1. WebSocket在建立握手时，数据是通过http传输的，但是在建立后，在真正传输的时候是不需要http协议的，借助于tcp传输信道进行双攻通信。
2. Websocket和http都是基于tcp协议的 都是可靠性传输协议
   1. 都是应用层的协议
3. websocket是双向通信协议，http是单向的
4. 因为第一次是需要通过http协议连接，所以第一次也是需要三次握手的
5. 应用场景，弹幕，媒体聊天，实时更新，协同编辑等一些实时的功能

/// 客户端代码

//ws = new WebSocket("ws://10.10.11.51:1000");

//ws.onopen = function()

//{

// ws.send('websocekt测试');

//};

//ws.onmessage = function(e)

//{

// alert("收到服务端的消息：" + e.data);

//};

/// </summary>

public static void WebSocketApp()

{

FleckLog.Level = LogLevel.Debug;

var allSockets = new List<IWebSocketConnection>();

var server = new WebSocketServer("ws://10.10.11.51:1000");

server.Start(socket =>

{

//客户端连接上服务端，则在socket列表添加一个客户端

socket.OnOpen = () =>

{

Console.WriteLine("Open!");

allSockets.Add(socket);

};

//客户端离线，则从socket列表删除这个客户端

socket.OnClose = () =>

{

Console.WriteLine("Close!");

allSockets.Remove(socket);

};

socket.OnMessage = message =>

{

Console.WriteLine("接收客户端的消息:"+message);

};

});

//服务端向客户端传输数据

var input = Console.ReadLine();

while (input != "exit")

{

foreach (var socket in allSockets.ToList())

{

socket.Send(input);

}

input = Console.ReadLine();

}

}

1. 应用场景：服务端数据更新，需要通知客户端（双工通信--客户端和服务端即是发送者也是接受者--互相发送、互相接收） 比较适合高及时性服务
   1. 聊天软件
   2. 订阅
   3. 游戏
   4. 协同工作（文本编辑 ）
   5. 直播
   6. 股票基金
   7. 基于位置的应用