|  |
| --- |
| [Type the company name] |
| ChatPlatform Reference |
| [Type the document subtitle] |

|  |
| --- |
| [Type the author name]  [Pick the date] |

Table of Contents

[Press ALT+A and F9 to update the TOC and other fields]

ChatPlatform Reference

Namespaces

[ChatClient](#topic_0000000000000019), [ChatPlatform](#topic_0000000000000000), [ChatPlatformUnitTest](#topic_000000000000002A)

ChatClient Namespace

This namespace holds all the references to the client application.

Classes

[Client](#topic_000000000000001A), [ClientHandler](#topic_000000000000001B), [MessageRecievedEventArgs](#topic_0000000000000027)

Enumerations

[MESSAGE\_TYPE](#topic_0000000000000023)

Client Class

This class is what contains the main execution method.

[System.Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)

**ChatClient.Client**

|  |  |
| --- | --- |
| C# |  |
| public class Client | |

Requirements

**Namespace:**[ChatClient](#topic_0000000000000019)

**Assembly:** ChatClient (in ChatClient.exe)

Methods

[Equals](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [Finalize](https://docs.microsoft.com/en-us/dotnet/api/system.object.finalize) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetHashCode](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetType](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [MemberwiseClone](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [ToString](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object))

ClientHandler Class

This class manages the connection to the server. It is contained in its own object so that multiple can be dynamically spawned if more than one client connection is needed.

[System.Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)

**ChatClient.ClientHandler**

|  |  |
| --- | --- |
| C# |  |
| public class ClientHandler | |

Requirements

**Namespace:**[ChatClient](#topic_0000000000000019)

**Assembly:** ChatClient (in ChatClient.exe)

Constructors

[ClientHandler](#topic_000000000000001D)

Methods

[Equals](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [Finalize](https://docs.microsoft.com/en-us/dotnet/api/system.object.finalize) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetHashCode](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetType](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [MemberwiseClone](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [PrintMessage](#topic_0000000000000020), [ReceiveMessageFromServer](#topic_000000000000001F), [SendLoginMessage](#topic_000000000000001E), [SendMessage](#topic_0000000000000021), [StopClient](#topic_0000000000000022), [ToString](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object))

Events

[ChatRecievedEventHandler](#topic_000000000000001C)

ClientHandler Constructor

This constructor creates an instance of the ClientHandler.

|  |  |
| --- | --- |
| C# |  |
| public ClientHandler(  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *address*,  [int](https://docs.microsoft.com/en-us/dotnet/api/system.int32) *port*,  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *username* ) | |

Parameters

address

|  |
| --- |
| The IP Address of the server |

port

|  |
| --- |
| The port the server is running on |

username

|  |
| --- |
| The username of the client |

Source code

|  |
| --- |
| public ClientHandler(string address, Int32 port, string username)  {  client = new TcpClient(address, port);  stream = client.GetStream();  this.username = username;    //Subscribes incoming events to the PrintMessage delegate.  ChatRecievedEventHandler += PrintMessage;    //Sends the login message to the server so it knows what the client username is  SendLoginMessage();  //Spawn a thread to wait for messages  Thread t = new Thread(new ThreadStart(ReceiveMessageFromServer));  t.Start();  } |

See Also

Applies to: [ClientHandler](#topic_000000000000001B)

ClientHandler.PrintMessage Method

This message allows the ChatRecievedEventHandler to print to the output window

|  |  |
| --- | --- |
| C# |  |
| public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) PrintMessage(  [object](https://docs.microsoft.com/en-us/dotnet/api/system.object) *sender*,  [EventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.eventargs) *e* ) | |

Parameters

sender

|  |
| --- |
| ClientHander responsible to invoking the ChatRecievedEventHandler event |

e

|  |
| --- |
| The MessageRecievedEventArgs sent by invoking the event |

Source code

|  |
| --- |
| public void PrintMessage(object sender, EventArgs e)  {  MessageRecievedEventArgs m = e as MessageRecievedEventArgs;  Console.WriteLine(m.message);  } |

See Also

Applies to: [ClientHandler](#topic_000000000000001B)

ClientHandler.ReceiveMessageFromServer Method

This loop runs on a separate thread so that the client can recieve messages that other clients have sent.

|  |  |
| --- | --- |
| C# |  |
| public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) ReceiveMessageFromServer() | |

Source code

|  |
| --- |
| public void ReceiveMessageFromServer()  {  while(true)  {  Byte[] data = new byte[256];  if (stream.Read(data, 0, data.Length) != 0)  {  ChatRecievedEventHandler?.Invoke(this, new MessageRecievedEventArgs(Encoding.ASCII.GetString(data).Replace("\0", "")));  }  }  } |

See Also

Applies to: [ClientHandler](#topic_000000000000001B)

ClientHandler.SendLoginMessage Method

Sends the login message to the server so it knows what the client username is.

|  |  |
| --- | --- |
| C# |  |
| public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) SendLoginMessage() | |

Source code

|  |
| --- |
| public void SendLoginMessage()  {  Byte[] data = System.Text.Encoding.ASCII.GetBytes(username + ":" + "" + "//" + MESSAGE\_TYPE.LOGIN);  stream.Write(data, 0, data.Length);  } |

See Also

Applies to: [ClientHandler](#topic_000000000000001B)

ClientHandler.SendMessage Method

Sends a message to the server.

|  |  |
| --- | --- |
| C# |  |
| public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) SendMessage(  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *s* ) | |

Parameters

s

|  |
| --- |
| The message being sent |

Source code

|  |
| --- |
| public void SendMessage(string s)  {  Byte[] data = System.Text.Encoding.ASCII.GetBytes(username + ":" + s + "//" + MESSAGE\_TYPE.MESSAGE\_SENT);  stream.Write(data, 0, data.Length);  } |

See Also

Applies to: [ClientHandler](#topic_000000000000001B)

ClientHandler.StopClient Method

This method closes the TCP and Network streams.

|  |  |
| --- | --- |
| C# |  |
| public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) StopClient() | |

Source code

|  |
| --- |
| public void StopClient()  {  stream.Close();  client.Close();  } |

See Also

Applies to: [ClientHandler](#topic_000000000000001B)

ChatRecievedEventHandler Event

This event handler keeps track of incoming messages and allows methods, like the PrintMessage() method to hook into the event called when a message is recieved.

|  |  |
| --- | --- |
| C# |  |
| public event [EventHandler](https://docs.microsoft.com/en-us/dotnet/api/system.eventhandler) ChatRecievedEventHandler | |

Source code

|  |
| --- |
| public event EventHandler ChatRecievedEventHandler; |

See Also

Applies to: [ClientHandler](#topic_000000000000001B)

MessageRecievedEventArgs Class

This EventArgs class is derived so that the client can trigger an event containing a message.

[System.Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)

[System.EventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.eventargs)

**ChatClient.MessageRecievedEventArgs**

|  |  |
| --- | --- |
| C# |  |
| public class MessageRecievedEventArgs : [EventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.eventargs) | |

Requirements

**Namespace:**[ChatClient](#topic_0000000000000019)

**Assembly:** ChatClient (in ChatClient.exe)

Constructors

[MessageRecievedEventArgs](#topic_0000000000000029)

Methods

[Equals](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [Finalize](https://docs.microsoft.com/en-us/dotnet/api/system.object.finalize) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetHashCode](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetType](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [MemberwiseClone](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [ToString](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object))

Fields

[message](#topic_0000000000000028)

MessageRecievedEventArgs Constructor

|  |  |
| --- | --- |
| C# |  |
| public MessageRecievedEventArgs(  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *message* ) | |

Parameters

message

|  |
| --- |
|  |

Source code

|  |
| --- |
| public MessageRecievedEventArgs(string message)  {  this.message = message;  } |

See Also

Applies to: [MessageRecievedEventArgs](#topic_0000000000000027)

message Field

|  |  |
| --- | --- |
| C# |  |
| public [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) message | |

Source code

|  |
| --- |
| public string message; |

See Also

Applies to: [MessageRecievedEventArgs](#topic_0000000000000027)

MESSAGE\_TYPE Enumeration

There are three kinds of messages sent to the server. Each one is encoded in the string sent to the server.

|  |  |  |
| --- | --- | --- |
| Constant | Value | Description |
| DISCONNECT | 1 | Lets the server know that the client is disconnecting. |
| LOGIN | 0 | Login message tells the server the username of the new client. |
| MESSAGE\_SENT | 2 | Lets the server know that a message intended for rebroadcast has been sent. |

Requirements

**Namespace:**[ChatClient](#topic_0000000000000019)

**Assembly:** ChatClient (in ChatClient.exe)

ChatPlatform Namespace

Classes

[ConnectionHandler](#topic_0000000000000001), [MessageRecievedEventArgs](#topic_000000000000000A), [ServerHandler](#topic_000000000000000F)

Enumerations

[MESSAGE\_TYPE](#topic_0000000000000006)

ConnectionHandler Class

This class holds the information for incoming server connections.

[System.Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)

**ChatPlatform.ConnectionHandler**

|  |  |
| --- | --- |
| C# |  |
| public class ConnectionHandler | |

Requirements

**Namespace:**[ChatPlatform](#topic_0000000000000000)

**Assembly:** ChatPlatform (in ChatPlatform.exe)

Constructors

[ConnectionHandler](#topic_0000000000000002)

Properties

[Username](#topic_0000000000000005)

Methods

[AwaitData](#topic_0000000000000003), [Equals](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [Finalize](https://docs.microsoft.com/en-us/dotnet/api/system.object.finalize) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetHashCode](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetType](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [MemberwiseClone](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [SendMeMessage](#topic_0000000000000004), [ToString](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object))

ConnectionHandler Constructor

This constructor creates a new object that holds the information of a client application.

|  |  |
| --- | --- |
| C# |  |
| public ConnectionHandler(  [TcpClient](https://docs.microsoft.com/en-us/dotnet/api/system.net.sockets.tcpclient) *client*,  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *name*,  [EventHandler](https://docs.microsoft.com/en-us/dotnet/api/system.eventhandler) *chatEventHandler*,  [UInt32](https://docs.microsoft.com/en-us/dotnet/api/system.uint32) *bufferSize* ) | |

Parameters

client

|  |
| --- |
| The incoming TcpClient pulled from the AcceptTcpClient method |

name

|  |
| --- |
| The username of the incoming connection |

chatEventHandler

|  |
| --- |
| The event handler for writing to the console |

bufferSize

|  |
| --- |
| Desired buffer size for incoming data |

Source code

|  |
| --- |
| public ConnectionHandler(TcpClient client, string name, EventHandler chatEventHandler, uint bufferSize)  {  bytes = new byte[bufferSize];  this.chatEventHandler = chatEventHandler;  this.name = name;    this.client = client;  stream = client.GetStream();  } |

See Also

Applies to: [ConnectionHandler](#topic_0000000000000001)

ConnectionHandler.Username Property

Public accessor for the username of the connection

|  |  |
| --- | --- |
| C# |  |
| public [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) Username {get; set;} | |

Source code

|  |
| --- |
| public string Username  {  get  {  return name;  }  set  {  name = value;  }  } |

See Also

Applies to: [ConnectionHandler](#topic_0000000000000001)

ConnectionHandler.AwaitData Method

This loop waits for incoming data

|  |  |
| --- | --- |
| C# |  |
| public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) AwaitData() | |

Source code

|  |
| --- |
| public void AwaitData()  {  try  {  int i;  while ((i = stream.Read(bytes, 0, bytes.Length)) != 0)  {  //Incoming data is stored in buffer  data = System.Text.Encoding.ASCII.GetString(bytes, 0, i);    //Username, message\_type, and message being parsed from the data  string name = data.Substring(0, data.IndexOf(':'));  MESSAGE\_TYPE messageType = (MESSAGE\_TYPE)Enum.Parse(typeof(MESSAGE\_TYPE), data.Substring(data.IndexOf("//")+2));  string message = data.Substring(data.IndexOf(':')+1, data.IndexOf("//") - data.IndexOf(':') - 1);    //Calls an event to write to the console  chatEventHandler?.Invoke(this, new MessageRecievedEventArgs(name, messageType, message));  }  }  catch (System.IO.IOException)  {  ServerHandler.DisconnectClient(this);  }  catch (Exception)  {  Console.WriteLine("Error Occurred");  ServerHandler.DisconnectClient(this);  }  } |

See Also

Applies to: [ConnectionHandler](#topic_0000000000000001)

ConnectionHandler.SendMeMessage Method

Sends a message back to the client

|  |  |
| --- | --- |
| C# |  |
| public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) SendMeMessage(  [byte](https://docs.microsoft.com/en-us/dotnet/api/system.byte)[] *message* ) | |

Parameters

message

|  |
| --- |
| The message being sent |

Source code

|  |
| --- |
| public void SendMeMessage(Byte[] message)  {  stream.Write(message, 0, message.Length);  } |

See Also

Applies to: [ConnectionHandler](#topic_0000000000000001)

MessageRecievedEventArgs Class

This EventArgs class is derived so that the client can trigger an event containing a username, message type, and message.

[System.Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)

[System.EventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.eventargs)

**ChatPlatform.MessageRecievedEventArgs**

|  |  |
| --- | --- |
| C# |  |
| public class MessageRecievedEventArgs : [EventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.eventargs) | |

Requirements

**Namespace:**[ChatPlatform](#topic_0000000000000000)

**Assembly:** ChatPlatform (in ChatPlatform.exe)

Constructors

[MessageRecievedEventArgs](#topic_000000000000000E)

Methods

[Equals](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [Finalize](https://docs.microsoft.com/en-us/dotnet/api/system.object.finalize) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetHashCode](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetType](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [MemberwiseClone](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [ToString](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object))

Fields

[message](#topic_000000000000000C), [sender](#topic_000000000000000B), [t](#topic_000000000000000D)

MessageRecievedEventArgs Constructor

|  |  |
| --- | --- |
| C# |  |
| public MessageRecievedEventArgs(  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *sender*,  [MESSAGE\_TYPE](#topic_0000000000000006) *t*,  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *message* ) | |

Parameters

sender

|  |
| --- |
|  |

t

|  |
| --- |
|  |

message

|  |
| --- |
|  |

Source code

|  |
| --- |
| public MessageRecievedEventArgs(string sender, MESSAGE\_TYPE t, string message)  {  this.sender = sender;  this.message = message;  this.t = t;  } |

See Also

Applies to: [MessageRecievedEventArgs](#topic_000000000000000A)

message Field

|  |  |
| --- | --- |
| C# |  |
| public [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) message | |

Source code

|  |
| --- |
| public string message; |

See Also

Applies to: [MessageRecievedEventArgs](#topic_000000000000000A)

sender Field

|  |  |
| --- | --- |
| C# |  |
| public [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) sender | |

Source code

|  |
| --- |
| public string sender; |

See Also

Applies to: [MessageRecievedEventArgs](#topic_000000000000000A)

t Field

|  |  |
| --- | --- |
| C# |  |
| public [MESSAGE\_TYPE](#topic_0000000000000006) t | |

Source code

|  |
| --- |
| public MESSAGE\_TYPE t; |

See Also

Applies to: [MessageRecievedEventArgs](#topic_000000000000000A)

ServerHandler Class

This class holds all the information for the server. It's contained in its own class so that multiple instances can be spawned if need be.

[System.Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)

**ChatPlatform.ServerHandler**

|  |  |
| --- | --- |
| C# |  |
| public static class ServerHandler | |

Requirements

**Namespace:**[ChatPlatform](#topic_0000000000000000)

**Assembly:** ChatPlatform (in ChatPlatform.exe)

Methods

[BeginAcceptConnections](#topic_0000000000000013), [Broadcast](#topic_0000000000000015), [DisconnectClient](#topic_0000000000000016), [Equals](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [Finalize](https://docs.microsoft.com/en-us/dotnet/api/system.object.finalize) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetHashCode](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetType](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [IsReady](#topic_0000000000000018), [MemberwiseClone](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [RecieveMessage](#topic_0000000000000014), [Start](#topic_0000000000000012), [Stop](#topic_0000000000000017), [ToString](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object))

Events

[ChatEventHandler](#topic_0000000000000010)

Fields

[ClientList](#topic_0000000000000011)

ServerHandler.BeginAcceptConnections Method

Allows the server to start accepting connetions

|  |  |
| --- | --- |
| C# |  |
| public static [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) BeginAcceptConnections() | |

Source code

|  |
| --- |
| public static void BeginAcceptConnections()  {  if (!acceptRunning)  {  Thread t = new Thread(new ThreadStart(PrivateBeginAcceptConnections));  t.Start();  }  } |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ServerHandler.Broadcast Method

This method sends a message back to all clients except the client that sent the message

|  |  |
| --- | --- |
| C# |  |
| public static [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) Broadcast(  [ConnectionHandler](#topic_0000000000000001) *sender*,  [string](https://docs.microsoft.com/en-us/dotnet/api/system.string) *message* ) | |

Parameters

sender

|  |
| --- |
| The client that sent the message |

message

|  |
| --- |
| The message being sent |

Source code

|  |
| --- |
| public static void Broadcast(ConnectionHandler sender, string message)  {  foreach(ConnectionHandler c in ClientList)  {  if (!c.Equals(sender))  {  Byte[] data = System.Text.Encoding.ASCII.GetBytes(message);  c.SendMeMessage(data);  }  }  } |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ServerHandler.DisconnectClient Method

This methods ensure that the disconnecting client is removed from the list.

|  |  |
| --- | --- |
| C# |  |
| public static [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) DisconnectClient(  [ConnectionHandler](#topic_0000000000000001) *h* ) | |

Parameters

h

|  |
| --- |
| Client that is disconnecting |

Source code

|  |
| --- |
| public static void DisconnectClient(ConnectionHandler h)  {  ChatEventHandler?.Invoke(h, new MessageRecievedEventArgs(h.Username, MESSAGE\_TYPE.DISCONNECT, ""));  ClientList.Remove(h);  } |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ServerHandler.IsReady Method

See if the server is ready to start

|  |  |
| --- | --- |
| C# |  |
| public static [bool](https://docs.microsoft.com/en-us/dotnet/api/system.boolean) IsReady() | |

Source code

|  |
| --- |
| public static bool IsReady()  {  return isReady;  } |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ServerHandler.RecieveMessage Method

Method called when the ChatEventHandler event is invoked

|  |  |
| --- | --- |
| C# |  |
| public static [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) RecieveMessage(  [object](https://docs.microsoft.com/en-us/dotnet/api/system.object) *sender*,  [EventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.eventargs) *e* ) | |

Parameters

sender

|  |
| --- |
| The ConnectionHandler responsible for invoking the method |

e

|  |
| --- |
| The arguments containing the username, message type, and message |

Source code

|  |
| --- |
| public static void RecieveMessage(object sender, EventArgs e)  {  MessageRecievedEventArgs m = e as MessageRecievedEventArgs;  ConnectionHandler c = sender as ConnectionHandler;    switch (m.t)  {  case MESSAGE\_TYPE.LOGIN:  c.Username = m.sender;  Console.WriteLine(c.Username + " connected.");  Broadcast(c, c.Username + " connected.");  break;  case MESSAGE\_TYPE.MESSAGE\_SENT:  Console.WriteLine(m.sender + ": " + m.message);  Broadcast(c, c.Username + ": " + m.message);  break;  case MESSAGE\_TYPE.DISCONNECT:  Console.WriteLine(m.sender + " disconnected.");  Broadcast(c, c.Username + " disconnected.");  break;  default:  break;  }    Console.WriteLine(ClientList.Count);  } |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ServerHandler.Start Method

Instantiates all required objects to run a server

|  |  |
| --- | --- |
| C# |  |
| public static [bool](https://docs.microsoft.com/en-us/dotnet/api/system.boolean) Start(  [int](https://docs.microsoft.com/en-us/dotnet/api/system.int32) *port* ) | |

Parameters

port

|  |
| --- |
| Startup port |

Returns

Returns true if the server was sucessfully setup

Source code

|  |
| --- |
| public static bool Start(int port)  {  // Subscribe the Event handler to the receive message function.  ChatEventHandler += RecieveMessage;    // Attempt to start a new TCP Listener server.  try  {  server = new TcpListener(IPAddress.Any, port);  server.Start();    // Set ready flag to true.  isReady = true;  return true;  }  catch(Exception e)  {  return false;  }  } |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ServerHandler.Stop Method

Closes the server

|  |  |
| --- | --- |
| C# |  |
| public static [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) Stop() | |

Source code

|  |
| --- |
| public static void Stop()  {  server.Stop();  } |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ChatEventHandler Event

Handles all incoming messages

|  |  |
| --- | --- |
| C# |  |
| public event [EventHandler](https://docs.microsoft.com/en-us/dotnet/api/system.eventhandler) ChatEventHandler | |

Source code

|  |
| --- |
| public static event EventHandler ChatEventHandler; |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

ClientList Field

Holds a list of connected clients

|  |  |
| --- | --- |
| C# |  |
| new public static [List](https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.list-1)<[ConnectionHandler](#topic_0000000000000001)> ClientList | |

Source code

|  |
| --- |
| public static List<ConnectionHandler> ClientList = new List<ConnectionHandler>(); |

See Also

Applies to: [ServerHandler](#topic_000000000000000F)

MESSAGE\_TYPE Enumeration

There are three kinds of messages sent to the server. Each one is encoded in the string sent to the server.

|  |  |  |
| --- | --- | --- |
| Constant | Value | Description |
| DISCONNECT | 1 | Lets the server know that the client is disconnecting. |
| LOGIN | 0 | Login message tells the server the username of the new client. |
| MESSAGE\_SENT | 2 | Lets the server know that a message intended for rebroadcast has been sent. |

Requirements

**Namespace:**[ChatPlatform](#topic_0000000000000000)

**Assembly:** ChatPlatform (in ChatPlatform.exe)

ChatPlatformUnitTest Namespace

Classes

[UnitTest1](#topic_000000000000002B)

UnitTest1 Class

[System.Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)

**ChatPlatformUnitTest.UnitTest1**

|  |  |
| --- | --- |
| C# |  |
| [TestClass()] public class UnitTest1 | |

Requirements

**Namespace:**[ChatPlatformUnitTest](#topic_000000000000002A)

**Assembly:** ChatPlatformUnitTest (in ChatPlatformUnitTest.dll)

Methods

[CheckServerStartup](#topic_000000000000002C), [ConnectionTest](#topic_000000000000002D), [Equals](https://docs.microsoft.com/en-us/dotnet/api/system.object.equals) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [Finalize](https://docs.microsoft.com/en-us/dotnet/api/system.object.finalize) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetHashCode](https://docs.microsoft.com/en-us/dotnet/api/system.object.gethashcode) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [GetType](https://docs.microsoft.com/en-us/dotnet/api/system.object.gettype) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [MemberwiseClone](https://docs.microsoft.com/en-us/dotnet/api/system.object.memberwiseclone) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object)), [TestClientNumber](#topic_0000000000000030), [TestLogin](#topic_000000000000002E), [TestTalkback](#topic_000000000000002F), [ToString](https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring) (inherited from [Object](https://docs.microsoft.com/en-us/dotnet/api/system.object))

UnitTest1.CheckServerStartup Method

|  |  |
| --- | --- |
| C# |  |
| [TestMethod()] public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) CheckServerStartup() | |

Source code

|  |
| --- |
| [TestMethod]  public void CheckServerStartup()  {  Assert.IsTrue(ChatPlatform.ServerHandler.Start(13000), "Server Unable to Start");  ChatPlatform.ServerHandler.Stop();  } |

See Also

Applies to: [UnitTest1](#topic_000000000000002B)

UnitTest1.ConnectionTest Method

|  |  |
| --- | --- |
| C# |  |
| [TestMethod()] public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) ConnectionTest() | |

Source code

|  |
| --- |
| [TestMethod]  public void ConnectionTest()  {  List<ChatPlatform.MessageRecievedEventArgs> list = new List<ChatPlatform.MessageRecievedEventArgs>();    ChatPlatform.ServerHandler.ChatEventHandler += delegate(object sender, EventArgs eventArgs)  {  ChatPlatform.MessageRecievedEventArgs m = eventArgs as ChatPlatform.MessageRecievedEventArgs;  list.Add(m);  };    ChatPlatform.ServerHandler.Start(13000);  ChatPlatform.ServerHandler.BeginAcceptConnections();  Thread.Sleep(1000);    ChatClient.ClientHandler c = new ChatClient.ClientHandler("127.0.0.1", 13000, "1");    Thread.Sleep(1000);  c.SendMessage("Test");  Thread.Sleep(1000);    Debug.WriteLine("Messages Recieved");  foreach (ChatPlatform.MessageRecievedEventArgs m in list)  Debug.WriteLine("{0}: {1} //{2}", m.sender, m.message, m.t);    Assert.IsTrue(list[1].message.Equals("Test"));    c.StopClient();  ChatPlatform.ServerHandler.Stop();  } |

See Also

Applies to: [UnitTest1](#topic_000000000000002B)

UnitTest1.TestClientNumber Method

|  |  |
| --- | --- |
| C# |  |
| [TestMethod()] public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) TestClientNumber() | |

Source code

|  |
| --- |
| [TestMethod]  public void TestClientNumber()  {  List<ChatClient.ClientHandler> list = new List<ChatClient.ClientHandler>();    ChatPlatform.ServerHandler.Start(13000);  ChatPlatform.ServerHandler.BeginAcceptConnections();  Thread.Sleep(1000);    try  {  while (list.Count < 1000)  {  ChatClient.ClientHandler c1 = new ChatClient.ClientHandler("127.0.0.1", 13000, "TESTUSER1");  list.Add(c1);  }  }  catch(Exception e)  {  Debug.WriteLine(e.Message);  }      } |

See Also

Applies to: [UnitTest1](#topic_000000000000002B)

UnitTest1.TestLogin Method

|  |  |
| --- | --- |
| C# |  |
| [TestMethod()] public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) TestLogin() | |

Source code

|  |
| --- |
| [TestMethod]  public void TestLogin()  {  ChatPlatform.ServerHandler.Start(13000);  ChatPlatform.ServerHandler.BeginAcceptConnections();  Thread.Sleep(1000);    ChatClient.ClientHandler c = new ChatClient.ClientHandler("127.0.0.1", 13000, "TESTUSER");  Thread.Sleep(1000);    Assert.IsTrue(ChatPlatform.ServerHandler.ClientList[0].Username.Equals("TESTUSER"));    c.StopClient();  ChatPlatform.ServerHandler.Stop();  } |

See Also

Applies to: [UnitTest1](#topic_000000000000002B)

UnitTest1.TestTalkback Method

|  |  |
| --- | --- |
| C# |  |
| [TestMethod()] public [void](https://docs.microsoft.com/en-us/dotnet/api/system.void) TestTalkback() | |

Source code

|  |
| --- |
| [TestMethod]  public void TestTalkback()  {  List<ChatClient.MessageRecievedEventArgs> list = new List<ChatClient.MessageRecievedEventArgs>();    ChatPlatform.ServerHandler.Start(13000);  ChatPlatform.ServerHandler.BeginAcceptConnections();  Thread.Sleep(1000);    ChatClient.ClientHandler c1 = new ChatClient.ClientHandler("127.0.0.1", 13000, "TESTUSER1");  ChatClient.ClientHandler c2 = new ChatClient.ClientHandler("127.0.0.1", 13000, "TESTUSER2");  Thread.Sleep(1000);    c2.ChatRecievedEventHandler += delegate (object sender, EventArgs eventArgs)  {  ChatClient.MessageRecievedEventArgs m = eventArgs as ChatClient.MessageRecievedEventArgs;  list.Add(m);  };    c1.SendMessage("TESTMESSAGE");  Thread.Sleep(1000);    foreach(ChatClient.MessageRecievedEventArgs m in list)  Debug.WriteLine("{0}", m.message);    Assert.IsTrue("TESTUSER1: TESTMESSAGE".Equals(list[0].message));    c1.StopClient();  c2.StopClient();  ChatPlatform.ServerHandler.Stop();  } |

See Also

Applies to: [UnitTest1](#topic_000000000000002B)

# Index

[AwaitData Method](#topic_0000000000000003)

[BeginAcceptConnections Method](#topic_0000000000000013)

[Broadcast Method](#topic_0000000000000015)

[ChatClient Namespace](#topic_0000000000000019)

[ChatEventHandler Event](#topic_0000000000000010)

[ChatPlatform Namespace](#topic_0000000000000000)

[ChatPlatform Reference](#topic_0000000000000031)

[ChatPlatformUnitTest Namespace](#topic_000000000000002A)

[ChatRecievedEventHandler Event](#topic_000000000000001C)

[CheckServerStartup Method](#topic_000000000000002C)

[Client Class](#topic_000000000000001A)

[ClientHandler Class](#topic_000000000000001B)

[ClientHandler Constructor](#topic_000000000000001D)

[ClientList Field](#topic_0000000000000011)

[ConnectionHandler Class](#topic_0000000000000001)

[ConnectionHandler Constructor](#topic_0000000000000002)

[ConnectionTest Method](#topic_000000000000002D)

[DisconnectClient Method](#topic_0000000000000016)

[IsReady Method](#topic_0000000000000018)

[MESSAGE\_TYPE Enumeration {ChatClient}](#topic_0000000000000023)

[MESSAGE\_TYPE Enumeration {ChatPlatform}](#topic_0000000000000006)

[MessageRecievedEventArgs Class {ChatClient}](#topic_0000000000000027)

[MessageRecievedEventArgs Class {ChatPlatform}](#topic_000000000000000A)

[MessageRecievedEventArgs Constructor {ChatClient.MessageRecievedEventArgs}](#topic_0000000000000029)

[MessageRecievedEventArgs Constructor {ChatPlatform.MessageRecievedEventArgs}](#topic_000000000000000E)

[PrintMessage Method](#topic_0000000000000020)

[ReceiveMessageFromServer Method](#topic_000000000000001F)

[RecieveMessage Method](#topic_0000000000000014)

[SendLoginMessage Method](#topic_000000000000001E)

[SendMeMessage Method](#topic_0000000000000004)

[SendMessage Method](#topic_0000000000000021)

[ServerHandler Class](#topic_000000000000000F)

[Start Method](#topic_0000000000000012)

[Stop Method](#topic_0000000000000017)

[StopClient Method](#topic_0000000000000022)

[TestClientNumber Method](#topic_0000000000000030)

[TestLogin Method](#topic_000000000000002E)

[TestTalkback Method](#topic_000000000000002F)

[UnitTest1 Class](#topic_000000000000002B)

[Username Property](#topic_0000000000000005)

[message Field {ChatClient.MessageRecievedEventArgs}](#topic_0000000000000028)

[message Field {ChatPlatform.MessageRecievedEventArgs}](#topic_000000000000000C)

[sender Field](#topic_000000000000000B)

[t Field](#topic_000000000000000D)