameters

remoteEndpoint
Type: System.Net::IPEndPoint


Return Value

See Also

NetPeer Class
Connect Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer..........Connect Method (IPEndPoint, NetOutgoingMessage)

NetPeer Class  See Also  Send Feedback

Create a connection to a remote endpoint

Namespace: Lidgren.Network
(2012.1.7.0)
Syntax

C#

```csharp
public virtual NetConnection Connect(IPEndPoint remoteEndpoint, NetOutgoingMessage hailMessage)
```

Visual Basic

```vbnet
Public Overridable Function Connect(_
    remoteEndpoint As IPEndPoint, _
    hailMessage As NetOutgoingMessage _) As NetConnection
```

Visual C++

```cpp
public:
    virtual NetConnection^ Connect(IPEndPoint^ remoteEndpoint, NetOutgoingMessage^ hailMessage)
```

Parameters

remoteEndpoint
   Type: System.Net:::IPEndPoint


hailMessage
   Type: Lidgren.Network:::NetOutgoingMessage


Return Value

}
See Also

NetPeer Class
Connect Overload
Lidgren.Network Namespace
Create a connection to a remote endpoint

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetConnection Connect(
    string host,
    int port
)

Visual Basic

Public Function Connect (_
    host As String, _
    port As Integer _
) As NetConnection

Visual C++

public:
NetConnection^ Connect(
    String^ host,
    int port
)

Parameters

host

Type: System::String

[port documentation for

port

Type: System::Int32

[port documentation for

Return Value
See Also

NetPeer Class
Connect Overload
Lidgren.Network Namespace
Create a connection to a remote endpoint

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public NetConnection Connect(
    string host,
    int port,
    NetOutgoingMessage hailMessage
)
```

### Visual Basic

```vbnet
Public Function Connect (_
    host As String, _
    port As Integer, _
    hailMessage As NetOutgoingMessage _
) As NetConnection
```

### Visual C++

```cpp
public:
    NetConnection^ Connect( 
        String^ host, 
        int port, 
        NetOutgoingMessage^ hailMessage 
    )
```

### Parameters

**host**
- Type: `System..........String`
  

**port**
- Type: `System..........Int32`
  
Type: Lidgren.Network.NetOutgoingMessage

[Missing `<param name="hailMessage"/>` documentation for

Return Value

[Missing `<returns>` documentation for
See Also

NetPeer Class
Connect Overload
Lidgren.Network Namespace
C# Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetPeer CreateMessage Method
NetPeer Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>CreateMessage()</code>()()()()</td>
<td>Creates a new message for sending</td>
</tr>
<tr>
<td><code>CreateMessage(Int32)</code></td>
<td>Creates a new message for sending</td>
</tr>
<tr>
<td><code>CreateMessage(String)</code></td>
<td>Creates a new message for sending and writes the provided string to it</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
NetPeer Members
Lidgren.Network Namespace
CreateMessage Method

NetPeer Class See Also Send Feedback

Creates a new message for sending

Namespace: Lidgren.Network
Syntax

C#

public NetOutgoingMessage CreateMessage()

Visual Basic

Public Function CreateMessage As NetOutgoingMessage

Visual C++

public: NetOutgoingMessage^ CreateMessage()

Return Value

See Also

NetPeer Class
CreateMessage Overload
Lidgren.Network Namespace
CreateMessage Method (Int32)

NetPeer Class See Also Send Feedback

Creates a new message for sending

Namespace: Lidgren.Network
Syntax

C#

public NetOutgoingMessage CreateMessage(int initialCapacity)

Visual Basic

Public Function CreateMessage(initialCapacity As Integer) As NetOutgoingMessage

Visual C++

public: NetOutgoingMessage^ CreateMessage(int initialCapacity)

Parameters

initialCapacity
    Type: System::Int32
    initial capacity in bytes

Return Value

See Also

NetPeer Class
CreateMessage Overload
Lidgren.Network Namespace
CreateMessage Method (String)

Creates a new message for sending and writes the provided string to it

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
### Syntax

**C#**

```csharp
public NetOutgoingMessage CreateMessage(
    string content
)
```

**Visual Basic**

```vbnet
Public Function CreateMessage ( _
    content As String _
) As NetOutgoingMessage
```

**Visual C++**

```cpp
public:
NetOutgoingMessage^ CreateMessage( _
    String^ content
)
```

### Parameters

**content**

Type: `System::String`

[Missing `<param name="content"/>` documentation for

### Return Value

[Missing `<returns>` documentation for
See Also

NetPeer Class
CreateMessage Overload
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>DiscoverKnownPeer(IPEndPoint)</code></td>
<td>Emit a discovery signal to a single known host</td>
</tr>
<tr>
<td><code>DiscoverKnownPeer(String, Int32)</code></td>
<td>Emit a discovery signal to a single known host</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
NetPeer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeer..:.:.DiscoverKnownPeer Method (IPEndPoint)

NetPeer Class  See Also  Send Feedback

Emit a discovery signal to a single known host

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void DiscoverKnownPeer(
    IPEndPoint endpoint
)
```

**Visual Basic**

```vbnet
Public Sub DiscoverKnownPeer ( _
    endpoint As IPEndPoint _
)
```

**Visual C++**

```cpp
public:
void DiscoverKnownPeer(
    IPEndPoint^ endpoint
)
```

Parameters

`endpoint`

Type: `System.Net...IPEndPoint`

See Also

NetPeer Class
DiscoverKnownPeer Overload
Lidgren.Network Namespace
Emit a discovery signal to a single known host

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool DiscoverKnownPeer(
    string host,
    int serverPort
)
```

Visual Basic

```vbnet
Public Function DiscoverKnownPeer (
    host As String,
    serverPort As Integer _
) As Boolean
```

Visual C++

```cpp
public:
    bool DiscoverKnownPeer(
        String^ host,
        int serverPort
    )
```

Parameters

host

Type: System::String


serverPort

Type: System::Int32


Return Value
See Also

NetPeer Class
DiscoverKnownPeer Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeer..::..DiscoverLocalPeers Method

NetPeer Class  See Also  Send Feedback

Emit a discovery signal to all hosts on your subnet

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public void DiscoverLocalPeers(
    int serverPort
)
```

**Visual Basic**

```vbnet
Public Sub DiscoverLocalPeers ( _
    serverPort As Integer _
)
```

**Visual C++**

```cpp
public:
void DiscoverLocalPeers(
    int serverPort
)
```

Parameters

serverPort

Type: `System::::Int32`

See Also

NetPeer Class
Lidgren.Network Namespace
If `NetPeerConfiguration.AutoFlushSendQueue()` is false; you need to call this to send all messages queued using `SendMessage()`

**Namespace:** [Lidgren.Network](https://www.lidgrennetwork.com)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void FlushSendQueue()

Visual Basic

Public Sub FlushSendQueue

Visual C++

public:
void FlushSendQueue()
See Also

NetPeer Class
Lidgren.Network Namespace
Get the connection, if any, for a certain remote endpoint

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public NetConnection GetConnection(IPEndPoint ep)
```

Visual Basic

```vbnet
Public Function GetConnection(ep As IPEndPoint) As NetConnection
```

Visual C++

```cpp
public: NetConnection^ GetConnection(IPEndPoint^ ep)
```

Parameters

ep

Type: `System.Net.IPEndPoint`


Return Value

See Also

NetPeer Class
Lidgren.Network Namespace
Send NetIntroduction to hostExternal and clientExternal; introducing client to host

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Introduce(
    IPEndPoint hostInternal,
    IPEndPoint hostExternal,
    IPEndPoint clientInternal,
    IPEndPoint clientExternal,
    string token
)

Visual Basic

Public Sub Introduce (_
    hostInternal As IPEndPoint, _
    hostExternal As IPEndPoint, _
    clientInternal As IPEndPoint, _
    clientExternal As IPEndPoint, _
    token As String _
)

Visual C++

public:
    void Introduce(
        IPEndPoint^ hostInternal,
        IPEndPoint^ hostExternal,
        IPEndPoint^ clientInternal,
        IPEndPoint^ clientExternal,
        String^ token
    )

Parameters

hostInternal
Type: System.Net.IPAddress

[Missing <param name="hostInternal"/> documentation for

Parameters

hostInternal
Type: System.Net.IPAddress

[Missing <param name="hostInternal"/> documentation for

Parameters

hostInternal
Type: System.Net.IPAddress

[ Missing <param name="hostInternal"/> documentation for
hostExternal
Type: System.Net.***.IPEndPoint


clientInternal
Type: System.Net.***.IPEndPoint


clientExternal
Type: System.Net.***.IPEndPoint


token
Type: System.***.String

See Also

NetPeer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer...::RawSend Method

**NetPeer Class**  See Also  Send Feedback

Send raw bytes; only used for debugging

**Namespace:**  [Lidgren.Network](#)

**Assembly:**  Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
# Syntax

## C#

```csharp
public void RawSend(
    byte[] arr,
    int offset,
    int length,
    IPEndPoint destination
)
```

## Visual Basic

```vbnet
Public Sub RawSend (_
    arr As Byte(), _
    offset As Integer, _
    length As Integer, _
    destination As IPEndPoint _
)
```

## Visual C++

```cpp
public:
void RawSend(
    array<unsigned char>>^ arr,
    int offset,
    int length,
    IPEndPoint^ destination
)
```

## Parameters

**arr**

Type: `array<System::::Byte>[]`  


**offset**

Type: `System::::Int32`
length
Type: System.Int32

destination
Type: System.Net.IPEndPoint
See Also

NetPeer Class
Lidgren.Network Namespace
Read a pending message from any connection, if any

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetIncomingMessage ReadMessage()

Visual Basic

Public Function ReadMessage As NetIncomingMessage

Visual C++

public: 
NetIncomingMessage^ ReadMessage()

Return Value

**See Also**

NetPeer Class
Lidgren.Network Namespace
Read a pending message from any connection, if any

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int ReadMessages(  
    IList<NetIncomingMessage> addTo  
)  

Visual Basic

Public Function ReadMessages ( _  
    addTo As IList(Of NetIncomingMessage) _  
) As Integer  

Visual C++

public:  
    int ReadMessages(  
        IList<NetIncomingMessage>^ addTo  
    )  

Parameters

addTo  
Type: System.Collections.Generic..::.IList<(Of <(NetIncomingMessage)>)>  

[Missing <param name="addTo"/> documentation for  

Return Value

[Missing <returns> documentation for  
See Also

NetPeer Class
Lidgren.Network Namespace
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recycle(NetIncomingMessage)</td>
<td>Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector</td>
</tr>
<tr>
<td>Recycle(IEnumerable&lt;NetIncomingMessage&gt;)</td>
<td>Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
NetPeer Members
Lidgren.Network Namespace
Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void Recycle(
    NetIncomingMessage msg
)
```

Visual Basic

```
Public Sub Recycle ( _
    msg As NetIncomingMessage _
)
```

Visual C++

```cpp
public:
void Recycle(
    NetIncomingMessage^ msg
)
```

Parameters

msg
  Type: Lidgren.Network.NetIncomingMessage
See Also

NetPeer Class
Recycle Overload
Lidgren.Network Namespace
Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public void Recycle(
    IEnumerable<NetIncomingMessage> toRecycle
)
```

### Visual Basic

```vbnet
Public Sub Recycle ( _
    toRecycle As IEnumerable(Of NetIncomingMessage) _
)
```

### Visual C++

```cpp
public:
    void Recycle(
        IEnumerable<NetIncomingMessage>^ toRecycle
    )
```

## Parameters

**toRecycle**

Type: `System.Collections.Generic(IEnumerable<NetIncomingMessage>)`  

See Also

NetPeer Class
Recycle Overload
Lidgren.Network Namespace
Call this to register a callback for when a new message arrives

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void RegisterReceivedCallback(
    SendOrPostCallback callback
)
```

Visual Basic

```vbnet
Public Sub RegisterReceivedCallback (_
    callback As SendOrPostCallback _
)
```

Visual C++

```cpp
public:
void RegisterReceivedCallback(
    SendOrPostCallback^ callback
)
```

Parameters

callback

Type: System.Threading.SendOrPostCallback

[Missing <param name="callback"/> documentation for "M:Lidgren.Network.NetPeer.RegisterReceivedCallback(System.Threading.SendOrPostCallback)".]
See Also

NetPeer Class
Lidgren.Network Namespace
Send a discovery response message

**Namespace:** [Lidgren.Network](#)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public void SendDiscoveryResponse(
    NetOutgoingMessage msg,
    IPEndPoint recipient
)
```

**Visual Basic**

```vbnet
Public Sub SendDiscoveryResponse (_
    msg As NetOutgoingMessage, _
    recipient As IPEndPoint _
)
```

**Visual C++**

```cpp
public:
void SendDiscoveryResponse(
    NetOutgoingMessage^ msg,
    IPEndPoint^ recipient
)
```

**Parameters**

**msg**

Type: Lidgren.Network,:..,:NetOutgoingMessage

[Missing <param name="msg"/> documentation for

**recipient**

Type: System.Net,:..,:IPEndPoint

[Missing <param name="recipient"/> documentation for
See Also

NetPeer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer::SendMessage Method

NetPeer Class  See Also  Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)</code></td>
<td>Send a message to a specific connection</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)</code></td>
<td>Send a message to a specific connection</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, List&lt;Of &lt;'(NetConnection&gt;')&gt;&gt;, NetDeliveryMethod, Int32)</code></td>
<td>Send a message to a list of connections</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
NetPeer Members
Lidgren.Network Namespace
Send a message to a specific connection

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetSendResult SendMessage(
    NetOutgoingMessage msg,
    NetConnection recipient,
    NetDeliveryMethod method
)

Visual Basic

Public Function SendMessage ( _
    msg As NetOutgoingMessage, _
    recipient As NetConnection, _
    method As NetDeliveryMethod _
) As NetSendResult

Visual C++

public:
NetSendResult SendMessage(
    NetOutgoingMessage^ msg,
    NetConnection^ recipient,
    NetDeliveryMethod method
)

Parameters

msg
    Type: Lidgren.Network::::NetOutgoingMessage
    The message to send

recipient
    Type: Lidgren.Network::::NetConnection
    The recipient connection

method
    Type: Lidgren.Network::::NetDeliveryMethod
How to deliver the message

Return Value

[Missing <returns> documentation for
]
See Also

NetPeer Class
SendMessage Overload
Lidgren.Network Namespace
Lidgren Network Library documentation


NetPeer Class See Also Send Feedback

Send a message to a specific connection

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public NetSendResult SendMessage(
    NetOutgoingMessage msg,
    NetConnection recipient,
    NetDeliveryMethod method,
    int sequenceChannel)
```

**Visual Basic**

```vbnet
Public Function SendMessage ( 
    msg As NetOutgoingMessage, _
    recipient As NetConnection, _
    method As NetDeliveryMethod, _
    sequenceChannel As Integer _
) As NetSendResult
```

**Visual C++**

```cpp
public:
    NetSendResult SendMessage(
        NetOutgoingMessage^ msg,
        NetConnection^ recipient,
        NetDeliveryMethod method,
        int sequenceChannel
    )
```

**Parameters**

- **msg**
  Type: `Lidgren.Network...NetOutgoingMessage`
  The message to send

- **recipient**
  Type: `Lidgren.Network...NetConnection`
  The recipient connection
method
  Type: Lidgren.Network..., NetDeliveryMethod
  How to deliver the message

sequenceChannel
  Type: System..., Int32
  Sequence channel within the delivery method

Return Value

[Missing <returns> documentation for
]
See Also

NetPeer Class
SendMessage Overload
Lidgren.Network Namespace
Lidgren Network Library documentation


NetPeer Class See Also Send Feedback

Send a message to a list of connections

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public void SendMessage(
    NetOutgoingMessage msg,
    List<NetConnection> recipients,
    NetDeliveryMethod method,
    int sequenceChannel
)
```

### Visual Basic

```vbnet
Public Sub SendMessage (_
    msg As NetOutgoingMessage, _
    recipients As List(Of NetConnection), _
    method As NetDeliveryMethod, _
    sequenceChannel As Integer _
)
```

### Visual C++

```cpp
public:
void SendMessage(
    NetOutgoingMessage^ msg,
    List<NetConnection>^ recipients,
    NetDeliveryMethod method,
    int sequenceChannel
)
```

## Parameters

**msg**

Type: [Lidgren.Network..::.NetOutgoingMessage](#)

The message to send

**recipients**

Type: [System.Collections.Generic..::.List](#)

The list of recipients to send to
method
  Type: Lidgren.Network..NetDeliveryMethod
  How to deliver the message

sequenceChannel
  Type: System..Int32
  Sequence channel within the delivery method
See Also

NetPeer Class
SendMessage Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer::SendUnconnectedMessage Method
NetPeer Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SendUnconnectedMessage(NetOutgoingMessage, IList&lt;Of &lt;&lt;'(IPEndPoint&gt;)&gt;&gt;))</td>
<td>Send a message to an unconnected host</td>
</tr>
<tr>
<td>SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)</td>
<td>Send a message to an unconnected host</td>
</tr>
<tr>
<td>SendUnconnectedMessage(NetOutgoingMessage, String, Int32)</td>
<td>Send a message to an unconnected host</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
NetPeer Members
Lidgren.Network Namespace
Send a message to an unconnected host

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void SendUnconnectedMessage(
    NetOutgoingMessage msg,
    IList<IPEndPoint> recipients
)

Visual Basic

Public Sub SendUnconnectedMessage ( _
    msg As NetOutgoingMessage, _
    recipients As IList(Of IPEndPoint) _
)

Visual C++

public:
void SendUnconnectedMessage(
    NetOutgoingMessage^ msg,
    IList<IPEndPoint^[^ recipients

Parameters

msg
Type: Lidgren.Network::NetOutgoingMessage

[Missing <param name="msg"/> documentation for

recipients
Type: System.Collections.Generic::IList<Of (<IPEndPoint>)>)

[Missing <param name="recipients"/> documentation for
See Also

NetPeer Class
SendUnconnectedMessage Overload
Lidgren.Network Namespace
Send a message to an unconnected host

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void SendUnconnectedMessage(
        NetOutgoingMessage msg,
        IPEndPoint recipient
    )

Visual Basic

Public Sub SendUnconnectedMessage ( _
        msg As NetOutgoingMessage, _
        recipient As IPEndPoint _
    )

Visual C++

public:
void SendUnconnectedMessage(
        NetOutgoingMessage^ msg,
        IPEndPoint^ recipient
    )

Parameters

msg
    Type: Lidgren.Network.NetOutgoingMessage

recipient
    Type: System.Net.IPEndPoint


See Also

NetPeer Class
SendUnconnectedMessage Overload
Lidgren.Network Namespace
Send a message to an unconnected host

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void SendUnconnectedMessage(
    NetOutgoingMessage msg,
    string host,
    int port
)

Visual Basic

Public Sub SendUnconnectedMessage ( _
    msg As NetOutgoingMessage, _
    host As String, _
    port As Integer _
)

Visual C++

public:
void SendUnconnectedMessage(    
    NetOutgoingMessage^ msg,
    String^ host,
    int port
)

Parameters

msg
   Type: Lidgren.Network.NetOutgoingMessage

[Missing <param name="msg"/> documentation for 

host
   Type: System.String

[Missing <param name="host"/> documentation for 
port

Type: System..::.Int32

[Missing <param name="port"/> documentation for
See Also

NetPeer Class
SendUnconnectedMessage Overload
Lidgren.Network Namespace
Send a message to this exact same netpeer (loopback)

**Namespace:** [Lidgren.Network](https://example.com)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public void SendUnconnectedToSelf(
    NetOutgoingMessage msg
)
```

### Visual Basic

```vbnet
Public Sub SendUnconnectedToSelf ( _
    msg As NetOutgoingMessage _
)
```

### Visual C++

```cpp
public:
void SendUnconnectedToSelf(
    NetOutgoingMessage^ msg
)
```

## Parameters

- **msg**

See Also

NetPeer Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeer..::..Shutdown Method

NetPeer Class See Also Send Feedback

Disconnects all active connections and closes the socket

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public void Shutdown(  
    string bye
)
```

**Visual Basic**

```vbnet
Public Sub Shutdown ( _
    bye As String _
)
```

**Visual C++**

```cpp
public:
void Shutdown(  
    String^ bye
)
```

**Parameters**

`bye`

Type: `System.String`

See Also

NetPeer Class
Lidgren.Network Namespace
Binds to socket and spawns the networking thread

**Namespace**: [Lidgren.Network](#)
Syntax

C#

public void Start()

Visual Basic

Public Sub Start

Visual C++

public:
void Start()
See Also

NetPeer Class
Lidgren.Network Namespace
Read a pending message from any connection, blocking up to maxMillis if needed

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetIncomingMessage WaitMessage(int maxMillis)

Visual Basic

Public Function WaitMessage(_
    maxMillis As Integer _
) As NetIncomingMessage

Visual C++

public:
NetIncomingMessage^ WaitMessage(_
    int maxMillis
)

Parameters

maxMillis
Type: System::::Int32


Return Value

See Also

NetPeer Class
Lidgren.Network Namespace
The **NetPeer** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration</strong></td>
<td>Gets the configuration used to instanciate this NetPeer</td>
</tr>
<tr>
<td><strong>Connections</strong></td>
<td>Gets a copy of the list of connections</td>
</tr>
<tr>
<td><strong>ConnectionsCount</strong></td>
<td>Gets the number of active connections</td>
</tr>
<tr>
<td><strong>MessageReceivedEvent</strong></td>
<td>Signalling event which can be waited on to determine when a message is queued for reading. Note that there is no guarantee that after the event is signaled the blocked thread will find the message in the queue. Other user created threads could be preempted and dequeue the message before the waiting thread wakes up.</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Gets the port number this NetPeer is listening and sending on, if Start() has been called</td>
</tr>
<tr>
<td><strong>Socket</strong></td>
<td>Gets the socket, if Start() has been called</td>
</tr>
<tr>
<td><strong>Statistics</strong></td>
<td>Statistics on this NetPeer since it was initialized</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Gets the NetPeerStatus of the NetPeer</td>
</tr>
<tr>
<td><strong>Tag</strong></td>
<td>Gets or sets the application defined object containing data about the peer</td>
</tr>
<tr>
<td><strong>UniqueIdentifier</strong></td>
<td>Gets a unique identifier for this NetPeer based on Mac address and ip/port. Note! Not available until Start() has been called!</td>
</tr>
<tr>
<td><strong>UPnP</strong></td>
<td>Returns an UPnP object if enabled in the NetPeerConfiguration</td>
</tr>
</tbody>
</table>
See Also

NetPeer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer.Configuration Property

**NetPeer Class** [See Also](#) [Send Feedback](#)

Gets the configuration used to instantiate this NetPeer

**Namespace:** [Lidgren.Network](#)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetPeerConfiguration Configuration { get; }

Visual Basic

Public ReadOnly Property Configuration As NetPeerConfiguration
Get

Visual C++

public:
    property NetPeerConfiguration& Configuration { NetPeerConfiguration& get ();
}


See Also

NetPeer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer..::.Connections Property

NetPeer Class  See Also  Send Feedback

Gets a copy of the list of connections

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public List<NetConnection> Connections { get; }
```

### Visual Basic

```vbnet
Public ReadOnly Property Connections As List(Of NetConnection)
Get
```

### Visual C++

```cpp
public:
property List<NetConnection>^ Connections { 
    List<NetConnection>^ get();
}
See Also

NetPeer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer...::ConnectionsCount Property

NetPeer Class See Also Send Feedback

 Gets the number of active connections

Namespace: Lidgren.Network
Syntax

C#

public int ConnectionsCount { get; }

Visual Basic

Public ReadOnly Property ConnectionsCount As Integer
    Get

Visual C++

public:
property int ConnectionsCount {
    int get ();
}
See Also

NetPeer Class
Lidgren.Network Namespace
Signalling event which can be waited on to determine when a message is queued for reading. Note that there is no guarantee that after the event is signaled the blocked thread will find the message in the queue. Other user created threads could be preempted and dequeue the message before the waiting thread wakes up.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public AutoResetEvent MessageReceivedEvent { get; }

Visual Basic

Public ReadOnly Property MessageReceivedEvent As AutoResetEvent
Get

Visual C++

public:
property AutoResetEvent^ MessageReceivedEvent {
    AutoResetEvent^ get ();
}
See Also

NetPeer Class
Lidgren.Network Namespace
Gets the port number this NetPeer is listening and sending on, if Start() has been called

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int Port { get; }

Visual Basic

Public ReadOnly Property Port As Integer
    Get

Visual C++

public:
    property int Port {
        int get ();
    }

See Also

NetPeer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeer.....Socket Property

NetPeer Class See Also Send Feedback

Gets the socket, if Start() has been called

Namespace: Lidgren.Network
Syntax

C#

public Socket Socket { get; }

Visual Basic

Public Readonly Property Socket As Socket
Get

Visual C++

public:
property Socket^ Socket {  
Socket^ get ();  
}
See Also

NetPeer Class
Lidgren.Network Namespace
Statistics on this NetPeer since it was initialized

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public NetPeerStatistics Statistics { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property Statistics As NetPeerStatistics
Get
```

**Visual C++**

```cpp
public:
property NetPeerStatistics^ Statistics {
NetPeerStatistics^ get ();
}
```
See Also

NetPeer Class
Lidgren.Network Namespace
NetPeer.Status Property

Gets the NetPeerStatus of the NetPeer

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public NetPeerStatus Status { get; }
```

**Visual Basic**

```vbnet
Public Readonly Property Status As NetPeerStatus
Get
```

**Visual C++**

```cpp
public:
    property NetPeerStatus Status {
    NetPeerStatus get();
    }
```
See Also

NetPeer Class
Lidgren.Network Namespace
Gets or sets the application defined object containing data about the peer

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public Object Tag { get; set; }

Visual Basic

Public Property Tag As Object
    Get
    Set

Visual C++

public:
    property Object^ Tag {
        Object^ get ();
        void set (Object^ value);
    }
See Also

NetPeer Class
Lidgren.Network Namespace
Gets a unique identifier for this NetPeer based on Mac address and ip/port. Note! Not available until Start() has been called!

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public long UniqueIdentifier { get; }

Visual Basic

Public ReadOnly Property UniqueIdentifier As Long
Get

Visual C++

public:
property long long UniqueIdentifier {
    long long get ();
}
See Also

NetPeer Class
Lidgren.Network Namespace
Returns an UPnP object if enabled in the NetPeerConfiguration

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetUPnP UPnP { get; }

Visual Basic

Public ReadOnly Property UPnP As NetUPnP
        Get

Visual C++

public:
    property NetUPnP^ UPnP {
        NetUPnP^ get ();
    }
See Also

NetPeer Class
Lidgren.Network Namespace
Partly immutable after NetPeer has been initialized

**Namespace:** [Lidgren.Network](http://Lidgren.Network)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public sealed class NetPeerConfiguration

Visual Basic

Public NotInheritable Class NetPeerConfiguration

Visual C++

public ref class NetPeerConfiguration sealed
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetPeerConfiguration
See Also

NetPeerConfiguration Members
Lidgren.Network Namespace
The **NetPeerConfiguration** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetPeerConfiguration</td>
<td>NetPeerConfiguration constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clone</td>
<td>Creates a memberwise shallow clone of this configuration</td>
</tr>
<tr>
<td>DisableMessageType</td>
<td>Disables receiving of the specified type of message</td>
</tr>
<tr>
<td>EnableMessageType</td>
<td>Enables receiving of the specified type of message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <code>Object</code> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <code>Type</code> of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>IsMessageTypeEnabled</td>
<td>Gets if receiving of the specified type of message is enabled</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>SetMessageTypeEnabled</td>
<td>Enables or disables receiving of the specified type of message</td>
</tr>
<tr>
<td></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>ToString</td>
<td></td>
</tr>
</tbody>
</table>

(Object)
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcceptIncomingConnections</td>
<td>Gets or sets if the NetPeer should accept incoming connections. This is automatically set to true in NetServer and false in NetClient.</td>
</tr>
<tr>
<td>AppIdentifier</td>
<td>Gets the identifier of this application; the library can only connect to matching app identifier peers.</td>
</tr>
<tr>
<td>AutoExpandMTU</td>
<td>Gets or sets if the NetPeer should send large messages to try to expand the maximum transmission unit size.</td>
</tr>
<tr>
<td>AutoFlushSendQueue</td>
<td>Enables or disables automatic flushing of the send queue. If disabled, you must manually call NetPeer.FlushSendQueue() to flush sent messages to network.</td>
</tr>
<tr>
<td>ConnectionTimeout</td>
<td>Gets or sets the number of seconds timeout will be postponed on a successful ping/pong.</td>
</tr>
<tr>
<td>DefaultOutgoingMessageCapacity</td>
<td>Gets or sets the default capacity in bytes when NetPeer.CreateMessage() is called without argument.</td>
</tr>
<tr>
<td>EnableUPnP</td>
<td>Enables UPnP support; enabling port forwarding and getting external ip.</td>
</tr>
<tr>
<td>ExpandMTUFailAttempts</td>
<td>Gets or sets the number of failed expand mtu attempts to perform before setting final MTU.</td>
</tr>
<tr>
<td>ExpandMTUFrequency</td>
<td>Gets or sets how often to send large messages to expand MTU if AutoExpandMTU is enabled.</td>
</tr>
<tr>
<td><strong>LocalAddress</strong></td>
<td>Gets or sets the local ip address to bind to. Defaults to IPAddress.Any. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>MaximumConnections</strong></td>
<td>Gets or sets the maximum amount of connections this peer can hold. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>MaximumHandshakeAttempts</strong></td>
<td>Gets or sets the maximum number of handshake attempts before failing to connect</td>
</tr>
<tr>
<td><strong>MaximumTransmissionUnit</strong></td>
<td>Gets or sets the maximum amount of bytes to send in a single packet, excluding ip, udp and lidgren headers. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>NetworkThreadName</strong></td>
<td>Gets or sets the name of the library network thread. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>PingInterval</strong></td>
<td>Gets or sets the time between latency calculating pings</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Gets or sets the local port to bind to. Defaults to 0. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>ReceiveBufferSize</strong></td>
<td>Gets or sets the size in bytes of the receiving buffer. Defaults to 131071 bytes. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>ResendHandshakeInterval</strong></td>
<td>Gets or sets the number of seconds between handshake attempts</td>
</tr>
<tr>
<td><strong>SendBufferSize</strong></td>
<td>Gets or sets the size in bytes of the sending buffer. Defaults to 131071 bytes. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>SimulatedAverageLatency</strong></td>
<td>Gets the average simulated one way latency in seconds</td>
</tr>
<tr>
<td><strong>SendBufferSize</strong></td>
<td>Gets or sets the simulated amount of</td>
</tr>
<tr>
<td><strong>SimulatedDuplicatesChance</strong></td>
<td>duplicated packets from 0.0f to 1.0f</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td><strong>SimulatedLoss</strong></td>
<td>Gets or sets the simulated amount of sent packets lost from 0.0f to 1.0f</td>
</tr>
<tr>
<td><strong>SimulatedMinimumLatency</strong></td>
<td>Gets or sets the minimum simulated amount of one way latency for sent packets in seconds</td>
</tr>
<tr>
<td><strong>SimulatedRandomLatency</strong></td>
<td>Gets or sets the simulated added random amount of one way latency for sent packets in seconds</td>
</tr>
<tr>
<td><strong>UseMessageRecycling</strong></td>
<td>Gets or sets if the library should recycling messages to avoid excessive garbage collection. Cannot be changed once NetPeer is initialized.</td>
</tr>
</tbody>
</table>
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
NetPeerConfiguration class

Namespace: Lidgren.Network
Syntax

C#

public NetPeerConfiguration(
    string appIdentifier
)

Visual Basic

Public Sub New (_
    appIdentifier As String _
)

Visual C++

public:
NetPeerConfiguration(
    String^ appIdentifier
)

Parameters

appIdentifier
Type: System::String

See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
The `NetPeerConfiguration` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clone</strong></td>
<td>Creates a memberwise shallow clone of this configuration</td>
</tr>
<tr>
<td><strong>DisableMessageType</strong></td>
<td>Disables receiving of the specified type of message</td>
</tr>
<tr>
<td><strong>EnableMessageType</strong></td>
<td>Enables receiving of the specified type of message</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is equal to the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://docs.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>IsMessageTypeEnabled</strong></td>
<td>Gets if receiving of the specified type of message is enabled</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>SetMessageTypeEnabled</strong></td>
<td>Enables or disables receiving of the specified type of message</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="https://docs.microsoft.com/en-us/dotnet/api/system.string">String</a> that represents the current <a href="https://docs.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
NetPeerConfiguration Clone Method

Creates a memberwise shallow clone of this configuration

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetPeerConfiguration Clone()

Visual Basic

Public Function Clone As NetPeerConfiguration

Visual C++

public:
NetPeerConfiguration^ Clone()

Return Value

See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Disables receiving of the specified type of message

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void DisableMessageType(
    NetIncomingMessageType type
)
```

Visual Basic

```
Public Sub DisableMessageType (_
    type As NetIncomingMessageType _
)
```

Visual C++

```cpp
public:
void DisableMessageType(
    NetIncomingMessageType type
)
```

Parameters

type


[Missing <param name="type"/> documentation for
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
NetPeerConfiguration..::..EnableMessageType Method

NetPeerConfiguration Class See Also Send Feedback

Enables receiving of the specified type of message

Namespace: Lidgren.Network
Syntax

C#

public void EnableMessageType(
    NetIncomingMessageType type
)

Visual Basic

Public Sub EnableMessageType (_
    type As NetIncomingMessageType _
)

Visual C++

public:
void EnableMessageType(
    NetIncomingMessageType type
)

Parameters

type

See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
NetPeerConfiguration..::..IsMessageTypeEnabled Method

Gets if receiving of the specified type of message is enabled

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public bool IsMessageTypeEnabled(
    NetIncomingMessageType type
)
```

**Visual Basic**

```vbnet
Public Function IsMessageTypeEnabled (_
    type As NetIncomingMessageType _
) As Boolean
```

**Visual C++**

```cpp
public:
bool IsMessageTypeEnabled(
    NetIncomingMessageType type
)
```

**Parameters**

type


**Return Value**


See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerConfiguration..::..SetMessageTypeEnabled Method

NetPeerConfiguration Class  See Also  Send Feedback

Enables or disables receiving of the specified type of message

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void SetMessageTypeEnabled(
    NetIncomingMessageType type,
    bool enabled
)

Visual Basic

Public Sub SetMessageTypeEnabled (_
    type As NetIncomingMessageType, _
    enabled As Boolean _
)

Visual C++

public:
void SetMessageTypeEnabled(
    NetIncomingMessageType type,
    bool enabled
)

Parameters

type
Type: Lidgren.Network.NetIncomingMessageType


enabled
Type: System.Boolean

See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
The `NetPeerConfiguration` type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AcceptIncomingConnections</strong></td>
<td>Gets or sets if the NetPeer should accept incoming connections. This is automatically set to true in NetServer and false in NetClient.</td>
</tr>
<tr>
<td><strong>AppIdentifier</strong></td>
<td>Gets the identifier of this application; the library can only connect to matching app identifier peers.</td>
</tr>
<tr>
<td><strong>AutoExpandMTU</strong></td>
<td>Gets or sets if the NetPeer should send large messages to try to expand the maximum transmission unit size</td>
</tr>
<tr>
<td><strong>AutoFlushSendQueue</strong></td>
<td>Enables or disables automatic flushing of the send queue. If disabled, you must manually call NetPeer.FlushSendQueue() to flush sent messages to network.</td>
</tr>
<tr>
<td><strong>ConnectionTimeout</strong></td>
<td>Gets or sets the number of seconds timeout will be postponed on a successful ping/pong</td>
</tr>
<tr>
<td><strong>DefaultOutgoingMessageCapacity</strong></td>
<td>Gets or sets the default capacity in bytes when NetPeer.CreateMessage() is called without argument</td>
</tr>
<tr>
<td><strong>EnableUPnP</strong></td>
<td>Enables UPnP support; enabling port forwarding and getting external ip</td>
</tr>
<tr>
<td><strong>ExpandMTUFailAttempts</strong></td>
<td>Gets or sets the number of failed expand mtu attempts to perform before setting final MTU</td>
</tr>
<tr>
<td><strong>ExpandMTUFrequency</strong></td>
<td>Gets or sets how often to send large messages to expand MTU if AutoExpandMTU is enabled</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>LocalAddress</strong></td>
<td>Gets or sets the local ip address to bind to. Defaults to IPAddress.Any. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>MaximumConnections</strong></td>
<td>Gets or sets the maximum amount of connections this peer can hold. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>MaximumHandshakeAttempts</strong></td>
<td>Gets or sets the maximum number of handshake attempts before failing to connect</td>
</tr>
<tr>
<td><strong>MaximumTransmissionUnit</strong></td>
<td>Gets or sets the maximum amount of bytes to send in a single packet, excluding ip, udp and lidgren headers. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>NetworkThreadName</strong></td>
<td>Gets or sets the name of the library network thread. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>PingInterval</strong></td>
<td>Gets or sets the time between latency calculating pings</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Gets or sets the local port to bind to. Defaults to 0. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>ReceiveBufferSize</strong></td>
<td>Gets or sets the size in bytes of the receiving buffer. Defaults to 131071 bytes. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>ResendHandshakeInterval</strong></td>
<td>Gets or sets the number of seconds between handshake attempts</td>
</tr>
<tr>
<td><strong>SendBufferSize</strong></td>
<td>Gets or sets the size in bytes of the sending buffer. Defaults to 131071 bytes. Cannot be changed once NetPeer is initialized.</td>
</tr>
<tr>
<td><strong>SimulatedAverageLatency</strong></td>
<td>Gets the average simulated one way latency in seconds</td>
</tr>
<tr>
<td></td>
<td>Gets or sets the simulated amount of</td>
</tr>
</tbody>
</table>
### SimulatedDuplicatesChance
- Gets or sets the simulated amount of duplicated packets from 0.0f to 1.0f.

### SimulatedLoss
- Gets or sets the simulated amount of sent packets lost from 0.0f to 1.0f.

### SimulatedMinimumLatency
- Gets or sets the minimum simulated amount of one way latency for sent packets in seconds.

### SimulatedRandomLatency
- Gets or sets the simulated added random amount of one way latency for sent packets in seconds.

### UseMessageRecycling
- Gets or sets if the library should recycling messages to avoid excessive garbage collection. Cannot be changed once NetPeer is initialized.
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerConfiguration::AcceptIncomingConnections Property

NetPeerConfiguration Class See Also Send Feedback

Gets or sets if the NetPeer should accept incoming connections. This is automatically set to true in NetServer and false in NetClient.

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public bool AcceptIncomingConnections { get; set; }
```

**Visual Basic**

```vbnet
Public Property AcceptIncomingConnections As Boolean
    Get
    Set
```

**Visual C++**

```c++
public:
    property bool AcceptIncomingConnections {
        bool get ();
        void set (bool value);
    }
```
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Gets the identifier of this application; the library can only connect to matching app identifier peers

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public string AppIdentifier { get; }

Visual Basic

Public ReadOnly Property AppIdentifier As String
Get

Visual C++

public:
property String^ AppIdentifier {
    String^ get ();
}

}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Gets or sets if the NetPeer should send large messages to try to expand the maximum transmission unit size.

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool AutoExpandMTU { get; set; }

Visual Basic

Public Property AutoExpandMTU As Boolean
Get
Set

Visual C++

public:
property bool AutoExpandMTU {
    bool get ();
    void set (bool value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Enables or disables automatic flushing of the send queue. If disabled, you must manually call NetPeer.FlushSendQueue() to flush sent messages to network.

Namespace: Lidgren.Network
Syntax

C#

public bool AutoFlushSendQueue { get; set; }

Visual Basic

Public Property AutoFlushSendQueue As Boolean
    Get
    Set

Visual C++

public:
    property bool AutoFlushSendQueue {
        bool get ();
        void set (bool value);
    }
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Gets or sets the number of seconds timeout will be postponed on a successful ping/pong

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public float ConnectionTimeout { get; set; }

Visual Basic

Public Property ConnectionTimeout As Single
  Get
  Set

Visual C++

public:
property float ConnectionTimeout {
  float get ();
  void set (float value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerConfiguration::DefaultOutgoingMessageCapacity Property

Gets or sets the default capacity in bytes when NetPeer.CreateMessage() is called without argument

Namespace: Lidgren.Network
Syntax

C#

public int DefaultOutgoingMessageCapacity { get; set; }

Visual Basic

Public Property DefaultOutgoingMessageCapacity As Integer
    Get
        Set

Visual C++

public:
    property int DefaultOutgoingMessageCapacity { 
        int get ();
        void set (int value);
    }
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Enables UPnP support; enabling port forwarding and getting external ip

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool EnableUPnP { get; set; }

Visual Basic

Public Property EnableUPnP As Boolean
Get
Set

Visual C++

public:
property bool EnableUPnP {
    bool get ();
    void set (bool value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Gets or sets the number of failed expand mtu attempts to perform before setting final MTU

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int ExpandMTUFailAttempts { get; set; }

Visual Basic

Public Property ExpandMTUFailAttempts As Integer
Get
Set

Visual C++

public:
property int ExpandMTUFailAttempts {
    int get ();
    void set (int value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerConfiguration::ExpandMTUFrequency Property

NetPeerConfiguration Class See Also Send Feedback

Gets or sets how often to send large messages to expand MTU if AutoExpandMTU is enabled

Namespace: Lidgren.Network
Syntax

C#

public float ExpandMTUFrequency { get; set; }

Visual Basic

Public Property ExpandMTUFrequency As Single
    Get
    Set

Visual C++

public:
    property float ExpandMTUFrequency {
        float get ();
        void set (float value);
    }
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
C#  Visual Basic  Visual C++

Lidgren Network Library documentation
NetPeerConfiguration..:..LocalAddress Property

**NetPeerConfiguration Class** See Also  Send Feedback

Gets or sets the local ip address to bind to. Defaults to IPAddress.Any. Cannot be changed once NetPeer is initialized.

**Namespace:** [Lidgren.Network](#)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public IPAddress LocalAddress { get; set; }

Visual Basic

Public Property LocalAddress As IPAddress
    Get
    Set

Visual C++

public:

property IPAddress^ LocalAddress {
    IPAddress^ get ();
    void set (IPAddress^ value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerConfiguration::MaximumConnections Property

Gets or sets the maximum amount of connections this peer can hold. Cannot be changed once NetPeer is initialized.

Namespace: Lidgren.Network
Syntax

C#

public int MaximumConnections { get; set; }

Visual Basic

Public Property MaximumConnections As Integer
  Get
  Set

Visual C++

public:
  property int MaximumConnections {
    int get ();
    void set (int value);
  }

See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Gets or sets the maximum number of handshake attempts before failing to connect

**Namespace:** [Lidgren.Network](https://www.lidgren-network.com)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int MaximumHandshakeAttempts { get; set; }

Visual Basic

Public Property MaximumHandshakeAttempts As Integer
    Get
    Set

Visual C++

public:
    property int MaximumHandshakeAttempts {
        int get ();
        void set (int value);
    }
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerConfiguration::MaximumTransmissionUnit Property
NetPeerConfiguration Class See Also Send Feedback

Gets or sets the maximum amount of bytes to send in a single packet, excluding ip, udp and lidgren headers. Cannot be changed once NetPeer is initialized.

Namespace: Lidgren.Network
Syntax

C#

public int MaximumTransmissionUnit { get; set; }

Visual Basic

Public Property MaximumTransmissionUnit As Integer
    Get
        Set

Visual C++

public:
    property int MaximumTransmissionUnit {
        int get();
        void set (int value);
    }
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerConfiguration...:..:NetworkThreadName Property

NetPeerConfiguration Class See Also Send Feedback

Gets or sets the name of the library network thread. Cannot be changed once NetPeer is initialized.

Namespace: Lidgren.Network
Syntax

C#

public string NetworkThreadName { get; set; }

Visual Basic

Public Property NetworkThreadName As String
    Get
    Set

Visual C++

public:
    property String^ NetworkThreadName { 
        String^ get ();
        void set (String^ value);
    }
See Also

- NetPeerConfiguration Class
- Lidgren.Network Namespace
Gets or sets the time between latency calculating pings

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public float PingInterval { get; set; }

Visual Basic

Public Property PingInterval As Single
Get
Set

Visual C++

public:
property float PingInterval {
    float get ();
    void set (float value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
NetPeerConfiguration.Port Property

Gets or sets the local port to bind to. Defaults to 0. Cannot be changed once NetPeer is initialized.

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int Port { get; set; }

Visual Basic

Public Property Port As Integer
Get
Set

Visual C++

public:
property int Port {
    int get ();
    void set (int value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Gets or sets the size in bytes of the receiving buffer. Defaults to 131071 bytes. Cannot be changed once NetPeer is initialized.

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int ReceiveBufferSize { get; set; }

Visual Basic

Public Property ReceiveBufferSize As Integer
    Get
        Set

Visual C++

public:
property int ReceiveBufferSize {  // C++ access keyword
    int get ();
    void set (int value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerConfiguration::ResendHandshakeInterval Property

Gets or sets the number of seconds between handshake attempts

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
(2012.1.7.0)
Syntax

C#

public float ResendHandshakeInterval { get; set; }

Visual Basic

Public Property ResendHandshakeInterval As Single
    Get
    Set

Visual C++

public:
    property float ResendHandshakeInterval {
        float get ();
        void set (float value);
    }
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
NetPeerConfiguration::SendBufferSize Property

Gets or sets the size in bytes of the sending buffer. Defaults to 131071 bytes. Cannot be changed once NetPeer is initialized.

Namespace: Lidgren.Network
Syntax

C#

public int SendBufferSize { get; set; }

Visual Basic

Public Property SendBufferSize As Integer
Get
Set

Visual C++

public:
property int SendBufferSize {
    int get ();
    void set (int value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerConfiguration...:.SimulatedAverageLatency Property
NetPeerConfiguration Class See Also Send Feedback

Gets the average simulated one way latency in seconds

Namespace: Lidgren.Network
Syntax

C#

```csharp
public float SimulatedAverageLatency { get; }
```

Visual Basic

```vbnet
Public ReadOnly Property SimulatedAverageLatency As Single
    Get
```

Visual C++

```cpp
public:
    property float SimulatedAverageLatency {
        float get ();
    }
```
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerConfiguration::SimulatedDuplicatesChance Property

Gets or sets the simulated amount of duplicated packets from 0.0f to 1.0f

Namespace: Lidgren.Network
Syntax

**C#**

public `float` SimulatedDuplicatesChance { get; set; }

**Visual Basic**

Public Property SimulatedDuplicatesChance As _Single_
Get
Set

**Visual C++**

public:
property `float` SimulatedDuplicatesChance {
    `float` get ();
    void set (`float` value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Gets or sets the simulated amount of sent packets lost from 0.0f to 1.0f

Namespace: Lidgren.Network
Syntax

C#

public float SimulatedLoss { get; set; }

Visual Basic

Public Property SimulatedLoss As Single
Get
Set

Visual C++

public:
property float SimulatedLoss {
    float get ()
    void set (float value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerConfiguration...:..SimulatedMinimumLatency Property

NetPeerConfiguration Class See Also Send Feedback

Gets or sets the minimum simulated amount of one way latency for sent packets in seconds

Namespace: Lidgren.Network
Syntax

C#

```csharp
public float SimulatedMinimumLatency { get; set; }
```

Visual Basic

```vbnet
Public Property SimulatedMinimumLatency As Single
    Get
        Set
    End Property
```

Visual C++

```c++
public:
    property float SimulatedMinimumLatency {
        float get ();
        void set (float value);
    }
```
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerConfiguration::SimulatedRandomLatency Property

NetPeerConfiguration Class See Also Send Feedback

Gets or sets the simulated added random amount of one way latency for sent packets in seconds

Namespace: Lidgren.Network
Syntax

C#

public float SimulatedRandomLatency { get; set; }

Visual Basic

Public Property SimulatedRandomLatency As Single
Get
Set

Visual C++

public:
property float SimulatedRandomLatency {
    float get();
    void set (float value);
}
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerConfiguration::UseMessageRecycling Property

NetPeerConfiguration Class See Also Send Feedback

Gets or sets if the library should recycling messages to avoid excessive garbage collection. Cannot be changed once NetPeer is initialized.

Namespace: Lidgren.Network
Syntax

C#

public bool UseMessageRecycling { get; set; }

Visual Basic

Public Property UseMessageRecycling As Boolean
    Get
        Set

Visual C++

public:
    property bool UseMessageRecycling {
        bool get ();
        void set (bool value);
    }
See Also

NetPeerConfiguration Class
Lidgren.Network Namespace
Statistics for a NetPeer instance

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#
public sealed class NetPeerStatistics

Visual Basic
Public NotInheritable Class NetPeerStatistics

Visual C++
public ref class NetPeerStatistics sealed
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetPeerStatistics
See Also

NetPeerStatistics Members
Lidgren.Network Namespace
The `NetPeerStatistics` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a string that represents this object (Overrides <a href="#">Object</a>::ToString().)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BytesInRecyclePool</td>
<td>Gets the number of bytes in the recycled pool</td>
</tr>
<tr>
<td>ReceivedBytes</td>
<td>Gets the number of received bytes since the NetPeer was initialized</td>
</tr>
<tr>
<td>ReceivedMessages</td>
<td>Gets the number of received messages since the NetPeer was initialized</td>
</tr>
<tr>
<td>ReceivedPackets</td>
<td>Gets the number of received packets since the NetPeer was initialized</td>
</tr>
<tr>
<td>SentBytes</td>
<td>Gets the number of sent bytes since the NetPeer was initialized</td>
</tr>
<tr>
<td>SentMessages</td>
<td>Gets the number of sent messages since the NetPeer was initialized</td>
</tr>
<tr>
<td>SentPackets</td>
<td>Gets the number of sent packets since the NetPeer was initialized</td>
</tr>
<tr>
<td>StorageBytesAllocated</td>
<td>Gets the number of bytes allocated (and possibly garbage collected) for message storage</td>
</tr>
</tbody>
</table>
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
The **NetPeerStatistics** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="https://learn.microsoft.com/en-us/dotnet/api/system.object">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="https://learn.microsoft.com/en-us/dotnet/api/system.object">Object</a> is reclaimed by garbage collection. (Inherited from <a href="https://learn.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="https://learn.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="https://learn.microsoft.com/en-us/dotnet/api/system.type">Type</a> of the current instance. (Inherited from <a href="https://learn.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="https://learn.microsoft.com/en-us/dotnet/api/system.object">Object</a>. (Inherited from <a href="https://learn.microsoft.com/en-us/dotnet/api/system.object">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerStatistics...::ToString Method
NetPeerStatistics Class See Also Send Feedback

Returns a string that represents this object

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public override string ToString()
```

**Visual Basic**

```vbnet
Public Overrides Function ToString As String
```

**Visual C++**

```cpp
public:
virtual String^ ToString() override
```

**Return Value**

See Also

NetPeerStatistics Class
Lidgren.Network Namespace
The `NetPeerStatistics` type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BytesInRecyclePool</td>
<td>Gets the number of bytes in the recycled pool</td>
</tr>
<tr>
<td>ReceivedBytes</td>
<td>Gets the number of received bytes since the NetPeer was initialized</td>
</tr>
<tr>
<td>ReceivedMessages</td>
<td>Gets the number of received messages since the NetPeer was initialized</td>
</tr>
<tr>
<td>ReceivedPackets</td>
<td>Gets the number of received packets since the NetPeer was initialized</td>
</tr>
<tr>
<td>SentBytes</td>
<td>Gets the number of sent bytes since the NetPeer was initialized</td>
</tr>
<tr>
<td>SentMessages</td>
<td>Gets the number of sent messages since the NetPeer was initialized</td>
</tr>
<tr>
<td>SentPackets</td>
<td>Gets the number of sent packets since the NetPeer was initialized</td>
</tr>
<tr>
<td>StorageBytesAllocated</td>
<td>Gets the number of bytes allocated (and possibly garbage collected) for message storage</td>
</tr>
</tbody>
</table>
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerStatistics.BytesInRecyclePool Property

Gets the number of bytes in the recycled pool

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int BytesInRecyclePool { get; }

Visual Basic

Public ReadOnly Property BytesInRecyclePool As Integer
Get

Visual C++

public:
property int BytesInRecyclePool {
    int get ();
}
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Gets the number of received bytes since the NetPeer was initialized

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int ReceivedBytes { get; }

Visual Basic

Public ReadOnly Property ReceivedBytes As Integer
    Get

Visual C++

public:
    property int ReceivedBytes {
        int get ();
    }
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerStatistics..:..ReceivedMessages Property

NetPeerStatistics Class See Also Send Feedback

Gets the number of received messages since the NetPeer was initialized

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public int ReceivedMessages { get; }
```

**Visual Basic**

```vbscript
Public ReadOnly Property ReceivedMessages As Integer
Get
```

**Visual C++**

```cpp
public:
property int ReceivedMessages {
    int get ();
}
```
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Gets the number of received packets since the NetPeer was initialized

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public int ReceivedPackets { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property ReceivedPackets As Integer
Get
```

**Visual C++**

```cpp
public:
property int ReceivedPackets {
    int get ();
}
```
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Gets the number of sent bytes since the NetPeer was initialized

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int SentBytes { get; }

Visual Basic

Public ReadOnly Property SentBytes As Integer
Get

Visual C++

public:
property int SentBytes {
    int get ();
}

See Also

[NetPeerStatistics Class](#)
[Lidgren.Network Namespace](#)
gets the number of sent messages since the NetPeer was initialized

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

public int SentMessages { get; }

**Visual Basic**

Public ReadOnly Property SentMessages As Integer
    Get
        SentMessages
    End Get

**Visual C++**

public:
    property int SentMessages {
        int get ();
    }

See Also

**NetPeerStatistics Class**  
**Lidgren.Network Namespace**
Gets the number of sent packets since the NetPeer was initialized

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int SentPackets { get; }

Visual Basic

Public ReadOnly Property SentPackets As Integer
    Get

Visual C++

public:
    property int SentPackets {
        int get();
    }
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetPeerStatistics::StorageBytesAllocated Property

Gets the number of bytes allocated (and possibly garbage collected) for message storage.

Namespace: Lidgren.Network
Syntax

C#

public long StorageBytesAllocated { get; }

Visual Basic

Public ReadOnly Property StorageBytesAllocated As Long
Get

Visual C++

public:
property long long StorageBytesAllocated {
    long long get ();
}
}
See Also

NetPeerStatistics Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetPeerStatus Enumeration

Status for a NetPeer instance

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public enum NetPeerStatus

Visual Basic

Public Enumeration NetPeerStatus

Visual C++

public enum class NetPeerStatus
<table>
<thead>
<tr>
<th><strong>Member name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>NotRunning</td>
<td></td>
</tr>
<tr>
<td>Starting</td>
<td></td>
</tr>
<tr>
<td>Running</td>
<td></td>
</tr>
<tr>
<td>ShutdownRequested</td>
<td></td>
</tr>
</tbody>
</table>
See Also

Lidgren.Network Namespace
NetQueue<Of <('T')>> Class

Thread safe (blocking) expanding queue with TryDequeue() and EnqueueFirst()

Namespace: Lidgren.Network
Syntax

C#

public sealed class NetQueue<T>

Visual Basic

Public NotInheritable Class NetQueue(Of T)

Visual C++

generic<typename T>
public ref class NetQueue sealed
Type Parameters

T

Inheritance Hierarchy

System..:::Object
Lidgren.Network:::NetQueue<(Of <"T">)>
See Also

NetQueue<Of <(T)>) Members
Lidgren.Network Namespace
The `NetQueue<Of <('T')>>` type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetQueue &lt;(Of &lt;T&gt;)&gt;</td>
<td>NetQueue constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clear</strong></td>
<td>Removes all objects from the queue</td>
</tr>
<tr>
<td><strong>Contains</strong></td>
<td>Determines whether an item is in the queue</td>
</tr>
<tr>
<td><strong>Enqueue(IEnumerable&lt;T&gt;)</strong></td>
<td>Adds an item last/tail of the queue</td>
</tr>
<tr>
<td><strong>Enqueue(T)</strong></td>
<td>Adds an item last/tail of the queue</td>
</tr>
<tr>
<td><strong>EnqueueFirst</strong></td>
<td>Places an item first, at the head of the queue</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td><strong>ToArray</strong></td>
<td>Copies the queue items to a new array</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td><strong>TryDequeue</strong></td>
<td>Gets an item from the head of the queue, or returns default(T) if empty</td>
</tr>
<tr>
<td><strong>TryDrain</strong></td>
<td>Gets an item from the head of the queue, or returns default(T) if empty</td>
</tr>
</tbody>
</table>
TryPeek

Returns default(T) if queue is empty
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity</td>
<td>Gets the current capacity for the queue</td>
</tr>
<tr>
<td>Count</td>
<td>Gets the number of items in the queue</td>
</tr>
</tbody>
</table>
See Also

NetQueue<(Of <(‘T‘)>)> Class
Lidgren.Network Namespace
NetQueue<Of <(T)▻▻> Constructor

NetQueue<Of <(T)▻▻> Class See Also Send Feedback

NetQueue constructor

Namespace: Lidgren.Network
Syntax

C#

public NetQueue(
    int initialCapacity
)

Visual Basic

Public Sub New ( _
    initialCapacity As Integer _
)

Visual C++

public:
NetQueue(
    int initialCapacity
)

Parameters

initialCapacity
Type: System::::Int32

See Also

NetQueue<Of <('T')>> Class
Lidgren.Network Namespace
The **NetQueue<Of <(<'T'>)>>** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Clear</strong></td>
<td>Removes all objects from the queue</td>
</tr>
<tr>
<td><strong>Contains</strong></td>
<td>Determines whether an item is in the queue</td>
</tr>
<tr>
<td><strong>Enqueue(IEnumerable&lt;T&gt;)</strong></td>
<td>Adds an item last/tail of the queue</td>
</tr>
<tr>
<td><strong>Enqueue(T)</strong></td>
<td>Adds an item last/tail of the queue</td>
</tr>
<tr>
<td><strong>EnqueueFirst</strong></td>
<td>Places an item first, at the head of the queue</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>ToArray</strong></td>
<td>Copies the queue items to a new array</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><strong>TryDequeue</strong></td>
<td>Gets an item from the head of the queue, or returns default(T) if empty</td>
</tr>
<tr>
<td><strong>TryDrain</strong></td>
<td>Gets an item from the head of the queue, or returns default(T) if empty</td>
</tr>
</tbody>
</table>
TryPeek

Returns default(T) if queue is empty
See Also

NetQueue(Of T) Class
Lidgren.Network Namespace
Removes all objects from the queue

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public void Clear()
```

**Visual Basic**

```vbnet
Public Sub Clear
```

**Visual C++**

```cpp
public:
void Clear()
```
See Also

NetQueue<(Of <('T')>)> Class
Lidgren.Network Namespace
Determines whether an item is in the queue

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool Contains(T item)

Visual Basic

Public Function Contains ( _
    item As T _
) As Boolean

Visual C++

public:
  bool Contains(T item)

Parameters

item
  Type: T

Return Value


See Also

NetQueue<(Of <('T')>)> Class
Lidgren.Network Namespace
NetQueue<(Of (<'T'>))> Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enqueue(IEnumerable&lt;T&gt;)</td>
<td>Adds an item last/tail of the queue</td>
</tr>
<tr>
<td>Enqueue(T)</td>
<td>Adds an item last/tail of the queue</td>
</tr>
</tbody>
</table>
See Also

NetQueue(Of T) Class
NetQueue(Of T) Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetQueue(Of `<T>`)>...Enqueue Method (IEnumerable(Of `<T>`>)

**NetQueue(Of `<T>`>) Class** See Also Send Feedback

Adds an item last/tail of the queue

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void Enqueue(
    IEnumerable<T> items
)

Visual Basic

Public Sub Enqueue ( _
    items As IEnumerable(Of T) _
)

Visual C++

public:
    void Enqueue(
        IEnumerable<T>^ items
    )

Parameters

items
    Type: System.Collections.Generic..::.IEnumerable<(Of (Of T))>

See Also

NetQueue<(Of '<T>')> Class
Enqueue Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetQueue<(Of <(<'T'>)>)>..:.Enqueue Method (T)

NetQueue<(Of <(<'T'>)>)> Class See Also Send Feedback

Adds an item last/tail of the queue

Namespace: Lidgren.Network
Syntax

C#

```csharp
public void Enqueue(
    T item
)
```

Visual Basic

```vbnet
Public Sub Enqueue (_
    item As T _
)
```

Visual C++

```cpp
public:
void Enqueue(
    T item
)
```

Parameters

item

Type: T

See Also

NetQueue<Of (<'T'>)> Class
Enqueue Overload
Lidgren.Network Namespace
NetQueue<(Of (<'T'>))> Class See Also Send Feedback

Places an item first, at the head of the queue

Namespace: Lidgren.Network
Syntax

C#

public void EnqueueFirst(
    T item
)

Visual Basic

Public Sub EnqueueFirst ( _
    item As T _
)

Visual C++

public:
void EnqueueFirst(
    T item
)

Parameters

item
Type: T

[Missing <param name="item"/> documentation for "M:Lidgren.Network.NetQueue`1.EnqueueFirst(0)"]
See Also

NetQueue<(Of <(<'T'>)>)) Class
Lidgren.Network Namespace
Copies the queue items to a new array

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public T[] ToArray()

Visual Basic

Public Function ToArray As T()

Visual C++

public:
array<T>^ ToArray()

Return Value

See Also

NetQueue<,(Of <(<'T'>)>)> Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetQueue<(Of (<'T'>)>)>::TryDequeue Method

NetQueue<(Of (<'T'>)>)> Class See Also Send Feedback

Gets an item from the head of the queue, or returns default(T) if empty

Namespace: Lidgren.Network
Syntax

C#

public bool TryDequeue(
    out T item
)

Visual Basic

Public Function TryDequeue ( _
    <OutAttribute> ByRef item As T _
) As Boolean

Visual C++

public:
    bool TryDequeue(
        [OutAttribute] T% item
    )

Parameters

item
    Type: T%


Return Value

See Also

NetQueue<(Of <(<T>)>)) Class
Lidgren.Network Namespace
Gets an item from the head of the queue, or returns default(T) if empty

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int TryDrain(
    IList<T> addTo
)

Visual Basic

Public Function TryDrain (_
    addTo As IList(Of T) _
) As Integer

Visual C++

public:
    int TryDrain(
    IList<T>^ addTo
    )

Parameters

addTo

Type: System.Collections.Generic.IList<T>

[Missing <param name="addTo"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetQueue<Of (<T>)> Class
Lidgren.Network Namespace
Returns default(T) if queue is empty

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
# Syntax

**C#**

```csharp
public T TryPeek(
    int offset
)
```

**Visual Basic**

```vbnet
Public Function TryPeek ( _
    offset As Integer _
) As T
```

**Visual C++**

```cpp
public:
T TryPeek(
    int offset
)
```

# Parameters

**offset**

Type: `System::Int32`


# Return Value

See Also

NetQueue<(Of <(<'T'>)>)> Class
Lidgren.Network Namespace
The **NetQueue<Of <-'T'>>** type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity</td>
<td>Gets the current capacity for the queue</td>
</tr>
<tr>
<td>Count</td>
<td>Gets the number of items in the queue</td>
</tr>
</tbody>
</table>
See Also

NetQueue<(Of `<T>`)> Class
Lidgren.Network Namespace
Gets the current capacity for the queue

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public int Capacity { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property Capacity As Integer
    Get
```

**Visual C++**

```cpp
public:
    property int Capacity {
        int get ();
    }
```
See Also

NetQueue<(<T>)> Class
Lidgren.Network Namespace
NetQueue<Of <('T')>> Class See Also Send Feedback

Gets the number of items in the queue

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public int Count { get; }
```

**Visual Basic**

```vbnet
Public ReadOnly Property Count As Integer
    Get
    
    End Get
End Property
```

**Visual C++**

```cpp
public:
    property int Count {
        int get ();
    }
```
See Also

NetQueue<(Of <(<'T'>)>>) Class
Lidgren.Network Namespace
A fast random number generator for .NET Colin Green, January 2005

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public class NetRandom

Visual Basic

Public Class NetRandom

Visual C++

public ref class NetRandom
Inheritance Hierarchy

System..::.Object
Lidgren.Network..::.NetRandom
See Also

NetRandom Members
Lidgren.Network Namespace
The **NetRandom** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NetRandom()</code></td>
<td>Initialises a new instance using time dependent seed.</td>
</tr>
<tr>
<td><code>NetRandom(Int32)</code></td>
<td>Initialises a new instance using an int value as seed.</td>
</tr>
<tr>
<td></td>
<td>This constructor signature is provided to maintain compatibility with System.Random</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>. )</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>. )</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>. )</td>
</tr>
<tr>
<td><strong>GetSeed</strong></td>
<td>Create a semi-random seed based on an object</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>. )</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>. )</td>
</tr>
<tr>
<td><strong>Next0000</strong></td>
<td>Generates a random int over the range 0 to int.MaxValue-1. MaxValue is not generated in order to remain functionally equivalent to System.Random.Next(). This does slightly eat into some of the performance gain over System.Random, but not much. For better performance see: Call NextInt() for an int over the range 0 to int.MaxValue. Call NextUInt() and cast the result to an int to generate an int over the full Int32 value range including negative values.</td>
</tr>
<tr>
<td><strong>Next(Int32)</strong></td>
<td>Generates a random int over the range 0 to upperBound-1, and not including upperBound.</td>
</tr>
<tr>
<td><strong>Next(Int32, Int32)</strong></td>
<td>Generates a random int over the range lowerBound to upperBound-1, and not including upperBound. upperBound must be &gt;= lowerBound. lowerBound may be negative. Generates a single random bit. This method's</td>
</tr>
</tbody>
</table>
**NextBool**

performance is improved by generating 32 bits in one operation and storing them ready for future calls.

Fills the provided byte array with random bytes.

**NextBytes**

This method is functionally equivalent to System.Random.NextBytes().

**NextDouble**

Generates a random double. Values returned are from 0.0 up to but not including 1.0.

Generates a random int over the range 0 to int.MaxValue, inclusive. This method differs from Next() only in that the range is 0 to int.MaxValue and not 0 to int.MaxValue-1. The slight difference in range means this method is slightly faster than Next() but is not functionally equivalent to System.Random.Next().

**NextInt**

Generates a random single. Values returned are from 0.0 up to but not including 1.0.

Generates a uint. Values returned are over the full range of a uint, uint.MinValue to uint.MaxValue, inclusive. This is the fastest method for generating a single random number because the underlying random number generator algorithm generates 32 random bits that can be cast directly to a uint.

**NextUInt**

Reinitialises using an int value as a seed.

**Reinitialise**

Returns a **String** that represents the current **Object**. (Inherited from **Object**.)
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>⚙️ Instance</td>
<td>Gets a global NetRandom instance</td>
</tr>
</tbody>
</table>
See Also

**NetRandom Class**
**Lidgren.Network Namespace**
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetRandom Constructor

See Also
Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NetRandom()</code></td>
<td>Initialises a new instance using time dependent seed.</td>
</tr>
<tr>
<td><code>NetRandom(Int32)</code></td>
<td>Initialises a new instance using an int value as seed.</td>
</tr>
<tr>
<td></td>
<td>This constructor signature is provided to maintain compatibility with System.Random</td>
</tr>
</tbody>
</table>
See Also

NetRandom Class
NetRandom Members
Lidgren.Network Namespace
Initialises a new instance using time dependent seed.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public NetRandom()
```

**Visual Basic**

```vbnet
Public Sub New
```

**Visual C++**

```cpp
public:
NetRandom()
```
See Also

NetRandom Class
NetRandom Overload
Lidgren.Network Namespace
Initialises a new instance using an int value as seed. This constructor signature is provided to maintain compatibility with System.Random.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetRandom(
    int seed
)

Visual Basic

Public Sub New (
    seed As Integer
)

Visual C++

public:
NetRandom(
    int seed
)

Parameters

seed
Type: System:::Int32

See Also

NetRandom Class
NetRandom Overload
Lidgren.Network Namespace
The `NetRandom` type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance</td>
<td>Gets a global NetRandom instance</td>
</tr>
</tbody>
</table>
See Also

NetRandom Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetRandom..::.Instance Field

**NetRandom Class** [See Also](#) [Send Feedback](#)

Gets a global NetRandom instance

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll)  
Version: 2012.1.7.0  
(2012.1.7.0)
Syntax

**C#**

```csharp
public static readonly NetRandom Instance
```

**Visual Basic**

```vbnet
Public Shared ReadOnly Instance As NetRandom
```

**Visual C++**

```cpp
public:
static initonly NetRandom^ Instance
```
See Also

NetRandom Class
Lidgren.Network Namespace
The `NetRandom` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Equals</code></td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>Finalize</code></td>
<td>Allows an <code>Object</code> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited</td>
</tr>
<tr>
<td></td>
<td>from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>GetHashCode</code></td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>GetSeed</code></td>
<td>Create a semi-random seed based on an object</td>
</tr>
<tr>
<td><code>GetType</code></td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>MemberwiseClone</code></td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td><code>Next()</code></td>
<td>Generates a random int over the range 0 to int.MaxValue-1. MaxValue is</td>
</tr>
<tr>
<td></td>
<td>not generated in order to remain functionally equivalent to System.Random.</td>
</tr>
<tr>
<td></td>
<td>Next(). This does slightly eat into some of the performance gain over</td>
</tr>
<tr>
<td></td>
<td>System.Random, but not much. For better performance see: Call NextInt() for</td>
</tr>
<tr>
<td></td>
<td>an int over the range 0 to int.MaxValue. Call NextUInt() and cast the</td>
</tr>
<tr>
<td></td>
<td>result to an int to generate an int over the full Int32 value range</td>
</tr>
<tr>
<td></td>
<td>including negative values.</td>
</tr>
<tr>
<td><code>Next(Int32)</code></td>
<td>Generates a random int over the range 0 to upperBound-1, and not including</td>
</tr>
<tr>
<td></td>
<td>upperBound.</td>
</tr>
<tr>
<td><code>Next(Int32, Int32)</code></td>
<td>Generates a random int over the range lowerBound to upperBound-1, and not</td>
</tr>
<tr>
<td></td>
<td>including upperBound. upperBound must be &gt;= lowerBound. lowerBound may be</td>
</tr>
<tr>
<td></td>
<td>negative.</td>
</tr>
<tr>
<td></td>
<td>Generates a single random bit. This method's</td>
</tr>
</tbody>
</table>
NextBool

Performance is improved by generating 32 bits in one operation and storing them ready for future calls.

NextBytes

Fills the provided byte array with random bytes.

This method is functionally equivalent to System.Random.NextBytes().

NextDouble

Generates a random double. Values returned are from 0.0 up to but not including 1.0.

Generates a random int over the range 0 to int.MaxValue, inclusive. This method differs from Next() only in that the range is 0 to int.MaxValue and not 0 to int.MaxValue-1. The slight difference in range means this method is slightly faster than Next() but is not functionally equivalent to System.Random.Next().

NextInt

Generates a random int over the range 0 to int.MaxValue, inclusive. This method differs from Next() only in that the range is 0 to int.MaxValue and not 0 to int.MaxValue-1. The slight difference in range means this method is slightly faster than Next() but is not functionally equivalent to System.Random.Next().

NextSingle

Generates a random single. Values returned are from 0.0 up to but not including 1.0.

Generates a uint. Values returned are over the full range of a uint, uint.MinValue to uint.MaxValue, inclusive. This is the fastest method for generating a single random number because the underlying random number generator algorithm generates 32 random bits that can be cast directly to a uint.

NextUInt

Reinitialise

Reinitialises using an int value as a seed.

ToString

Returns a String that represents the current Object. (Inherited from Object.)
See Also

NetRandom Class
Lidgren.Network Namespace
Create a semi-random seed based on an object

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int GetSeed(
    Object forObject
)

Visual Basic

Public Function GetSeed ( _
    forObject As Object _
) As Integer

Visual C++

public:
    int GetSeed(
        Object^ forObject
    )

Parameters

forObject
    Type: System:::Object

[Missing <param name="forObject"/> documentation for
]

Return Value

See Also

NetRandom Class
Lidgren.Network Namespace
NetRandom Class See Also Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Generates a random int over the range 0 to int.MaxValue-1. MaxValue is not generated in order to remain functionally equivalent to System.Random.Next(). This does slightly eat into some of the performance gain over System.Random, but not much. For better performance see: Call NextInt() for an int over the range 0 to int.MaxValue. Call NextUInt() and cast the result to an int to generate an int over the full Int32 value range including negative values.</td>
</tr>
<tr>
<td>Next()</td>
<td></td>
</tr>
<tr>
<td>Next(Int32)</td>
<td>Generates a random int over the range 0 to upperBound-1, and not including upperBound.</td>
</tr>
<tr>
<td>Next(Int32, Int32)</td>
<td>Generates a random int over the range lowerBound to upperBound-1, and not including upperBound. upperBound must be &gt;= lowerBound. lowerBound may be negative.</td>
</tr>
</tbody>
</table>
See Also

NetRandom Class
NetRandom Members
Lidgren.Network Namespace
Generates a random int over the range 0 to int.MaxValue-1. MaxValue is not generated in order to remain functionally equivalent to System.Random.Next(). This does slightly eat into some of the performance gain over System.Random, but not much. For better performance see: Call NextInt() for an int over the range 0 to int.MaxValue. Call NextUInt() and cast the result to an int to generate an int over the full Int32 value range including negative values.

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int Next()

Visual Basic

Public Function Next As Integer

Visual C++

public:
int Next()

Return Value

See Also

NetRandom Class
Next Overload
Lidgren.Network Namespace
Generates a random int over the range 0 to upperBound-1, and not including upperBound.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
# Syntax

**C#**

```csharp
public int Next(
    int upperBound
)
```

**Visual Basic**

```vbnet
Public Function Next ( _
    upperBound As Integer _
) As Integer
```

**Visual C++**

```cpp
public:
    int Next( 
        int upperBound
    )
```

# Parameters

**upperBound**

Type: `System::::Int32`


# Return Value

See Also

NetRandom Class
Next Overload
Lidgren.Network Namespace
Generates a random int over the range lowerBound to upperBound-1, and not including upperBound. upperBound must be >= lowerBound. lowerBound may be negative.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int Next(
    int lowerBound,
    int upperBound
)

Visual Basic

Public Function Next ( _
    lowerBound As Integer, _
    upperBound As Integer _
) As Integer

Visual C++

public:
    int Next(
        int lowerBound,
        int upperBound
    )

Parameters

lowerBound
    Type: System::::Int32

    [Missing <param name="lowerBound"/> documentation for

upperBound
    Type: System::::Int32

    [Missing <param name="upperBound"/> documentation for

Return Value
[Missing <returns> documentation for
See Also

NetRandom Class
Next Overload
Lidgren.Network Namespace
Generates a single random bit. This method's performance is improved by generating 32 bits in one operation and storing them ready for future calls.

Namespace: Lidgren.Network
Syntax

C#

public bool NextBool()

Visual Basic

Public Function NextBool As Boolean

Visual C++

public:
bool NextBool()

Return Value

See Also

NetRandom Class
Lidgren.Network Namespace
Fills the provided byte array with random bytes. This method is functionally equivalent to System.Random.NextBytes().

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public void NextBytes(
    byte[] buffer
)

Visual Basic

Public Sub NextBytes ( _
    buffer As Byte() _
)

Visual C++

public:
void NextBytes(
    array<unsigned char>^ buffer
)

Parameters

buffer
Type: array<System::::Byte>[]()[[]]

See Also

NetRandom Class
Lidgren.Network Namespace
Generates a random double. Values returned are from 0.0 up to but not including 1.0.

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public double NextDouble()
```

Visual Basic

```vbs
Public Function NextDouble As Double
```

Visual C++

```cpp
public:
double NextDouble()
```

Return Value

See Also

NetRandom Class
Lidgren.Network Namespace
NetRandom..::..NextInt Method

Generates a random int over the range 0 to int.MaxValue, inclusive. This method differs from Next() only in that the range is 0 to int.MaxValue and not 0 to int.MaxValue-1. The slight difference in range means this method is slightly faster than Next() but is not functionally equivalent to System.Random.Next().

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public int NextInt()

Visual Basic

Public Function NextInt As Integer

Visual C++

public:
  int NextInt()

Return Value

See Also

NetRandom Class
Lidgren.Network Namespace
Generates a random single. Values returned are from 0.0 up to but not including 1.0.

**Namespace:** [Lidgren.Network](http://LidgrenNetwork)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0

(2012.1.7.0)
### Syntax

**C#**

```csharp
public float NextSingle()
```

**Visual Basic**

```vbnet
Public Function NextSingle As Single
```

**Visual C++**

```cpp
public:
float NextSingle()
```

### Return Value

See Also

NetRandom Class
Lidgren.Network Namespace
Generates a uint. Values returned are over the full range of a uint, uint.MinValue to uint.MaxValue, inclusive. This is the fastest method for generating a single random number because the underlying random number generator algorithm generates 32 random bits that can be cast directly to a uint.

Namespace: Lidgren.Network
Syntax

C#

public uint NextUInt()

Visual Basic

Public Function NextUInt As UInteger

Visual C++

public:
    unsigned int NextUInt()

Return Value

See Also

NetRandom Class
Lidgren.Network Namespace
Reinitialises using an int value as a seed.

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public void Reinitialise(
    int seed
)
```

Visual Basic

```vbnet
Public Sub Reinitialise (_
    seed As Integer _
)
```

Visual C++

```cpp
public:
void Reinitialise(
    int seed
)
```

Parameters

seed

Type: System::Int32

See Also

NetRandom Class
Lidgren.Network Namespace
RC2 encryption

Namespace: Lidgren.Network
Syntax

C#

public class NetRC2Encryption : INetEncryption

Visual Basic

Public Class NetRC2Encryption
    Implements INetEncryption

Visual C++

public ref class NetRC2Encryption : INetEncryption
Inheritance Hierarchy

System...Object
Lidgren.Network...NetRC2Encryption
See Also

NetRC2Encryption Members
Lidgren.Network Namespace
The NetRC2Encryption type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetRC2Encryption(String)</td>
<td>NetRC2Encryption constructor</td>
</tr>
<tr>
<td>NetRC2Encryption(array&lt;Byte&gt;[], array&lt;Byte&gt;[])</td>
<td>NetRC2Encryption constructor</td>
</tr>
<tr>
<td>NetRC2Encryption(String, Int32)</td>
<td>NetRC2Encryption constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Decrypt</strong></td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td><strong>Encrypt</strong></td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance.</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>.</td>
</tr>
</tbody>
</table>
See Also

NetRC2Encryption Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetRC2Encryption Constructor

NetRC2Encryption Class See Also Send Feedback
# Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetRC2Encryption(String)</td>
<td>NetRC2Encryption constructor</td>
</tr>
<tr>
<td>NetRC2Encryption(array&lt;Byte&gt;[][], array&lt;Byte&gt;[][])</td>
<td>NetRC2Encryption constructor</td>
</tr>
<tr>
<td>NetRC2Encryption(String, Int32)</td>
<td>NetRC2Encryption constructor</td>
</tr>
</tbody>
</table>
See Also

NetRC2Encryption Class
NetRC2Encryption Members
Lidgren.Network Namespace
NetRC2Encryption constructor

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetRC2Encryption(
    string key
)

Visual Basic

Public Sub New ( _
    key As String _
)

Visual C++

public:
NetRC2Encryption(
    String^ key
)

Parameters

key

Type: System.String

[Missing <param name="key"/> documentation for 
See Also

NetRC2Encryption Class
NetRC2Encryption Overload
Lidgren.Network Namespace
NetRC2Encryption Constructor (array<Byte>[][], array<Byte>[][])  

NetRC2Encryption Class See Also Send Feedback

NetRC2Encryption constructor  

Namespace: Lidgren.Network  
**Syntax**

**C#**

```csharp
public NetRC2Encryption(
    byte[] key,
    byte[] iv
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    key As Byte(), _
    iv As Byte() _
)
```

**Visual C++**

```c++
public:
NetRC2Encryption(
    array<\unsigned\_char>^ key,
    array<\unsigned\_char>^ iv
)
```

**Parameters**

**key**

Type: array<`System::::Byte`>[][]([])


**iv**

Type: array<`System::::Byte`>[][]([])

See Also

NetRC2Encryption Class
NetRC2Encryption Overload
Lidgren.Network Namespace
NetRC2Encryption Constructor (String, Int32)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetRC2Encryption(
    string key,
    int bitsize
)

Visual Basic

Public Sub New ( _
    key As String, _
    bitsize As Integer _
)

Visual C++

public:
NetRC2Encryption(
    String^ key,
    int bitsize
)

Parameters

key
    Type: System..::.String

    [Missing <param name="key"/> documentation for

bitsize
    Type: System..::.Int32

    [Missing <param name="bitsize"/> documentation for
See Also

NetRC2Encryption Class
NetRC2Encryption Overload
Lidgren.Network Namespace
The NetRC2Encryption type exposes the following members.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetRC2Encryption Class
Lidgren.Network Namespace
Decrypt incoming message

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool Decrypt(
    NetIncomingMessage msg
)
```

Visual Basic

```vbnet
Public Function Decrypt(_
    msg As NetIncomingMessage _
) As Boolean
```

Visual C++

```cpp
public:
    virtual bool Decrypt(_
        NetIncomingMessage^ msg
    ) sealed
```

Parameters

msg

Type: Lidgren.Network.NetIncomingMessage


Return Value


Implements

INetEncryption.NetRC2Encryption.Decrypt(NetIncomingMessage)
See Also

NetRC2Encryption Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetRC2Encryption..:..Encrypt Method

NetRC2Encryption Class See Also Send Feedback

Encrypt outgoing message

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
### Syntax

**C#**

```csharp
public bool Encrypt(
    NetOutgoingMessage msg
)
```

**Visual Basic**

```vbnet
Public Function Encrypt ( _
    msg As NetOutgoingMessage _
) As Boolean
```

**Visual C++**

```cpp
public:
    virtual bool Encrypt(
        NetOutgoingMessage^ msg
    ) sealed
```

### Parameters

- **msg**
  
  Type: `Lidgren.Network..::.NetOutgoingMessage`

### Return Value

[Missing <returns> documentation for

### Implements

- `INetEncryption..::.Encrypt(NetOutgoingMessage)`
See Also

NetRC2Encryption Class
Lidgren.Network Namespace
Result of a SendMessage call

Namespace: Lidgren.Network
Syntax

C#

public enum NetSendResult

Visual Basic

Public Enumeration NetSendResult

Visual C++

public enum class NetSendResult
# Members

<table>
<thead>
<tr>
<th>Member name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed</td>
<td>NotConnected</td>
</tr>
<tr>
<td>Sent</td>
<td></td>
</tr>
<tr>
<td>Queued</td>
<td></td>
</tr>
<tr>
<td>Dropped</td>
<td></td>
</tr>
</tbody>
</table>
See Also

Lidgren.Network Namespace
Specialized version of NetPeer used for "server" peers

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public class NetServer : NetPeer

Visual Basic

Public Class NetServer
    Inherits NetPeer

Visual C++

public ref class NetServer : public NetPeer
Inheritance Hierarchy

System..::.Object
Lidgren.Network..::.NetPeer
Lidgren.Network..::.NetServer
See Also

NetServer Members
Lidgren.Network Namespace
The **NetServer** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetServer</td>
<td>NetServer constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect(IPEndPoint)</td>
<td>Create a connection to a remote endpoint. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>Connect(IPEndPoint, NetOutgoingMessage)</td>
<td>Create a connection to a remote endpoint. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>Connect(String, Int32)</td>
<td>Create a connection to a remote endpoint. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>Connect(String, Int32, NetOutgoingMessage)</td>
<td>Create a connection to a remote endpoint. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>CreateMessage()</td>
<td>Creates a new message for sending. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>CreateMessage(Int32)</td>
<td>Creates a new message for sending. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>CreateMessage(String)</td>
<td>Creates a new message for sending and writes the provided string to it. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>DiscoverKnownPeer(IPEndPoint)</td>
<td>Emit a discovery signal to a known host. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>DiscoverKnownPeer(String, Int32)</td>
<td>Emit a discovery signal to a known host. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>DiscoverLocalPeers</td>
<td>Emit a discovery signal to all hosts on your subnet. (Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
FlushSendQueue

If NetPeerConfiguration.AutoFlushSendQueue() is false; you need to call this to send all messages queued using SendMessage() (Inherited from NetPeer.)

GetConnection

Get the connection, if any, for a remote endpoint (Inherited from NetPeer.)

GetHashCode

Serves as a hash function for a particular type. (Inherited from Object.)

GetType

Gets the Type of the current instance. (Inherited from Object.)

Introduce

Send NetIntroduction to hostExternal and clientExternal; introducing client to host (Inherited from NetPeer.)

MemberwiseClone

Creates a shallow copy of the current instance. (Inherited from Object.)

RawSend

Send raw bytes; only used for debugging (Inherited from NetPeer.)

ReadMessage

Read a pending message from any connection, if any (Inherited from NetPeer.)

ReadMessages

Read a pending message from any connection, if any (Inherited from NetPeer.)

Recycle(NetIncomingMessage)

Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)

Recycle(IEnumerable<NetIncomingMessage>)

Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)

RegisterReceivedCallback

Call this to register a callback when a new message arrives (Inherited from NetPeer.)

Send a discovery response message
<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SendDiscoveryResponse</strong></td>
<td>Send a message to a specific connection (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)</strong></td>
<td>Send a message to a specific connection (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)</strong></td>
<td>Send a message to a specific connection (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>SendMessage(NetOutgoingMessage, List&lt;Of NetConnection&gt;, NetDeliveryMethod, Int32)</strong></td>
<td>Send a message to a list of connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>SendToAll(NetOutgoingMessage, NetDeliveryMethod)</strong></td>
<td>Send a message to all connections</td>
</tr>
<tr>
<td><strong>SendToAll(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)</strong></td>
<td>Send a message to all connections except one</td>
</tr>
<tr>
<td><strong>SendUnconnectedMessage(NetOutgoingMessage, IList&lt;Of IPEndPoint&gt;)</strong></td>
<td>Send a message to an unconnected host (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)</strong></td>
<td>Send a message to an unconnected host (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>SendUnconnectedMessage(NetOutgoingMessage, String, Int32)</strong></td>
<td>Send a message to an unconnected host (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>SendUnconnectedToSelf</strong></td>
<td>Send a message to this exact netpeer (loopback) (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Shutdown</strong></td>
<td>Disconnects all active connections and closes the socket (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td>Binds to socket and spawns the networking thread (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a string that represents this object (Overrides Object..::.ToString)</td>
</tr>
<tr>
<td><strong>WaitMessage</strong></td>
<td>Reads a pending message from a connection, blocking up to an endpoint needed (Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>Gets the configuration used to instanciate this NetPeer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connections</td>
<td>Gets a copy of the list of connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>ConnectionsCount</td>
<td>Gets the number of active connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>MessageReceivedEvent</td>
<td>Signalling event which can be waited on to determine when a message is queued for reading. Note that there is no guarantee that after the event is signaled the blocked thread will find the message in the queue. Other user created threads could be preempted and dequeue the message before the waiting thread wakes up. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Port</td>
<td>Gets the port number this NetPeer is listening and sending on, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Socket</td>
<td>Gets the socket, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Statistics</td>
<td>Statistics on this NetPeer since it was initialized (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Status</td>
<td>Gets the NetPeerStatus of the NetPeer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Tag</td>
<td>Gets or sets the application defined object containing data about the peer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td></td>
<td>Gets a unique identifier for this NetPeer based on Mac address and ip/port. Note! Not</td>
</tr>
</tbody>
</table>
**UniqueId** available until Start() has been called!
(Inherited from **NetPeer**.)

**UPnP** Returns an UPnP object if enabled in the NetPeerConfiguration
(Inherited from **NetPeer**.)
See Also

NetServer Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetServer Constructor

NetServer constructor

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public NetServer(
    NetPeerConfiguration config
)
```

**Visual Basic**

```vbscript
Public Sub New ( _
    config As NetPeerConfiguration _
)
```

**Visual C++**

```cpp
public:
    NetServer(
        NetPeerConfiguration^ config
    )
```

**Parameters**

`config`

Type: [Lidgren.Network.NetServer](https://example.com)

See Also

NetServer Class
Lidgren.Network Namespace
The NetServer type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect(IPEndPoint)</td>
<td>Create a connection to a remote endpoint. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(IPEndPoint, NetOutgoingMessage)</td>
<td>Create a connection to a remote endpoint. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(String, Int32)</td>
<td>Create a connection to a remote endpoint. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(String, Int32, NetOutgoingMessage)</td>
<td>Create a connection to a remote endpoint. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage()()()()()</td>
<td>Creates a new message for sending. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(Int32)</td>
<td>Creates a new message for sending. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(String)</td>
<td>Creates a new message for sending. The provided string is written to it. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>DiscoverKnownPeer(IPEndPoint)</td>
<td>Emit a discovery signal to a single known host. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>DiscoverKnownPeer(String, Int32)</td>
<td>Emit a discovery signal to a single known host. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>DiscoverLocalPeers</td>
<td>Emit a discovery signal to all hosts on your subnet. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
FlushSendQueue
If NetPeerConfiguration.AutoFlushSendQueue() is false; you need to call this to send all messages queued using SendMessage (Inherited from NetPeer.)

GetConnection
Get the connection, if any, to a remote endpoint (Inherited from NetPeer.)

GetHashCode
Serves as a hash function for a particular type (Inherited from Object.)

GetType
Gets the Type of the current instance (Inherited from Object.)

Introduce
Send NetIntroduction to host External and client External; introducing client to host (Inherited from NetPeer.)

MemberwiseClone
Creates a shallow copy of the current object (Inherited from Object.)

RawSend
Send raw bytes; only used for debugging (Inherited from NetPeer.)

ReadMessage
Read a pending message from any connection, if any (Inherited from NetPeer.)

ReadMessages
Read a pending message from any connection, if any (Inherited from NetPeer.)

Recycle(NetIncomingMessage)
Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)

Recycle(IEnumerable<NetIncomingMessage>?)
Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)

RegisterReceivedCallback
Call this to register a callback for when a new message arrives (Inherited from NetPeer.)

Send a discovery response message
**SendDiscoveryResponse**  
(Inherited from NetPeer.)

**SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)**  
Send a message to a specific connection  
(Inherited from NetPeer.)

**SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)**  
Send a message to a specific connection and with a timeout  
(Inherited from NetPeer.)

**SendMessage(NetOutgoingMessage, List<Of <'(NetConnection>>>, NetDeliveryMethod, Int32)**  
Send a message to a list of connections  
(Inherited from NetPeer.)

**SendToAll(NetOutgoingMessage, NetDeliveryMethod)**  
Send a message to all connections  
(Inherited from NetPeer.)

**SendToAll(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)**  
Send a message to all connections except one  
(Inherited from NetPeer.)

**SendUnconnectedMessage(NetOutgoingMessage, IList<Of <'(IPEndPoint>>))**  
Send a message to an unconnected host  
(Inherited from NetPeer.)

**SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)**  
Send a message to an unconnected host  
(Inherited from NetPeer.)

**SendUnconnectedMessage(NetOutgoingMessage, String, Int32)**  
Send a message to an unconnected host  
(Inherited from NetPeer.)

**SendUnconnectedToSelf**  
Send a message to this exact netpeer (loopback)  
(Inherited from NetPeer.)

**Shutdown**  
Disconnects all active connections and closes the socket  
(Inherited from NetPeer.)

**Start**  
Binds to socket and spawns the networking thread  
(Inherited from NetPeer.)

**ToString**  
Returns a string that represents this object  
(Overrides Object::<..ToString)

**WaitMessage**  
Read a pending message from a connection, blocking up to maxMillis if needed  
(Inherited from NetPeer.)
See Also

NetServer Class
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect(IPEndPoint)</td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td>(Inherited from NetPeer.)</td>
<td></td>
</tr>
<tr>
<td>Connect(IPEndPoint,</td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td>NetOutgoingMessage)</td>
<td>(Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Connect(String, Int32)</td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td>(Inherited from NetPeer.)</td>
<td></td>
</tr>
<tr>
<td>Connect(String, Int32,</td>
<td>Create a connection to a remote endpoint</td>
</tr>
<tr>
<td>NetOutgoingMessage)</td>
<td>(Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
NetServer Members
Lidgren.Network Namespace
Lidgren Network Library documentation
NetServer::CreateMessage Method
NetServer Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateMessage()()()</td>
<td>Creates a new message for sending (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(Int32)</td>
<td>Creates a new message for sending (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>CreateMessage(String)</td>
<td>Creates a new message for sending and writes the provided string to it (Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
NetServer Members
Lidgren.Network Namespace
C#  □ Visual Basic
□ Visual C++
□ Include Protected Members
□ Include Inherited Members
Lidgren Network Library documentation
NetServer...DiscoverKnownPeer Method
NetServer Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DiscoverKnownPeer(IPEndPoint)</td>
<td>Emit a discovery signal to a single known host</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
<tr>
<td>DiscoverKnownPeer(String, Int32)</td>
<td>Emit a discovery signal to a single known host</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">NetPeer</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
NetServer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetServer::Recycle Method

NetServer Class  See Also  Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recycle(NetIncomingMessage)</td>
<td>Recycles a NetIncomingMessage instance for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>Recycle(IEnumerable&lt;NetIncomingMessage&gt;)</td>
<td>Recycles a list of NetIncomingMessage instances for reuse; taking pressure off the garbage collector (Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
NetServer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetServer::SendMessage Method

NetServer Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod)</code></td>
<td>Send a message to a specific connection (Inherited from <code>NetPeer</code>.)</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)</code></td>
<td>Send a message to a specific connection (Inherited from <code>NetPeer</code>.)</td>
</tr>
<tr>
<td><code>SendMessage(NetOutgoingMessage, List&lt;Of '&lt;(NetConnection)'&gt;), NetDeliveryMethod, Int32)</code></td>
<td>Send a message to a list of connections (Inherited from <code>NetPeer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
NetServer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetServer::SendToAll Method

NetServer Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SendToAll(NetOutgoingMessage, NetDeliveryMethod)</code></td>
<td>Send a message to all connections</td>
</tr>
<tr>
<td><code>SendToAll(NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)</code></td>
<td>Send a message to all connections except one</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
NetServer Members
Lidgren.Network Namespace
Send a message to all connections

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public void SendToAll(
    NetOutgoingMessage msg,
    NetDeliveryMethod method
)
```

**Visual Basic**

```vbnet
Public Sub SendToAll ( _
    msg As NetOutgoingMessage, _
    method As NetDeliveryMethod _
)
```

**Visual C++**

```cpp
public:
void SendToAll(
    NetOutgoingMessage^ msg,
    NetDeliveryMethod method
)
```

**Parameters**

- **msg**
  
  Type: Lidgren.NetworkNetMessage
  
  The message to send

- **method**
  
  Type: Lidgren.NetworkNetMessage
  
  How to deliver the message
See Also

NetServer Class
SendToAll Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetServer.SendToAll Method (NetOutgoingMessage, NetConnection, NetDeliveryMethod, Int32)

Send a message to all connections except one

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
### Syntax

**C#**

```csharp
public void SendToAll(
    NetOutgoingMessage msg,
    NetConnection except,
    NetDeliveryMethod method,
    int sequenceChannel
)
```

**Visual Basic**

```vbnet
Public Sub SendToAll ( _
    msg As NetOutgoingMessage, _
    except As NetConnection, _
    method As NetDeliveryMethod, _
    sequenceChannel As Integer _
)
```

**Visual C++**

```cpp
public:
void SendToAll(
    NetOutgoingMessage^ msg,
    NetConnection^ except,
    NetDeliveryMethod method,
    int sequenceChannel
)
```

### Parameters

**msg**

Type: [Lidgren.Network..NetOutgoingMessage](#)

The message to send

**except**

Type: [Lidgren.Network..NetConnection](#)

Don't send to this particular connection
method
  Type: Lidgren.Network...NetDeliveryMethod
  How to deliver the message

sequenceChannel
  Type: System..Int32
  Which sequence channel to use for the message
See Also

NetServer Class
SendToAll Overload
Lidgren.Network Namespace
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetServer...SendUnconnectedMessage Method

NetServer Class See Also Send Feedback
### Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SendUnconnectedMessage(NetOutgoingMessage, IList&lt;Of &lt;&lt;(IPEndPoint)&gt;&gt;)</code></td>
<td>Send a message to an unconnected host (Inherited from <code>NetPeer</code>.)</td>
</tr>
<tr>
<td><code>SendUnconnectedMessage(NetOutgoingMessage, IPEndPoint)</code></td>
<td>Send a message to an unconnected host (Inherited from <code>NetPeer</code>.)</td>
</tr>
<tr>
<td><code>SendUnconnectedMessage(NetOutgoingMessage, String, Int32)</code></td>
<td>Send a message to an unconnected host (Inherited from <code>NetPeer</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
NetServer Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetServer...::ToString Method

**NetServer Class** [See Also](#) [Send Feedback](#)

Returns a string that represents this object

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public override string ToString()

Visual Basic

Public Overrides Function ToString As String

Visual C++

public:
virtual String^ ToString() override

Return Value

See Also

NetServer Class
Lidgren.Network Namespace
The **NetServer** type exposes the following members.
### Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Configuration</strong></td>
<td>Gets the configuration used to instantiate this NetPeer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Connections</strong></td>
<td>Gets a copy of the list of connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>ConnectionsCount</strong></td>
<td>Gets the number of active connections (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>MessageReceivedEvent</strong></td>
<td>Signalling event which can be waited on to determine when a message is queued for reading. Note that there is no guarantee that after the event is signaled the blocked thread will find the message in the queue. Other user created threads could be preempted and dequeue the message before the waiting thread wakes up. (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Port</strong></td>
<td>Gets the port number this NetPeer is listening and sending on, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Socket</strong></td>
<td>Gets the socket, if Start() has been called (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Statistics</strong></td>
<td>Statistics on this NetPeer since it was initialized (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>Gets the NetPeerStatus of the NetPeer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td><strong>Tag</strong></td>
<td>Gets or sets the application defined object containing data about the peer (Inherited from NetPeer.)</td>
</tr>
<tr>
<td></td>
<td>Gets a unique identifier for this NetPeer based on Mac address and ip/port. Note! Not</td>
</tr>
<tr>
<td>UniqueIdentifier</td>
<td>available until Start() has been called! (Inherited from NetPeer.)</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------</td>
</tr>
<tr>
<td>UPnP</td>
<td>Returns an UPnP object if enabled in the NetPeerConfiguration (Inherited from NetPeer.)</td>
</tr>
</tbody>
</table>
See Also

NetServer Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetSRP Class

Members See Also Send Feedback

Helper methods for implementing SRP authentication

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

public static class NetSRP

**Visual Basic**

Public NotInheritable Class NetSRP

**Visual C++**

public ref class NetSRP abstract sealed
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetSRP
See Also

NetSRP Members
Lidgren.Network Namespace
The `NetSRP` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ComputeClientEphemeral</td>
<td>Compute client public ephemeral value (A)</td>
</tr>
<tr>
<td>ComputeClientSessionValue</td>
<td>Computes the client session value</td>
</tr>
<tr>
<td>ComputePrivateKey</td>
<td>Compute private key (x)</td>
</tr>
<tr>
<td>ComputeServerEphemeral</td>
<td>Compute server ephemeral value (B)</td>
</tr>
<tr>
<td>ComputeServerSessionValue</td>
<td>Computes the server session value</td>
</tr>
<tr>
<td>ComputeServerVerifier</td>
<td>Creates a verifier that the server can later use to authenticate users later on (v)</td>
</tr>
<tr>
<td>ComputeU</td>
<td>Compute intermediate value (u)</td>
</tr>
<tr>
<td>CreateEncryption</td>
<td>Create XTEA symmetrical encryption object from sessionValue</td>
</tr>
<tr>
<td>CreateRandomEphemeral</td>
<td>Create 32 bytes of random ephemeral value</td>
</tr>
<tr>
<td>CreateRandomSalt</td>
<td>Create 16 bytes of random salt</td>
</tr>
<tr>
<td>Hash</td>
<td>SHA hash data</td>
</tr>
</tbody>
</table>
See Also

NetSRP Class
Lidgren.Network Namespace
The **NetSRP** type exposes the following members.
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ComputeClientEphemeral</strong></td>
<td>Compute client public ephemeral value (A)</td>
</tr>
<tr>
<td><strong>ComputeClientSessionValue</strong></td>
<td>Computes the client session value</td>
</tr>
<tr>
<td><strong>ComputePrivateKey</strong></td>
<td>Compute private key (x)</td>
</tr>
<tr>
<td><strong>ComputeServerEphemeral</strong></td>
<td>Compute server ephemeral value (B)</td>
</tr>
<tr>
<td><strong>ComputeServerSessionValue</strong></td>
<td>Computes the server session value</td>
</tr>
<tr>
<td><strong>ComputeServerVerifier</strong></td>
<td>Creates a verifier that the server can later use to authenticate users later on (v)</td>
</tr>
<tr>
<td><strong>ComputeU</strong></td>
<td>Compute intermediate value (u)</td>
</tr>
<tr>
<td><strong>CreateEncryption</strong></td>
<td>Create XTEA symmetrical encryption object from sessionValue</td>
</tr>
<tr>
<td><strong>CreateRandomEphemeral</strong></td>
<td>Create 32 bytes of random ephemeral value</td>
</tr>
<tr>
<td><strong>CreateRandomSalt</strong></td>
<td>Create 16 bytes of random salt</td>
</tr>
<tr>
<td><strong>Hash</strong></td>
<td>SHA hash data</td>
</tr>
</tbody>
</table>
See Also

NetSRP Class
Lidgren.Network Namespace
Compute client public ephemeral value (A)

Namespace: Lidgren.Network
# Syntax

**C#**

public static byte[] ComputeClientEphemeral(
    byte[] clientPrivateEphemeral
)

**Visual Basic**

Public Shared Function ComputeClientEphemeral ( _
    clientPrivateEphemeral As Byte() _
) As Byte()

**Visual C++**

public:
    static array<unsigned char>^ ComputeClientEphemeral(
        array<unsigned char>^ clientPrivateEphemeral
    )

## Parameters

clientPrivateEphemeral
Type: array<System..::.Byte>[][]


## Return Value

See Also

NetSRP Class
Lidgren.Network Namespace
Computes the client session value

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static byte[] ComputeClientSessionValue(
    byte[] serverPublicEphemeral,
    byte[] xdata,
    byte[] udata,
    byte[] clientPrivateEphemeral
)

Visual Basic

Public Shared Function ComputeClientSessionValue (_
    serverPublicEphemeral As Byte(), _
    xdata As Byte(), _
    udata As Byte(), _
    clientPrivateEphemeral As Byte() _
) As Byte()

Visual C++

public:
static array<unsigned char>^ ComputeClientSessionValue(
    array<unsigned char>^ serverPublicEphemeral,
    array<unsigned char>^ xdata,
    array<unsigned char>^ udata,
    array<unsigned char>^ clientPrivateEphemeral
)

Parameters

serverPublicEphemeral
  Type: array<System..::.Byte>[]()[[]]

  [Missing <param name="serverPublicEphemeral"/> documentation for
  Byte[][],System.Byte[]][]]]
udata
Type: array<System::::Byte>[][]

clientPrivateEphemeral
Type: array<System::::Byte>[][]

Return Value


See Also

NetSRP Class
Lidgren.Network Namespace
Computer private key (x)

*Namespace*: Lidgren.Network
Syntax

C#

```csharp
public static byte[] ComputePrivateKey(
    string username,
    string password,
    byte[] salt
)
```

Visual Basic

```vbnet
Public Shared Function ComputePrivateKey ( _
    username As String, _
    password As String, _
    salt As Byte() _
) As Byte()
```

Visual C++

```cpp
public:
static array<unsigned char>^ ComputePrivateKey(
    String^ username,
    String^ password,
    array<unsigned char>^ salt
)
```

Parameters

username
  Type: System::String

[Missing <param name="username"/> documentation for

password
  Type: System::String

[Missing <param name="password"/> documentation for
salt

Type: array<System..::.Byte>[][][]


Return Value

See Also

NetSRP Class
Lidgren.Network Namespace
Compute server ephemeral value (B)

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static byte[] ComputeServerEphemeral(
    byte[] serverPrivateEphemeral,
    byte[] verifier
)

Visual Basic

Public Shared Function ComputeServerEphemeral ( _
    serverPrivateEphemeral As Byte(), _
    verifier As Byte() _
) As Byte()

Visual C++

public:
static array<unsigned char>^ ComputeServerEphemeral(
    array<unsigned char>^ serverPrivateEphemeral, array<unsigned char>^ verifier
)

Parameters

serverPrivateEphemeral
Type: array<System..::.Byte>[]([][])

[Missing <param name="serverPrivateEphemeral"/> documentation for
]

verifier
Type: array<System..::.Byte>[]([][])

[Missing <param name="verifier"/> documentation for
]

Return Value
[Missing <returns> documentation for
See Also

NetSRP Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetSRP...:..:ComputeServerSessionValue Method

NetSRP Class See Also Send Feedback

Computes the server session value

Namespace: Lidgren.Network
Syntax

C#

```csharp
public static byte[] ComputeServerSessionValue(
    byte[] clientPublicEphemeral,
    byte[] verifier,
    byte[] udata,
    byte[] serverPrivateEphemeral
)
```

Visual Basic

```vbnet
Public Shared Function ComputeServerSessionValue (_
    clientPublicEphemeral As Byte(), _
    verifier As Byte(), _
    udata As Byte(), _
    serverPrivateEphemeral As Byte() _
) As Byte()
```

Visual C++

```cpp
public:
static array<unsigned char>^ ComputeServerSessionValue(
    array<unsigned char>^ clientPublicEphemeral,
    array<unsigned char>^ verifier,
    array<unsigned char>^ udata,
    array<unsigned char>^ serverPrivateEphemeral
)
```

Parameters

clientPublicEphemeral
Type: array<System..::.Byte>[]()[[]]


verifier
Type: array<System..::.Byte>[]()[[]]
udata
Type: array<

serverPrivateEphemeral
Type: array<

Return Value

[Missing <returns> documentation for
]
See Also

NetSRP Class
Lidgren.Network Namespace
Creates a verifier that the server can later use to authenticate users later on.

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static byte[] ComputeServerVerifier(
  byte[] privateKey
)

Visual Basic

Public Shared Function ComputeServerVerifier ( _
  privateKey As Byte() _
) As Byte()

Visual C++

public:
  static array<unsigned char>^ ComputeServerVerifier(
    array<unsigned char>^ privateKey
  )

Parameters

privateKey
  Type: array<System::::Byte>[][]

[Missing <param name="privateKey"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetSRP Class
Lidgren.Network Namespace
Compute intermediate value (u)

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public static byte[] ComputeU(
    byte[] clientPublicEphemeral, 
    byte[] serverPublicEphemeral
)
```

### Visual Basic

```vbnet
Public Shared Function ComputeU ( _
    clientPublicEphemeral As Byte(), _
    serverPublicEphemeral As Byte() _
) As Byte()
```

### Visual C++

```cpp
public:
static array<unsigned char>^ ComputeU(
    array<unsigned char>^ clientPublicEphemeral, 
    array<unsigned char>^ serverPublicEphemeral
)
```

## Parameters

**clientPublicEphemeral**
*Type: array<System..::.Byte>[][][]*


**serverPublicEphemeral**
*Type: array<System..::.Byte>[][][]*


## Return Value
[Missing <returns> documentation for
See Also

NetSRP Class
Lidgren.Network Namespace
Create XTEA symmetrical encryption object from sessionValue

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static NetXtea CreateEncryption(
    byte[] sessionValue
)

Visual Basic

Public Shared Function CreateEncryption ( _
    sessionValue As Byte() _
) As NetXtea

Visual C++

public:
static NetXtea^ CreateEncryption(
    array<unsigned char>^ sessionValue
)

Parameters

sessionValue
Type: array<System::::Byte>[]()[[]]

[Missing <param name="sessionValue"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetSRP Class
Lidgren.Network Namespace
Create 32 bytes of random ephemeral value

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static byte[] CreateRandomEphemeral()

Visual Basic

Public Shared Function CreateRandomEphemeral As Byte()  

Visual C++

public:
static array<unsigned char>^ CreateRandomEphemeral()

Return Value

See Also

NetSRP Class
Lidgren.Network Namespace
Create 16 bytes of random salt

Namespace: Lidgren.Network
## Syntax

**C#**

```csharp
public static byte[] CreateRandomSalt()
```

**Visual Basic**

```vbnet
Public Shared Function CreateRandomSalt As Byte()
```

**Visual C++**

```cpp
public:
static array<unsigned char>* CreateRandomSalt()
```

### Return Value

See Also

NetSRP Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetSRP..::..Hash Method

NetSRP Class See Also Send Feedback

SHA hash data

Namespace: Lidgren.Network
Syntax

C#

public static byte[] Hash(
    byte[] data
)

Visual Basic

Public Shared Function Hash ( _
    data As Byte() _
) As Byte()

Visual C++

public: 
static array<unsigned char>^ Hash(
    array<unsigned char>^ data
)

Parameters

data
    Type: array<System::::Byte>[]()[[]]

[Missing <param name="data"/> documentation for "M:Lidgren_NETWORK.NetSRP.Hash(System.Byte[])"]

Return Value

[Missing <returns> documentation for "M:Lidgren_NETWORK.NetSRP.Hash(System.Byte[])"]
See Also

NetSRP Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetTime Class

Members  See Also  Send Feedback

Time service

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#
public static class NetTime

Visual Basic
Public NotInheritable Class NetTime

Visual C++
public ref class NetTime abstract sealed
Inheritance Hierarchy

System...Object
Lidgren.Network...NetTime
See Also

NetTime Members
Lidgren.Network Namespace
The NetTime type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ToReadable</code></td>
<td>Given seconds it will output a human friendly readable string (milliseconds if less than 60 seconds)</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Now</strong></td>
<td>Get number of seconds since the application started</td>
</tr>
</tbody>
</table>
See Also

NetTime Class
Lidgren.Network Namespace
The **NetTime** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ToReadable</td>
<td>Given seconds it will output a human friendly readable string (milliseconds if less than 60 seconds)</td>
</tr>
</tbody>
</table>
See Also

NetTime Class
Lidgren.Network Namespace
NetTime...ToReadable Method

Given seconds it will output a human friendly readable string (milliseconds if less than 60 seconds)

Namespace: Lidgren.Network  
Syntax

C#

```csharp
public static string ToReadable(
    double seconds
)
```

Visual Basic

```vbnet
Public Shared Function ToReadable ( _
    seconds As Double _
) As String
```

Visual C++

```cpp
public:
static String^ ToReadable(
    double seconds
)
```

Parameters

seconds

Type: System::..::Double


Return Value

See Also

NetTime Class
Lidgren.Network Namespace
The NetTime type exposes the following members.
Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Now</td>
<td>Get number of seconds since the application started</td>
</tr>
</tbody>
</table>
See Also

NetTime Class
Lidgren.Network Namespace
Get number of seconds since the application started

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static double Now { get; }

Visual Basic

Public Shared ReadOnly Property Now As Double
  Get

Visual C++

public:
  static property double Now {
    double get ();
  }

See Also

NetTime Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetTripleDESEncryption Class

Members See Also Send Feedback

Triple DES encryption

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public class NetTripleDESEncryption : INetEncryption

Visual Basic

Public Class NetTripleDESEncryption
    Implements INetEncryption

Visual C++

public ref class NetTripleDESEncryption : INetEncryption
Inheritance Hierarchy

System...
Lidgren.Network...

Object
NetTripleDESEncryption
See Also

NetTripleDESEncryption Members
Lidgren.Network Namespace
The NetTripleDESEncryption type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetTripleDESEncryption(String)</td>
<td>NetTripleDESEncryption constructor</td>
</tr>
<tr>
<td>NetTripleDESEncryption(array&lt;Byte&gt;[][], array&lt;Byte&gt;[][])</td>
<td>NetTripleDESEncryption constructor</td>
</tr>
<tr>
<td>NetTripleDESEncryption(String, Int32)</td>
<td>NetTripleDESEncryption constructor</td>
</tr>
</tbody>
</table>
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

- NetTripleDESEncryption Class
- Lidgren.Network Namespace
C#  Visual Basic  Visual C++  Include Protected Members  Include Inherited Members  Lidgren Network Library documentation
NetTripleDESCryption Constructor
NetTripleDESCryption Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetTripleDESEncryption(String)</td>
<td>NetTriplSDESEncryption constructor</td>
</tr>
<tr>
<td>NetTripleDESEncryption(array&lt;Byte&gt;[][], array&lt;Byte&gt;[][][], array&lt;Byte&gt;[][][])</td>
<td>NetTriplSDESEncryption constructor</td>
</tr>
<tr>
<td>NetTripleDESEncryption(String, Int32)</td>
<td>NetTriplSDESEncryption constructor</td>
</tr>
</tbody>
</table>
See Also

NetTripleDESEncryption Class
NetTripleDESEncryption Members
Lidgren.Network Namespace
NetTripleDESEncryption Constructor (String)

Namespace: Lidgren.Network
Syntax

C#

public NetTripleDESEncryption(
    string key
)

Visual Basic

Public Sub New ( _
    key As String _
)

Visual C++

public:
NetTripleDESEncryption(
    String^ key
)

Parameters

key

Type: System..::.String

See Also

NetTripleDESEncryption Class
NetTripleDESEncryption Overload
Lidgren.Network Namespace
NetTripleDESEncryption Constructor (array<Byte>[][], array<Byte>[][])
Syntax

C#

```csharp
public NetTripleDESEncryption(
    byte[] key,
    byte[] iv
)
```

Visual Basic

```vbnet
Public Sub New (_
    key As Byte(), _
    iv As Byte() _
)
```

Visual C++

```cpp
public:
NetTripleDESEncryption(
    array<unsigned char>^ key,
    array<unsigned char>^ iv
)
```

Parameters

key

Type: array<System..::.Byte>[]()[[]]


iv

Type: array<System..::.Byte>[]()[[]]

See Also

NetTripleDESEncryption Class
NetTripleDESEncryption Overload
Lidgren.Network Namespace
NetTripleDESEncryption constructor

Namespace: Lidgren.Network
Syntax

C#

```csharp
public NetTripleDESEncryption(
    string key,
    int bitsize
)
```

Visual Basic

```vbnet
Public Sub New (_
    key As String, _
    bitsize As Integer _
)
```

Visual C++

```cpp
public:
NetTripleDESEncryption(
    String^ key,
    int bitsize
)
```

Parameters

key
Type: System::String


bitsize
Type: System::Int32

See Also

NetTripleDESEncryption Class
NetTripleDESEncryption Overload
Lidgren.Network Namespace
The `NetTripleDESEncryption` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

NetTripleDESEncryption Class
Lidgren.Network Namespace
Decrypt incoming message

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool Decrypt(NetIncomingMessage msg)

Visual Basic

Public Function Decrypt(msg As NetIncomingMessage) As Boolean

Visual C++

public:
virtual bool Decrypt(NetIncomingMessage^ msg)
) sealed

Parameters

msg
Type: Lidgren.Network.NetIncomingMessage

[Missing <param name="msg"/> documentation for

Return Value

[Missing <returns> documentation for

Implements

INetEncryption, Decrypt(NetIncomingMessage)
See Also

NetTripleDESEncryption Class
Lidgren.Network Namespace
Encrypt outgoing message

Namespace: Lidgren.Network
Syntax

C#

public bool Encrypt(
    NetOutgoingMessage msg
)

Visual Basic

Public Function Encrypt ( _
    msg As NetOutgoingMessage _
) As Boolean

Visual C++

public:
    virtual bool Encrypt(
        NetOutgoingMessage^ msg
    ) sealed

Parameters

msg
    Type: Lidgren.Network.NetOutgoingMessage

[Missing <param name="msg"/> documentation for

Return Value

[Missing <returns> documentation for

Implements

INetEncryption.NetTripleDESEncryption.Encrypt(NetOutgoingMessage)
See Also

NetTripleDESEncryption Class
Lidgren.Network Namespace
UPnP support class

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll)  
**Version:** 2012.1.7.0 (2012.1.7.0)
Syntax

### C#

public class NetUPnP

### Visual Basic

Public Class NetUPnP

### Visual C++

public ref class NetUPnP
Inheritance Hierarchy

System...Object
Lidgren.Network...NetUPnP
See Also

NetUPnP Members
Lidgren.Network Namespace
The **NetUPnP** type exposes the following members.
### Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetUPnP</td>
<td>NetUPnP constructor</td>
</tr>
</tbody>
</table>
# Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DeleteForwardingRule</strong></td>
<td>Delete a forwarding rule from the router using UPnP</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <strong>Object</strong> is equal to the current <strong>Object</strong>. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup operations before the <strong>Object</strong> is reclaimed by garbage collection. (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ForwardPort</strong></td>
<td>Add a forwarding rule to the router using UPnP</td>
</tr>
<tr>
<td><strong>GetExternalIP</strong></td>
<td>Retrieve the extern ip using UPnP</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.                           (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.                                (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>.                          (Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <strong>String</strong> that represents the current <strong>Object</strong>.               (Inherited from <strong>Object</strong>.)</td>
</tr>
</tbody>
</table>
See Also

NetUPnP Class
Lidgren.Network Namespace
NetUPnP constructor

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

**C#**

```csharp
public NetUPnP(
    NetPeer peer
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    peer As NetPeer _
)
```

**Visual C++**

```cpp
public:
NetUPnP(
    NetPeer^ peer
)
```

## Parameters

peer

Type: Lidgren.Network.NetPeer

See Also

NetUPnP Class
Lidgren.Network Namespace
The NetUPnP type exposes the following members.
### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DeleteForwardingRule</strong></td>
<td>Delete a forwarding rule from the router using UPnP</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified Object is equal to the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection.</td>
</tr>
<tr>
<td><strong>ForwardPort</strong></td>
<td>Add a forwarding rule to the router using UPnP</td>
</tr>
<tr>
<td><strong>GetExternalIP</strong></td>
<td>Retrieve the extern ip using UPnP</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a String that represents the current Object.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

NetUPnP Class
Lidgren.Network Namespace
Delete a forwarding rule from the router using UPnP

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
public bool DeleteForwardingRule(
    int port
)
```

### Visual Basic

```vbnet
Public Function DeleteForwardingRule ( _
    port As Integer _
) As Boolean
```

### Visual C++

```cpp
public:
    bool DeleteForwardingRule(
        int port
    )
```

## Parameters

**port**

Type: `System::Int32`


## Return Value

See Also

NetUPnP Class
Lidgren.Network Namespace
Add a forwarding rule to the router using UPnP

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public bool ForwardPort(
    int port,
    string description
)

Visual Basic

Public Function ForwardPort ( _
    port As Integer, _
    description As String _
) As Boolean

Visual C++

public:
    bool ForwardPort(
        int port,
        String^ description
    )

Parameters

port
    Type: System::Int32


description
    Type: System::String


Return Value
[Missing <returns> documentation for
See Also

NetUPnP Class
Lidgren.Network Namespace
Retrieve the external IP using UPnP

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public IPAddress GetExternalIP()
```

**Visual Basic**

```vbnet
Public Function GetExternalIP As IPAddress
```

**Visual C++**

```cpp
public: IPAddress^ GetExternalIP()
```

**Return Value**

See Also

NetUPnP Class
Lidgren.Network Namespace
Utility methods

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static class NetUtility

Visual Basic

Public NotInheritable Class NetUtility

Visual C++

public ref class NetUtility abstract sealed
Inheritance Hierarchy

System...Object
Lidgren.Network...NetUtility
See Also

NetUtility Members
Lidgren.Network Namespace
The **NetUtility** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BitsToHoldUInt</strong></td>
<td>Returns how many bits are necessary to hold a certain number</td>
</tr>
<tr>
<td><strong>BytesToHoldBits</strong></td>
<td>Returns how many bytes are required to hold a certain number of bits</td>
</tr>
<tr>
<td><strong>GetMacAddress</strong></td>
<td>Returns the physical (MAC) address for the first usable network interface</td>
</tr>
<tr>
<td><strong>GetMyAddress</strong></td>
<td>Gets my local IP address (not necessarily external) and subnet mask</td>
</tr>
<tr>
<td><strong>GetWindowSize</strong></td>
<td>Gets the window size used internally in the library for a certain delivery method</td>
</tr>
<tr>
<td><strong>IsLocal(IPAddress)</strong></td>
<td>Returns true if the IPAddress supplied is on the same subnet as this host</td>
</tr>
<tr>
<td><strong>IsLocal(IPEndPoint)</strong></td>
<td>Returns true if the IPEndPoint supplied is on the same subnet as this host</td>
</tr>
<tr>
<td><strong>Resolve(String)</strong></td>
<td>Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname</td>
</tr>
<tr>
<td><strong>Resolve(String, Int32)</strong></td>
<td>Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number</td>
</tr>
<tr>
<td><strong>ResolveAsync(String, NetUtility.ResolveAddressCallback)</strong></td>
<td>Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname (asynchronous version)</td>
</tr>
</tbody>
</table>
ResolveAsync(String, Int32, NetUtility..;..ResolveEndPointCallback)
Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number (asynchronous version)

ToByteArray
Convert a hexadecimal string to a byte array

ToHexString(array<Byte>[][][])
Create a hex string from an array of bytes

ToHexString(Int64)
Create a hex string from an Int64 value

ToHumanReadable
Converts a number of bytes to a shorter, more readable string representation
See Also

NetUtility Class
Lidgren.Network Namespace
The **NetUtility** type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BitsToHoldUInt</strong></td>
<td>Returns how many bits are necessary to hold a certain number</td>
</tr>
<tr>
<td><strong>BytesToHoldBits</strong></td>
<td>Returns how many bytes are required to hold a certain number of bits</td>
</tr>
<tr>
<td><strong>GetMacAddress</strong></td>
<td>Returns the physical (MAC) address for the first usable network interface</td>
</tr>
<tr>
<td><strong>GetMyAddress</strong></td>
<td>Gets my local IP address (not necessarily external) and subnet mask</td>
</tr>
<tr>
<td><strong>GetWindowSize</strong></td>
<td>Gets the window size used internally in the library for a certain delivery method</td>
</tr>
<tr>
<td><strong>IsLocal(IPAddress)</strong></td>
<td>Returns true if the IPAddress supplied is on the same subnet as this host</td>
</tr>
<tr>
<td><strong>IsLocal(IPEndPoint)</strong></td>
<td>Returns true if the IPEndPoint supplied is on the same subnet as this host</td>
</tr>
<tr>
<td><strong>Resolve(String)</strong></td>
<td>Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname</td>
</tr>
<tr>
<td><strong>Resolve(String, Int32)</strong></td>
<td>Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number</td>
</tr>
<tr>
<td><strong>ResolveAsync(String, NetUtility::::ResolveAddressCallback)</strong></td>
<td>Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname (asynchronous version)</td>
</tr>
<tr>
<td>Method</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>ResolveAsync(String, Int32, NetUtility::&lt;ResolveEndPointCallback&gt;)</code></td>
<td>Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number (asynchronous version)</td>
</tr>
<tr>
<td><code>ToByteArray</code></td>
<td>Convert a hexadecimal string to a byte array</td>
</tr>
<tr>
<td><code>ToHexString(array&lt;Byte&gt;[][])</code></td>
<td>Create a hex string from an array of bytes</td>
</tr>
<tr>
<td><code>ToHexString(Int64)</code></td>
<td>Create a hex string from an Int64 value</td>
</tr>
<tr>
<td><code>ToHumanReadable</code></td>
<td>Converts a number of bytes to a shorter, more readable string representation</td>
</tr>
</tbody>
</table>
See Also

NetUtility Class
Lidgren.Network Namespace
**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)

Returns how many bits are necessary to hold a certain number
Syntax

C#

public static int BitsToHoldUInt(
    uint value
)

Visual Basic

Public Shared Function BitsToHoldUInt ( _
    value As UInteger _
) As Integer

Visual C++

public:
static int BitsToHoldUInt(
    unsigned int value
)

Parameters

value
    Type: System::::UInt32

[Missing <param name="value"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetUtility Class
Lidgren.Network Namespace
Returns how many bytes are required to hold a certain number of bits

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static int BytesToHoldBits(
        int numBits
    )

Visual Basic

Public Shared Function BytesToHoldBits ( _
    numBits As Integer _
) As Integer

Visual C++

public:
static int BytesToHoldBits(
        int numBits
    )

Parameters

numBits
    Type: System::::Int32

[Missing <param name="numBits"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetUtility Class
Lidgren.Network Namespace
Returns the physical (MAC) address for the first usable network interface

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static PhysicalAddress GetMacAddress()

Visual Basic

Public Shared Function GetMacAddress As PhysicalAddress

Visual C++

public:
static PhysicalAddress^ GetMacAddress()

Return Value

See Also

NetUtility Class
Lidgren.Network Namespace
Lidgren Network Library documentation
NetUtility..::..GetMyAddress Method

NetUtility Class See Also Send Feedback

Gets my local IP address (not necessarily external) and subnet mask

Namespace: Lidgren.Network
Syntax

C#

public static IPAddress GetMyAddress(out IPAddress mask)

Visual Basic

Public Shared Function GetMyAddress ( _
    <OutAttribute> ByRef mask As IPAddress _
) As IPAddress

Visual C++

public:
static IPAddress^ GetMyAddress(
    [OutAttribute] IPAddress^% mask
)

Parameters

mask
Type: System.Net...::IPAddress%


Return Value


See Also

NetUtility Class
Lidgren.Network Namespace
Gets the window size used internally in the library for a certain delivery method

Namespace: Lidgren.Network
## Syntax

### C#

```csharp
public static int GetWindowSize(
    NetDeliveryMethod method
)
```

### Visual Basic

```vbnet
Public Shared Function GetWindowSize ( _
    method As NetDeliveryMethod _
) As Integer
```

### Visual C++

```cpp
public:
static int GetWindowSize(
    NetDeliveryMethod method
)
```

## Parameters

- **method**

[Missing `<param name="method"/>` documentation for
]

## Return Value

[Missing `<returns>` documentation for
]
See Also

NetUtility Class
Lidgren.Network Namespace
C# □ Visual Basic
□ Visual C++
□ Include Protected Members
□ Include Inherited Members
Lidgren Network Library documentation
NetUtility...:...IsLocal Method
NetUtility Class See Also Send Feedback
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsLocal(IPAddress)</td>
<td>Returns true if the IPAddress supplied is on the same subnet as this host</td>
</tr>
<tr>
<td>IsLocal(IPEndPoint)</td>
<td>Returns true if the IPEndPoint supplied is on the same subnet as this host</td>
</tr>
</tbody>
</table>
See Also

NetUtility Class
NetUtility Members
Lidgren.Network Namespace
Returns true if the IPAddress supplied is on the same subnet as this host

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public static bool IsLocal(
    IPAddress remote
)
```

**Visual Basic**

```vbnet
Public Shared Function IsLocal ( _
    remote As IPAddress _
) As Boolean
```

**Visual C++**

```cpp
public:
static bool IsLocal(
    IPAddress^ remote
)
```

**Parameters**

remote

Type: `System.Net.IPAddress`

[Missing `<param name="remote"/>` documentation for

**Return Value**

[Missing `<returns>` documentation for
See Also

NetUtility Class
IsLocal Overload
Lidgren.Network Namespace
Returns true if the IPEndPoint supplied is on the same subnet as this host.

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public static bool IsLocal(
    IPEndPoint endpoint
)
```

**Visual Basic**

```vbnet
Public Shared Function IsLocal ( _
    endpoint As IPEndPoint _
) As Boolean
```

**Visual C++**

```cpp
public:
static bool IsLocal(
    IPEndPoint^ endpoint
)
```

**Parameters**

- `endpoint`
  
  Type: `System.Net...IPEndPoint`

  [Missing <param name="endpoint"/> documentation for

**Return Value**

[Missing <returns> documentation for
See Also

NetUtility Class
IsLocal Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetUtility..::.Resolve Method
NetUtility Class See Also Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Resolve(String)</code></td>
<td>Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname</td>
</tr>
<tr>
<td><code>Resolve(String, Int32)</code></td>
<td>Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number</td>
</tr>
</tbody>
</table>
See Also

NetUtility Class
NetUtility Members
Lidgren.Network Namespace
Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```
public static IPAddress Resolve(
    string ipOrHost
)
```

Visual Basic

```
Public Shared Function Resolve ( _
    ipOrHost As String _
) As IPAddress
```

Visual C++

```
public:
static IPAddress^ Resolve(
    String^ ipOrHost
)
```

Parameters

ipOrHost
Type: System.String

[Missing <param name="ipOrHost"/> documentation for
]

Return Value

]
See Also

NetUtility Class
Resolve Overload
Lidgren.Network Namespace
Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number

Namespace: Lidgren.Network
Syntax

C#

```csharp
public static IPEndPoint Resolve(
    string ipOrHost,
    int port
)
```

Visual Basic

```vbnet
Public Shared Function Resolve ( _
    ipOrHost As String, _
    port As Integer _
) As IPEndPoint
```

Visual C++

```cpp
public:
static IPEndPoint^ Resolve(
    String^ ipOrHost,
    int port
)
```

Parameters

**ipOrHost**
Type: **System::String**


**port**
Type: **System::Int32**


Return Value
See Also

NetUtility Class
Resolve Overload
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetUtility..::..ResolveAsync Method
NetUtility Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ResolveAsync(String, NetUtility,.....,ResolveAddressCallback)</code></td>
<td>Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname (asynchronous version)</td>
</tr>
<tr>
<td><code>ResolveAsync(String, Int32, NetUtility,.....,ResolveEndPointCallback)</code></td>
<td>Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number (asynchronous version)</td>
</tr>
</tbody>
</table>
See Also

NetUtility Class
NetUtility Members
Lidgren.Network Namespace
Get IPv4 address from notation (xxx.xxx.xxx.xxx) or hostname (asynchronous version)

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static void ResolveAsync(
    string ipOrHost,
    NetUtility..::..ResolveAddressCallback callback
)

Visual Basic

Public Shared Sub ResolveAsync (_
    ipOrHost As String, _
    callback As NetUtility..::..ResolveAddressCallback _
)

Visual C++

public:
static void ResolveAsync(
    String^ ipOrHost,
    NetUtility..::..ResolveAddressCallback^ callback
)

Parameters

ipOrHost
Type: System::::String


callback
Type: Lidgren.Network..::..NetUtility..::..ResolveAddressCallback

See Also

- NetUtility Class
- ResolveAsync Overload
- Lidgren.Network Namespace
Get IPv4 endpoint from notation (xxx.xxx.xxx.xxx) or hostname and port number (asynchronous version)

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

**C#**

```csharp
public static void ResolveAsync(
    string ipOrHost,
    int port,
    NetUtility...::...ResolveEndPointCallback callback
)
```

**Visual Basic**

```vbnet
Public Shared Sub ResolveAsync (_
    ipOrHost As String, _
    port As Integer, _
    callback As NetUtility...::...ResolveEndPointCallback _
)
```

**Visual C++**

```cpp
public:
static void ResolveAsync(
    String^ ipOrHost,
    int port,
    NetUtility...::...ResolveEndPointCallback^ callback
)
```

Parameters

**ipOrHost**

Type: **System....String**

[Missing <param name="ipOrHost"/> documentation for

**port**

Type: **System....Int32**

[Missing <param name="port"/> documentation for


callback


See Also

NetUtility Class
ResolveAsync Overload
Lidgren.Network Namespace
Convert a hexadecimal string to a byte array

**Namespace:** Lidgren.Network
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public static byte[] ToByteArray(
    string hexString
)

Visual Basic

Public Shared Function ToByteArray ( _
    hexString As String _
) As Byte()

Visual C++

public:
static array<unsigned char>^ ToByteArray(
    String^ hexString
)

Parameters

hexString
Type: System:::String

[Missing <param name="hexString"/> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetUtility Class
Lidgren.Network Namespace
NetUtility Class  

See Also  
Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ToHexString(array&lt;Byte&gt;[][])</code></td>
<td>Create a hex string from an array of bytes</td>
</tr>
<tr>
<td><code>ToHexString(Int64)</code></td>
<td>Create a hex string from an Int64 value</td>
</tr>
</tbody>
</table>
See Also

NetUtility Class
NetUtility Members
Lidgren.Network Namespace
Create a hex string from an array of bytes

**Namespace:** [Lidgren.Network](#)

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public static string ToHexString(
    byte[] data
)
```

Visual Basic

```vbnet
Public Shared Function ToHexString ( _
    data As Byte() _
) As String
```

Visual C++

```cpp
public:
static String^ ToHexString(
    array<unsigned char>^ data
)
```

Parameters

data

Type: array<System..::.Byte>[][][]

[Missing <param name="data" /> documentation for
]

Return Value

[Missing <returns> documentation for
]
See Also

NetUtility Class
ToHexString Overload
Lidgren.Network Namespace
Create a hex string from an Int64 value

Namespace: Lidgren.Network
Syntax

C#

```csharp
public static string ToHexString(
    long data
)
```

Visual Basic

```vbnet
Public Shared Function ToHexString ( _
    data As Long _
) As String
```

Visual C++

```cpp
public:
static String^ ToHexString( 
    long long data 
)
```

Parameters

data
Type: `System::::Int64`


Return Value

See Also

NetUtility Class
ToHexString Overload
Lidgren.Network Namespace
Lidgren Network Library documentation
NetUtility..::..ToHumanReadable Method

NetUtility Class See Also Send Feedback

Converts a number of bytes to a shorter, more readable string representation

Namespace: Lidgren.Network
Syntax

C#

```csharp
public static string ToHumanReadable(
    long bytes
)
```

Visual Basic

```vbnet
Public Shared Function ToHumanReadable ( _
    bytes As Long _
) As String
```

Visual C++

```cpp
public:
static String^ ToHumanReadable(
    long long bytes
)
```

Parameters

`bytes`
Type: `System::::Int64`

[Missing `<param name="bytes"/>` documentation for 
]

Return Value

[Missing `<returns>` documentation for
]
See Also

NetUtility Class
Lidgren.Network Namespace
Lidgren Network Library documentation

NetUtility..:.:.ResolveAddressCallback Delegate

NetUtility Class See Also Send Feedback


Namespace: Lidgren.Network
Syntax

C#

public delegate void ResolveAddressCallback(
    IPAddress adr
)

Visual Basic

Public Delegate Sub ResolveAddressCallback ( _
   adr As IPAddress _
)

Visual C++

public delegate void ResolveAddressCallback(
    IPAddress^ adr
)

Parameters

adr

Type: System.Net...IPAddress
See Also

Lidgren.Network Namespace

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public delegate void ResolveEndPointCallback(IPEndPoint endpoint)

Visual Basic

Public Delegate Sub ResolveEndPointCallback ( _
endpoint As IPEndPoint _
)

Visual C++

public delegate void ResolveEndPointCallback(IPEndPoint* endpoint)

Parameters

endpoint
  Type: System.Net.IPEndPoint
See Also

Lidgren.Network Namespace
Example class; not very good encryption

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public class NetXorEncryption : INetEncryption

Visual Basic

Public Class NetXorEncryption
    Implements INetEncryption

Visual C++

public ref class NetXorEncryption : INetEncryption
Inheritance Hierarchy

System....Object
Lidgren.Network....NetXorEncryption
See Also

NetXorEncryption Members
Lidgren.Network Namespace
The NetXorEncryption type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NetXorEncryption(array&lt;Byte&gt;[][])</code></td>
<td>NetXorEncryption constructor</td>
</tr>
<tr>
<td><code>NetXorEncryption(String)</code></td>
<td>NetXorEncryption constructor</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt an incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt an outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform other cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <a href="#">Type</a> of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
See Also

NetXorEncryption Class
Lidgren.Network Namespace
C#  Visual Basic
Visual C++
Include Protected Members
Include Inherited Members
Lidgren Network Library documentation
NetXorEncryption Constructor
NetXorEncryption Class  See Also  Send Feedback
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetXorEncryption(array[Byte][][[]])</td>
<td>NetXorEncryption constructor</td>
</tr>
<tr>
<td>NetXorEncryption(String)</td>
<td>NetXorEncryption constructor</td>
</tr>
</tbody>
</table>
See Also

NetXorEncryption Class
NetXorEncryption Members
Lidgren.Network Namespace
NetXorEncryption Constructor (array<Byte>[][][])

NetXorEncryption Class See Also Send Feedback

NetXorEncryption constructor

Namespace: Lidgren.Network
Syntax

C#

public NetXorEncryption(  
    byte[] key
)

Visual Basic

Public Sub New (  
    key As Byte() _
)

Visual C++

public:
NetXorEncryption(  
    array<unsigned char>^ key
)

Parameters

key
  Type: array<System..::.Byte>[][]

See Also

NetXorEncryption Class
NetXorEncryption Overload
Lidgren.Network Namespace
C#  Visual Basic  Visual C++
Lidgren Network Library documentation
NetXorEncryption Constructor (String)

NetXorEncryption Class  See Also  Send Feedback

NetXorEncryption constructor

Namespace: Lidgren.Network
Syntax

C#

public NetXorEncryption(
    string key
)

Visual Basic

Public Sub New (_
    key As String _
)

Visual C++

public:
NetXorEncryption(
    String^ key
)

Parameters

key

Type: System.String

See Also

NetXorEncryption Class
NetXorEncryption Overload
Lidgren.Network Namespace
The `NetXorEncryption` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt an incoming message</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypt an outgoing message</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified Object is equal to the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from Object.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the Type of the current instance. (Inherited from Object.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current Object. (Inherited from Object.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a String that represents the current Object. (Inherited from Object.)</td>
</tr>
</tbody>
</table>
See Also

NetXorEncryption Class
Lidgren.Network Namespace
Decrypt an incoming message

Namespace: Lidgren.Network
Syntax

**C#**

```csharp
public bool Decrypt(NetIncomingMessage msg)
```

**Visual Basic**

```vbnet
Public Function Decrypt(_
    msg As NetIncomingMessage _
) As Boolean
```

**Visual C++**

```cpp
public:
    virtual bool Decrypt(NetIncomingMessage^ msg)
) sealed
```

**Parameters**

- **msg**

**Return Value**


**Implements**

See Also

NetXorEncryption Class
Lidgren.Network Namespace
Encrypt an outgoing message

**Namespace:** Lidgren.Network  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

```csharp
public bool Encrypt(
    NetOutgoingMessage msg
)
```

Visual Basic

```vbnet
Public Function Encrypt (_
    msg As NetOutgoingMessage _
) As Boolean
```

Visual C++

```cpp
public:
    virtual bool Encrypt( _
        NetOutgoingMessage^ msg
    ) sealed
```

Parameters

msg

Type: Lidgren.Network::NetOutgoingMessage

[Missing <param name="msg"/> documentation for
]

Return Value

[Missing <returns> documentation for
]

Implements

INetEncryption::Encrypt(NetOutgoingMessage)
See Also

NetXorEncryption Class
Lidgren.Network Namespace
Methods to encrypt and decrypt data using the XTEA algorithm

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public sealed class NetXtea : NetBlockEncryptionBase

Visual Basic

Public NotInheritable Class NetXtea _
    Inherits NetBlockEncryptionBase

Visual C++

public ref class NetXtea sealed : public NetBlockEncryptionBase
Inheritance Hierarchy

System..::..Object
Lidgren.Network..::..NetBlockEncryptionBase
Lidgren.Network..::..NetXtea
See Also

NetXtea Members
Lidgren.Network Namespace
The **NetXtea** type exposes the following members.
## Constructors

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>NetXtea(array&lt;Byte&gt;[][])</code></td>
<td>16 byte key</td>
</tr>
<tr>
<td><code>NetXtea(String)</code></td>
<td>String to hash for key</td>
</tr>
<tr>
<td><code>NetXtea(array&lt;Byte&gt;[][], Int32)</code></td>
<td>16 byte key</td>
</tr>
</tbody>
</table>
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Decrypt</strong></td>
<td>Decrypt an incoming message encrypted with corresponding Encrypt. (Inherited from <a href="#">NetBlockEncryptionBase</a>.)</td>
</tr>
<tr>
<td><strong>DecryptBlock</strong></td>
<td>Decrypts a block of bytes. (Overrides <a href="#">NetBlockEncryptionBase..::..DecryptBlock(array&lt;Byte&gt;[] Int32, array&lt;Byte&gt;[][])</a>.)</td>
</tr>
<tr>
<td><strong>Encrypt</strong></td>
<td>Encrypt an outgoing message with this algorithm; no writing can be done to the message after encryption, or message will be corrupted. (Inherited from <a href="#">NetBlockEncryptionBase</a>.)</td>
</tr>
<tr>
<td><strong>EncryptBlock</strong></td>
<td>Encrypts a block of bytes. (Overrides <a href="#">NetBlockEncryptionBase..::..EncryptBlock(array&lt;Byte&gt;[] Int32, array&lt;Byte&gt;[][])</a>.)</td>
</tr>
<tr>
<td><strong>Equals</strong></td>
<td>Determines whether the specified <a href="#">Object</a> is equal to the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <a href="#">Object</a> to attempt to free resources and perform cleanup operations before the <a href="#">Object</a> is reclaimed by garbage collection. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Serves as a hash function for a particular type. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <a href="#">Type</a> of the current instance. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns a <a href="#">String</a> that represents the current <a href="#">Object</a>. (Inherited from <a href="#">Object</a>.)</td>
</tr>
</tbody>
</table>
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlockSize</td>
<td>Gets the block size for this cipher (Overrides NetBlockEncryptionBase::BlockSize.)</td>
</tr>
</tbody>
</table>
See Also

NetXtea Class
Lidgren.Network Namespace
## Overload List

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetXtea(array&lt;Byte&gt;[][])</td>
<td>16 byte key</td>
</tr>
<tr>
<td>NetXtea(String)</td>
<td>String to hash for key</td>
</tr>
<tr>
<td>NetXtea(array&lt;Byte&gt;[][], Int32)</td>
<td>16 byte key</td>
</tr>
</tbody>
</table>
See Also

NetXtea Class
NetXtea Members
Lidgren.Network Namespace
Lidgren Network Library documentation

NetXtea Constructor (array<Byte>[][]) ([[]])

NetXtea Class See Also Send Feedback

16 byte key

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
**Syntax**

**C#**

```csharp
public NetXtea(
    byte[] key
)
```

**Visual Basic**

```vbnet
Public Sub New (_
    key As Byte() _
)
```

**Visual C++**

```cpp
public:
NetXtea(
    array<unsigned char>^ key
)
```

**Parameters**

**key**

Type: array`<System::...::Byte>[]`[

See Also

NetXtea Class
NetXtea Overload
Lidgren.Network Namespace
Lidgren Network Library documentation

NetXtea Constructor (String)

NetXtea Class See Also Send Feedback

String to hash for key

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#

public NetXtea(
    string key
)

Visual Basic

Public Sub New ( _
    key As String _
)

Visual C++

public:
NetXtea(  
    String^ key
)

Parameters

key

Type: System::String

See Also

NetXtea Class
NetXtea Overload
Lidgren.Network Namespace
C# □ Visual Basic □ Visual C++
Lidgren Network Library documentation
NetXtea Constructor (array<Byte>[][], Int32)
NetXtea Class See Also Send Feedback

16 byte key

Namespace: Lidgren.Network
**Syntax**

**C#**

```csharp
public NetXtea(
    byte[] key,
    int rounds
)
```

**Visual Basic**

```vbnet
Public Sub New ( _
    key As Byte(), _
    rounds As Integer _
)
```

**Visual C++**

```cpp
public:
NetXtea(
    array<unsigned char>^ key,
    int rounds
)
```

**Parameters**

**key**
Type: array<System::::Byte>[][]


**rounds**
Type: System::::Int32

See Also

NetXtea Class
NetXtea Overload
Lidgren.Network Namespace
The NetXtea type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt</td>
<td>Decrypt an incoming message encrypted with corresponding Encrypt. (Inherited from <code>NetBlockEncryptionBase</code>.)</td>
</tr>
<tr>
<td>DecryptBlock</td>
<td>Decrypts a block of bytes (Overrides <code>NetBlockEncryptionBase..::.DecryptBlock(array&lt;Byte&gt;[] Int32, array&lt;Byte&gt;[][])</code>.)</td>
</tr>
<tr>
<td>Encrypt</td>
<td>Encrypts an outgoing message with this algorithm; no writing can be done to the message after encryption, or message will be corrupted. (Inherited from <code>NetBlockEncryptionBase</code>.)</td>
</tr>
<tr>
<td>EncryptBlock</td>
<td>Encrypts a block of bytes (Overrides <code>NetBlockEncryptionBase..::.EncryptBlock(array&lt;Byte&gt;[] Int32, array&lt;Byte&gt;[][])</code>.)</td>
</tr>
<tr>
<td>Equals</td>
<td>Determines whether the specified <code>Object</code> is equal to the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>Finalize</td>
<td>Allows an <code>Object</code> to attempt to free resources and perform cleanup operations before the <code>Object</code> is reclaimed by garbage collection. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Serves as a hash function for a particular type. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>GetType</td>
<td>Gets the <code>Type</code> of the current instance. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>MemberwiseClone</td>
<td>Creates a shallow copy of the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
<tr>
<td>ToString</td>
<td>Returns a <code>String</code> that represents the current <code>Object</code>. (Inherited from <code>Object</code>.)</td>
</tr>
</tbody>
</table>
See Also

NetXtea Class
Lidgren.Network Namespace
Decrypts a block of bytes

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
## Syntax

### C#

```csharp
protected override void DecryptBlock(
    byte[] source,
    int sourceOffset,
    byte[] destination
)
```

### Visual Basic

```vbnet
Protected Overrides Sub DecryptBlock (_
    source As Byte(), _
    sourceOffset As Integer, _
    destination As Byte() _
)
```

### Visual C++

```cpp
protected:
virtual void DecryptBlock(
    array<unsigned char>^ source,
    int sourceOffset,
    array<unsigned char>^ destination
) override
```

### Parameters

- **source**
  - Type: `array<System...::::Byte>[][][]`


- **sourceOffset**
  - Type: `System...::::Int32`

destination

Type: array<System::Byte>[][]([])

See Also

NetXtea Class
Lidgren.Network Namespace
Encrypts a block of bytes

**Namespace:** [Lidgren.Network](#)  
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#
protected override void EncryptBlock(
    byte[] source,
    int sourceOffset,
    byte[] destination
)

Visual Basic
Protected Overrides Sub EncryptBlock(_
    source As Byte(), _
    sourceOffset As Integer, _
    destination As Byte() _
)

Visual C++
protected:
virtual void EncryptBlock(
    array<unsigned char>^ source,
    int sourceOffset,
    array<unsigned char>^ destination
) override

Parameters

source
Type: array<System..::.Byte>[]()[[]]

[Missing <param name="source"/> documentation for
]

sourceOffset
Type: System..::.Int32

[Missing <param name="sourceOffset"/> documentation for
]
destination

Type: array<System::<: Byte>[][][]

[Missing <param name="destination"/> documentation for
See Also

NetXtea Class
Lidgren.Network Namespace
The NetXtea type exposes the following members.
## Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlockSize</td>
<td>Gets the block size for this cipher</td>
</tr>
<tr>
<td></td>
<td>(Overrides NetBlockEncryptionBase:::BlockSize.)</td>
</tr>
</tbody>
</table>
See Also

NetXtea Class
Lidgren.Network Namespace
Gets the block size for this cipher

**Namespace:** [Lidgren.Network](http://example.com)
**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0
(2012.1.7.0)
Syntax

C#

public override int BlockSize { get; }

Visual Basic

Public Overrides ReadOnly Property BlockSize As Integer
Get

Visual C++

public:
virtual property int BlockSize {
    int get () override;
}

See Also

NetXtea Class
Lidgren.Network Namespace
[Missing <summary> documentation for "T:Lidgren.Network.SingleUIntUnion"]

**Namespace:** Lidgren.Network

**Assembly:** Lidgren.Network (in Lidgren.Network.dll) Version: 2012.1.7.0 (2012.1.7.0)
Syntax

C#
public struct SingleUIntUnion

Visual Basic
Public Structure SingleUIntUnion

Visual C++
public value class SingleUIntUnion
See Also

SingleUIntUnion Members
Lidgren.Network Namespace
The `SingleUIntUnion` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Indicates whether this instance and a specified object are equal.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <strong>ValueType</strong>.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an <strong>Object</strong> to attempt to free resources and perform other cleanup</td>
</tr>
<tr>
<td></td>
<td>operations before the <strong>Object</strong> is reclaimed by garbage collection.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Returns the hash code for this instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <strong>ValueType</strong>.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the <strong>Type</strong> of the current instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current <strong>Object</strong>.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <strong>Object</strong>.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns the fully qualified type name of this instance.</td>
</tr>
<tr>
<td></td>
<td>(Inherited from <strong>ValueType</strong>.)</td>
</tr>
</tbody>
</table>
### Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SingleValue</td>
<td></td>
</tr>
<tr>
<td>UIntValue</td>
<td></td>
</tr>
</tbody>
</table>
See Also

SingleUIntUnion Structure
Lidgren.Network Namespace
The `SingleUIntUnion` type exposes the following members.
## Fields

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SingleValue</td>
<td></td>
</tr>
<tr>
<td>UIntValue</td>
<td></td>
</tr>
</tbody>
</table>
See Also

SingleUIntUnion Structure
Lidgren.Network Namespace
Namespace: Lidgren.Network
Syntax

C#

```csharp
public float SingleValue
```

Visual Basic

```vbc
Public SingleValue As Single
```

Visual C++

```cpp
public:
float SingleValue
```
See Also

SingleUIntUnion Structure
Lidgren.Network Namespace
Lidgren Network Library documentation

SingleUIntUnion

See Also

Send Feedback

[Missing <summary> documentation for "F:Lidgren.Network.SingleUIntUnion.UIntValue"]

Namespace: Lidgren.Network
Syntax

C#

public uint UIntValue

Visual Basic

Public UIntValue AsUInteger

Visual C++

public:
unsigned int UIntValue
See Also

SingleUIntUnion Structure
Lidgren.Network Namespace
The `SingleUIntUnion` type exposes the following members.
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equals</strong></td>
<td>Indicates whether this instance and a specified object are equal.</td>
</tr>
<tr>
<td>(Inherited from</td>
<td>ValueType.)</td>
</tr>
<tr>
<td><strong>Finalize</strong></td>
<td>Allows an Object to attempt to free resources and perform other cleanup operations before the Object is reclaimed by garbage collection. (Inherited from Object.)</td>
</tr>
<tr>
<td><strong>GetHashCode</strong></td>
<td>Returns the hash code for this instance.</td>
</tr>
<tr>
<td>(Inherited from</td>
<td>ValueType.)</td>
</tr>
<tr>
<td><strong>GetType</strong></td>
<td>Gets the Type of the current instance.</td>
</tr>
<tr>
<td>(Inherited from</td>
<td>Object.)</td>
</tr>
<tr>
<td><strong>MemberwiseClone</strong></td>
<td>Creates a shallow copy of the current Object.</td>
</tr>
<tr>
<td>(Inherited from</td>
<td>Object.)</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Returns the fully qualified type name of this instance.</td>
</tr>
<tr>
<td>(Inherited from</td>
<td>ValueType.)</td>
</tr>
</tbody>
</table>
See Also

SingleUIntUnion Structure
Lidgren.Network Namespace