

Namespace WebSockets.Core

Classes

[ClientHandshake](#)

A sans-io implementation of the client side of the WebSocket protocol.

[DateTimeProvider](#)

[Handshake](#)

The base protocol class providing functionality shared by both clients and servers.

[MessageProtocol](#)

The base protocol class providing functionality shared by both clients and servers.

[NonceGenerator](#)

[PublicExtensionMethods](#)

[ServerHandshake](#)

A sans-io implementation of the server side of the WebSocket protocol.

The business layer logic is not provided. For example when a ping is received, the implementer is expected to return the pong. This is also the case for a close.

[WebRequest](#)

[WebResponse](#)

Structs

[Reserved](#)

Interfaces

[IDateTimeProvider](#)

[INonceGenerator](#)

Enums

[HandshakeState](#)

[ProtocolState](#)

Class ClientHandshake

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

A sans-io implementation of the client side of the WebSocket protocol.

```
public class ClientHandshake : Handshake
```

Inheritance

[object](#) ← [Handshake](#) ← ClientHandshake

Inherited Members

[Handshake.State](#), [Handshake.SelectedSubProtocol](#), [Handshake.ReadData\(byte\[\], ref long, long\)](#),
[Handshake.WriteData\(byte\[\], long, long\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

Constructors

ClientHandshake(string, string[])

Construct a client handshake.

```
public ClientHandshake(string origin, string[] subProtocols)
```

Parameters

origin [string](#)

The origin is the url of the initiator of the request.

subProtocols [string](#)[]

A (possibly empty) array of requested sub-protocols.

ClientHandshake(string, string[], IDateTimeProvider, INonceGenerator)

Construct a client handshake.

```
public ClientHandshake(string origin, string[] subProtocols, IDateTimeProvider  
dateTimeProvider, INonceGenerator nonceGenerator)
```

Parameters

origin [string](#)

The origin is the url of the initiator of the request.

subProtocols [string](#)[]

A (possibly empty) array of requested sub-protocols.

dateTimeProvider [IDateTimeProvider](#)

An object which provides the current time.

nonceGenerator [INonceGenerator](#)

An object providing secrets.

Methods

ReadResponse()

Read a handshake response.

```
public WebResponse? ReadResponse()
```

Returns

[WebResponse](#)

The response from the server, or null if a complete response has yet to be received.

WriteRequest(string, string)

Write a handshake request.

```
public void WriteRequest(string path, string host)
```

Parameters

path [string](#)

The path on the server.

host [string](#)

The server name.

Class DateTimeProvider

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public class DateTimeProvider : IDateTimeProvider
```

Inheritance

[object](#) ← DateTimeProvider

Implements

[IDateTimeProvider](#)

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,
[object.ToString\(\)](#)

Properties

Now

```
public DateTime Now { get; }
```

Property Value

[DateTime](#)

Class Handshake

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

The base protocol class providing functionality shared by both clients and servers.

```
public abstract class Handshake
```

Inheritance

[object](#) ← Handshake

Derived

[ClientHandshake](#), [ServerHandshake](#)

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,
[object.ToString\(\)](#)

Constructors

Handshake(bool, string[], IDateTimeProvider)

Construct the protocol.

```
protected Handshake(bool isClient, string[] subProtocols,  
IDateTimeProvider dateServiceProvider)
```

Parameters

isClient [bool](#)

If true the protocol is for a client; otherwise it is for a server.

subProtocols [string](#)[]

The supported sub-protocols.

dateTimeProvider [IDateTimeProvider](#)

A date/time provider.

Properties

SelectedSubProtocol

The sub-protocol negotiated during the handshake.

```
public string? SelectedSubProtocol { get; protected set; }
```

Property Value

[string](#) ↗

The (possibly null) selected sub-protocol.

State

The state of the connection.

```
public HandshakeState State { get; protected set; }
```

Property Value

[HandshakeState](#)

The connection state.

Methods

ReadData(byte[], ref long, long)

Read handshake data from the provided array into the handshake buffer.

```
public void ReadData(byte[] source, ref long offset, long length)
```

Parameters

source [byte\[\]](#)

The buffer containing the data.

offset [long](#)

The offset into the buffer.

length [long](#)

The length of the data.

WriteData(byte[], long, long)

Write data from the handshake buffer into the provided array.

```
public void WriteData(byte[] destination, long offset, long length)
```

Parameters

destination [byte\[\]](#)

The array to receive the data.

offset [long](#)

The point in the buffer to start writing the data.

length [long](#)

The length of the buffer.

Enum HandshakeState

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public enum HandshakeState
```

Fields

Failed = 2

Pending = 0

Succeeded = 1

Interface IDateTimeProvider

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public interface IDateTimeProvider
```

Properties

Now

```
DateTime Now { get; }
```

Property Value

[DateTime](#)

Interface INonceGenerator

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public interface INonceGenerator
```

Methods

CreateClientKey()

```
string CreateClientKey()
```

Returns

[string](#)

CreateMask()

```
byte[] CreateMask()
```

Returns

[byte](#)[]

Class MessageProtocol

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

The base protocol class providing functionality shared by both clients and servers.

```
public class MessageProtocol
```

Inheritance

[object](#) ← MessageProtocol

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,
[object.ToString\(\)](#)

Constructors

MessageProtocol(bool, INonceGenerator)

Construct the protocol.

```
public MessageProtocol(bool isClient, INonceGenerator nonceGenerator)
```

Parameters

isClient [bool](#)

If true the protocol is for a client; otherwise it is for a server.

nonceGenerator [INonceGenerator](#)

A generator for secrets.

Properties

State

The state of the connection.

```
public ProtocolState State { get; protected set; }
```

Property Value

[ProtocolState](#)

The connection state.

Methods

ReadData(byte[], ref long, long)

```
public bool ReadData(byte[] buffer, ref long offset, long length)
```

Parameters

buffer [byte](#)[]

offset [long](#)

length [long](#)

Returns

[bool](#)

ReadMessage()

```
public Message? ReadMessage()
```

Returns

[Message](#)

WriteData(byte[], long, long)

```
public void WriteData(byte[] buffer, long offset, long length)
```

Parameters

buffer [byte](#)[]

offset [long](#)

length [long](#)

WriteMessage(Message)

```
public void WriteMessage(Message message)
```

Parameters

message [Message](#)

Class NonceGenerator

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public class NonceGenerator : INonceGenerator
```

Inheritance

[object](#) ← NonceGenerator

Implements

[INonceGenerator](#)

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,
[object.ToString\(\)](#)

Fields

_random

```
public static readonly Random _random
```

Field Value

[Random](#)

Methods

CreateClientKey()

```
public string CreateClientKey()
```

Returns

string ↴

CreateMask()

```
public byte[] CreateMask()
```

Returns

byte ↴ []

Enum ProtocolState

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public enum ProtocolState
```

Fields

Closed = 2

Closing = 1

Connected = 0

Faulted = 3

Class PublicExtensionMethods

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public static class PublicExtensionMethods
```

Inheritance

[object](#) ← PublicExtensionMethods

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,
[object.ToString\(\)](#)

Methods

SingleCommaValues(IDictionary<string, IList<string>>, string)

```
public static string[]? SingleCommaValues(this IDictionary<string, IList<string>>  
headers, string key)
```

Parameters

headers [IDictionary](#)<string, [IList](#)<string>>

key [string](#)

Returns

[string](#)[]

SingleValue(IDictionary<string, IList<string>>, string)

```
public static string? SingleValue(this IDictionary<string, IList<string>> headers,  
string key)
```

Parameters

headers [IDictionary<string, IList<string>>](#)

key [string](#)

Returns

[string](#)

Struct Reserved

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public struct Reserved : IEquatable<Reserved>
```

Implements

[IEquatable](#)<[Reserved](#)>

Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

Constructors

Reserved(bool, bool, bool)

```
public Reserved(bool isRsv1, bool isRsv2, bool isRsv3)
```

Parameters

isRsv1 [bool](#)

isRsv2 [bool](#)

isRsv3 [bool](#)

Properties

AllFalse

```
public static Reserved AllFalse { get; }
```

Property Value

Reserved

IsRsv1

```
public readonly bool IsRsv1 { get; }
```

Property Value

[bool](#) ↗

IsRsv2

```
public readonly bool IsRsv2 { get; }
```

Property Value

[bool](#) ↗

IsRsv3

```
public readonly bool IsRsv3 { get; }
```

Property Value

[bool](#) ↗

Methods

Equals(Reserved)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(Reserved other)
```

Parameters

other Reserved

An object to compare with this object.

Returns

bool ↗

true ↗ if the current object is equal to the **other** parameter; otherwise, false ↗.

Class ServerHandshake

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

A sans-io implementation of the server side of the WebSocket protocol.

The business layer logic is not provided. For example when a ping is received, the implementer is expected to return the pong. This is also the case for a close.

```
public class ServerHandshake : Handshake
```

Inheritance

[object](#) ← [Handshake](#) ← ServerHandshake

Inherited Members

[Handshake.State](#) , [Handshake.SelectedSubProtocol](#) , [Handshake.ReadData\(byte\[\], ref long, long\)](#) ,
[Handshake.WriteData\(byte\[\], long, long\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

Constructors

ServerHandshake(string[])

```
public ServerHandshake(string[] subProtocols)
```

Parameters

subProtocols [string](#)[]

ServerHandshake(string[], IDateTimeProvider)

```
public ServerHandshake(string[] subProtocols, IDateTimeProvider dateServiceProvider)
```

Parameters

`subProtocols` [string\[\]](#)[]

`dateTimeProvider` [IDateTimeProvider](#)

Methods

ReadRequest()

`public WebRequest? ReadRequest()`

Returns

[WebRequest](#)

WriteRejectResponse(string)

`public void WriteRejectResponse(string reason)`

Parameters

`reason` [string\[\]](#)

WriteResponse(WebRequest)

`public void WriteResponse(WebRequest webRequest)`

Parameters

`webRequest` [WebRequest](#)

Class WebRequest

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public class WebRequest
```

Inheritance

[object](#) ← WebRequest

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

Constructors

WebRequest(string, string, string, IDictionary<string,
IList<string>>)

```
public WebRequest(string verb, string path, string version, IDictionary<string,  
IList<string>> headers)
```

Parameters

verb [string](#)

path [string](#)

version [string](#)

headers [IDictionary<string, IList<string>>](#)

Properties

Headers

```
public IDictionary<string, IList<string>> Headers { get; }
```

Property Value

[IDictionary](#)<[string](#), [IList](#)<[string](#)>>

Path

```
public string Path { get; }
```

Property Value

[string](#)

Verb

```
public string Verb { get; }
```

Property Value

[string](#)

Version

```
public string Version { get; }
```

Property Value

[string](#)

Methods

Parse(string)

```
public static WebRequest Parse(string data)
```

Parameters

data [string](#) ↗

Returns

[WebRequest](#)

ToString()

Returns a string that represents the current object.

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

Returns

[string](#) ↗

A string that represents the current object.

Class WebResponse

Namespace: [WebSockets.Core](#)

Assembly: WebSockets.Core.dll

```
public class WebResponse
```

Inheritance

[object](#) ← WebResponse

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,
[object.ToString\(\)](#)

Constructors

WebResponse(string, int, string, IDictionary<string, IList<string>>, byte[]?)

```
public WebResponse(string version, int code, string reason, IDictionary<string, IList<string>> headers, byte[]? body)
```

Parameters

version [string](#)

code [int](#)

reason [string](#)

headers [IDictionary<string, IList<string>>](#)

body [byte\[\]](#)

Properties

Body

```
public byte[]? Body { get; }
```

Property Value

[byte](#)[]

Code

```
public int Code { get; }
```

Property Value

[int](#)

Headers

```
public IDictionary<string, IList<string>> Headers { get; }
```

Property Value

[IDictionary](#)<[string](#), [IList](#)<[string](#)>>

Reason

```
public string Reason { get; }
```

Property Value

[string](#)

Version

```
public string Version { get; }
```

Property Value

[string](#)

Methods

Parse(byte[])

```
public static WebResponse Parse(byte[] data)
```

Parameters

[data](#) [byte](#)[]

Returns

[WebResponse](#)

ToBytes()

```
public byte[] ToBytes()
```

Returns

[byte](#)[]

Namespace WebSockets.Core.Messages

Classes

[BinaryMessage](#)

[CloseMessage](#)

A message indicating the connection should be closed.

If this is an initiating message the other side will respond with a close with the same code and reason.

[DataMessage](#)

The base class for all messages which hold raw data.

[Message](#)

The base class for all WebSocket messages.

[PingMessage](#)

A ping message.

[PongMessage](#)

The message used to respond to a ping message.

[TextMessage](#)

A message with text data.

Enums

[MessageType](#)

The types of messages.

Class BinaryMessage

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

```
public class BinaryMessage : DataMessage, IEquatable<Message>,
IEquatable<DataMessage>
```

Inheritance

[object](#) ← [Message](#) ← [DataMessage](#) ← [BinaryMessage](#)

Implements

[IEquatable](#)<[Message](#)>, [IEquatable](#)<[DataMessage](#)>

Inherited Members

[DataMessage.Data](#), [DataMessage.Equals\(DataMessage\)](#), [Message.Type](#),
[Message.Serialize\(bool, Reserved?, long, INonceGenerator\)](#), [Message.Deserialize\(byte\[\]\)](#),
[Message.Equals\(Message\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

Constructors

BinaryMessage(byte[])

```
public BinaryMessage(byte[] data)
```

Parameters

data [byte](#)[]

Class CloseMessage

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

A message indicating the connection should be closed.

If this is an initiating message the other side will respond with a close with the same code and reason.

```
public class CloseMessage : Message, IEquatable<Message>, IEquatable<CloseMessage>
```

Inheritance

[object](#) ← [Message](#) ← CloseMessage

Implements

[IEquatable](#)<[Message](#)>, [IEquatable](#)<[CloseMessage](#)>

Inherited Members

[Message.Type](#), [Message.Serialize\(bool, Reserved?, long, INonceGenerator\)](#),
[Message.Deserialize\(byte\[\]\)](#), [Message.Equals\(Message\)](#), [object.Equals\(object\)](#),
[object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),
[object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

Constructors

CloseMessage(ushort?, string?)

Construct a close message.

If a reason is given a code must be specified.

```
public CloseMessage(ushort? code, string? reason)
```

Parameters

code [ushort](#)?

An optional code.

reason [string](#)

An optional reason.

Properties

Code

A code indicating the reason for the close.

```
public ushort? Code { get; }
```

Property Value

[ushort](#)?

The code.

Reason

A reason for the close.

```
public string? Reason { get; }
```

Property Value

[string](#)

The reason.

Methods

Equals(CloseMessage?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(CloseMessage? other)
```

Parameters

other [CloseMessage](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

Class DataMessage

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

The base class for all messages which hold raw data.

```
public abstract class DataMessage : Message, IEquatable<Message>,  
IEquatable<DataMessage>
```

Inheritance

[object](#) ← [Message](#) ← DataMessage

Implements

[IEquatable](#)<[Message](#)>, [IEquatable](#)<[DataMessage](#)>

Derived

[BinaryMessage](#), [PingMessage](#), [PongMessage](#)

Inherited Members

[Message.Type](#) , [Message.Serialize\(bool, Reserved?, long, INonceGenerator\)](#) ,
[Message.Deserialize\(byte\[\]\)](#) , [Message.Equals\(Message\)](#) , [object.Equals\(object\)](#) ,
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

Constructors

DataMessage(MessageType, byte[])

Construct a data message.

```
public DataMessage(MessageType type, byte[] data)
```

Parameters

type [MessageType](#)

The type of the message.

data [byte](#)[]

The message data.

Properties

Data

The data associated with the message.

```
public byte[] Data { get; }
```

Property Value

[byte](#)[]

The message data.

Methods

Equals(DataMessage?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(DataMessage? other)
```

Parameters

other [DataMessage](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

Class Message

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

The base class for all WebSocket messages.

```
public abstract class Message : IEquatable<Message>
```

Inheritance

[object](#) ← Message

Implements

[IEquatable](#)<[Message](#)>

Derived

[CloseMessage](#), [DataMessage](#), [TextMessage](#)

Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,
[object.ToString\(\)](#)

Constructors

Message(MessageType)

Construct a message.

```
protected Message(MessageType type)
```

Parameters

type [MessageType](#)

The message type.

Properties

Type

The message type.

```
public MessageType Type { get; }
```

Property Value

[MessageType](#)

The type of the message.

Methods

Deserialize(byte[])

Deserialize data into a message.

```
public static Message Deserialize(byte[] data)
```

Parameters

[data byte\[\]](#)

The data to deserialize.

Returns

[Message](#)

The deserialized message.

Equals(Message?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(Message? other)
```

Parameters

other [Message](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

Serialize(bool, Reserved?, long, INonceGenerator?)

Serialize the message to bytes.

```
public byte[] Serialize(bool isClient, Reserved? reserved = null, long maxFrameSize  
= 9223372036854775807, INonceGenerator? nonceGenerator = null)
```

Parameters

isClient [bool](#)

If true this is a client message, otherwise it is a server message.

reserved [Reserved?](#)

The reserved bits.

maxFrameSize [long](#)

The maximum size of a frame.

nonceGenerator [INonceGenerator](#)

A generator for client masks.

Returns

[byte](#) []

The message, serialized to bytes.

Enum MessageType

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

The types of messages.

```
public enum MessageType
```

Fields

Binary = 1

Close = 4

Ping = 2

Pong = 3

Text = 0

Class PingMessage

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

A ping message.

```
public class PingMessage : DataMessage, IEquatable<Message>, IEquatable<DataMessage>
```

Inheritance

[object](#) ← [Message](#) ← [DataMessage](#) ← PingMessage

Implements

[IEquatable](#)<[Message](#)>, [IEquatable](#)<[DataMessage](#)>

Inherited Members

[DataMessage.Data](#), [DataMessage.Equals\(DataMessage\)](#), [Message.Type](#),
[Message.Serialize\(bool, Reserved?, long, INonceGenerator\)](#), [Message.Deserialize\(byte\[\]\)](#),
[Message.Equals\(Message\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

Constructors

PingMessage(byte[])

Construct a ping message.

```
public PingMessage(byte[] data)
```

Parameters

data [byte](#)[]

The data that the pong message should return.

Class PongMessage

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

The message used to respond to a ping message.

```
public class PongMessage : DataMessage, IEquatable<Message>, IEquatable<DataMessage>
```

Inheritance

[object](#) ← [Message](#) ← [DataMessage](#) ← PongMessage

Implements

[IEquatable](#)<[Message](#)>, [IEquatable](#)<[DataMessage](#)>

Inherited Members

[DataMessage.Data](#), [DataMessage.Equals\(DataMessage\)](#), [Message.Type](#),
[Message.Serialize\(bool, Reserved?, long, INonceGenerator\)](#), [Message.Deserialize\(byte\[\]\)](#),
[Message.Equals\(Message\)](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

Constructors

PongMessage(byte[])

Construct a pong message.

```
public PongMessage(byte[] data)
```

Parameters

data [byte](#)[]

The data sent by the ping messages.

Class TextMessage

Namespace: [WebSockets.Core.Messages](#)

Assembly: WebSockets.Core.dll

A message with text data.

```
public class TextMessage : Message, IEquatable<Message>, IEquatable<TextMessage>
```

Inheritance

[object](#) ← [Message](#) ← TextMessage

Implements

[IEquatable](#)<[Message](#)>, [IEquatable](#)<[TextMessage](#)>

Inherited Members

[Message.Type](#), [Message.Serialize\(bool, Reserved?, long, INonceGenerator\)](#),
[Message.Deserialize\(byte\[\]\)](#), [Message.Equals\(Message\)](#), [object.Equals\(object\)](#),
[object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),
[object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

Constructors

TextMessage(string)

Construct the text message.

```
public TextMessage(string text)
```

Parameters

text [string](#)

The message text.

Properties

Text

The message text.

```
public string Text { get; }
```

Property Value

[string](#)

The text.

Methods

Equals(TextMessage?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(TextMessage? other)
```

Parameters

other [TextMessage](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).