

Documentação **WebSockets-WebChat**

PROJETO RC

ARTUR QUARESMA Nº1 12ºM

Conteúdo

WebSocket	2
Startup.cs	3
MessageHub.cs	4
Index.cshtml	5
Chat.js	7
Referencias	10

WebSocket

Utilizei ASP.NET para o projeto de WebSocket e tive utilizar a biblioteca **SignalR** que permite usar funcionalidade WEB em tempo real e assíncrona

The SignalR logo features a small icon of concentric circles with a central square, resembling a signal or ripple effect, positioned above the letter 'i' in the word 'SignalR'.

SignalR



Startup.cs

```
namespace WebSocket_CHAT
{
    1 reference
    public class Startup
    {
        0 references
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddMvc();
            services.AddSignalR();
        }

        0 references
        public void Configure(IApplicationBuilder app, IHostingEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }

            app.UseStaticFiles();
            app.UseSignalR(config => {
                config.MapHub<MessageHub>("/messages");
            });
            app.UseMvc();
        }
    }
}
```

Ficheiro que inicia os serviços pretendidos

MessageHub.cs

```
public class MessageHub : Hub
{
    public static List<string> Utilizadores { get; set; } = new List<string>();

    public async Task NewUser(string user)
    {
        foreach (var elem in Utilizadores)
        {
            await Clients.Caller.SendAsync("NewUser", elem);
        }

        if (!Utilizadores.Contains(user))
        {
            Utilizadores.Add(user);
            await Clients.All.SendAsync("NewUser", user);
        }
    }

    public Task SendMessageToAll(string message)
    {
        return Clients.All.SendAsync("ReceiveMessage", message);
    }

    public Task SendMessageToCaller(string message)
    {
        return Clients.Caller.SendAsync("ReceiveMessage", message);
    }

    public Task JoinGroup(string group)
    {
        return Groups.AddToGroupAsync(Context.ConnectionId, group);
    }

    public Task SendMessageToGroup(string group, string message)
    {
        return Clients.Group(group).SendAsync("ReceiveMessage", message);
    }

    public override async Task OnConnectedAsync()
    {
        await Clients.All.SendAsync("UserConnected", Context.ConnectionId);
        await base.OnConnectedAsync();
    }

    public override async Task OnDisconnectedAsync(Exception ex)
    {
        await Clients.All.SendAsync("UserDisconnected", Context.ConnectionId);
        await base.OnDisconnectedAsync(ex);
    }
}
```

Ficheiro C# para controlar eventos das mensagens, grupos e utilizadores.

Autor: Derek Comartin

Link do trabalho: <https://codeopinion.com/practical-asp-net-core-signalr-overview/>

Index.cshtml

@page

```
<link rel="stylesheet" type="text/css" href="~/css/style.css">
<link href="~/css/font-awesome.min.css" rel="stylesheet" />

<h1 class="titulo">WebChat Signalr</h1>

<!-- users-->
<div class="users col-sm-3">

    <div class="input-group col-sm-12">
        <h3><span>Utilizadores Online</span></h3>
    </div>

    <div class="panel panel-default">

        <div class="panel-body fixed-panel">
            <ul class="list-group" id="usersList"></ul>
            <div id="userList"></div>
        </div>

        Nome: <span id='usernameSpan'>__</span>

        <div class="panel-footer" id="nameFooter">
            <div class="input-group col-sm-12">

                <input id="usernameInput" type="text" class="form-
control" placeholder="Username..." maxlength="16">

                <div class="input-group-btn">
                    <button class="btn btn-success" id="sendUserName">Username</button>
                </div>
            </div>
        </div>
    </div>
</div>

<!-- chat-->
<div class="chat col-sm-9">

    <div class="grupos">
        <input type="text" id="nomeGrupo" value="" />
        <input type="button" id="joinGroup" value="Join Private Group" />
    </div>
</div>
```

```

</div>

<br />
<br />
<br />

<!-- ChatArea -->

<div class="panel panel-default">

    <div class="panel-body fixed-panel chat">
        <ul class="list-group" id="messages"></ul>
    </div>

    <div class="panel-footer">
        <div class="input-group col-sm-12">

            <select id="group">
                <option value="All">Everyone</option>
            </select>

            <textarea id="message" name="message" class="form-
control" placeholder="Mensagem" rows="1"></textarea>
            <div class="input-group-btn">
                <input class="buttonSend" type="button" id="sendButton" value="Enviar" />
            </div>

        </div>
    </div>
</div>

<script src="~/js/jquery-3.3.1.js"></script>
<script src="~/lib/aspnet/signalr/dist/browser/signalr.js"></script>
<script src="~/js/chat.js"></script>
<script src="~/js/jscolor.js"></script>

```

Chat.js

```
"use strict";

var connection = new signalR.HubConnectionBuilder().withUrl("/messages").build();
var username;
```

Criação de Variáveis globais

connection é utilizada para todas as conexões de utilizadores

username criado para atribuição do nome do utilizador

```
function tempoMsg() {
    var data = new Date();
    return " " + data.getHours() + ":" + data.getMinutes() + ":" + data.getSeconds() + " ";
}

function isEmptyOrSpaces(str) {
    return str === null || str.match(/^ *$/) !== null;
}
```

Funções globais

tempoMsg → função para indicar o tempo da mensagem

isEmptyOrSpaces() → função para detetar espaço branco ou vazio

```
//registo do utilizador
$("#sendUserName").click(function () {
    username = $('#usernameInput:text').val();

    if (isEmptyOrSpaces(username))
        username = "Anonymous";

    $('#usernameSpan').html(username);

    //dá-se o início da conexão
    connection.start().catch(function (err) {
        return console.error(err.toString());
    });

    connection.invoke("NewUser", username).catch(function (err) {
        return console.error(err.toString());
    });

    $connection.invoke("SendMessageToAll", username);

});
```


Registo do utilizador e é feita a conexão

```
//receber mensagens
connection.on("ReceiveMessage", function (message) {
    var msg = message.replace(/&/g, "&amp;").replace(/</g, "&lt;").replace(/>/g, "&gt;");
    var div = document.createElement("div");
    div.innerHTML = msg + "<hr/>";
    document.getElementById("messages").appendChild(div);
    div.scrollIntoView();
});
```

Recebe as mensagens e adiciona na view

```
//utilizador conecta
connection.on("UserConnected", function (connectionId) {

    connection.invoke("SendMessageToAll", ("" + username + " ON"));

    //adiciona User nos User ON
    var userTag = document.createElement("li");
    userTag.textContent = username;
    document.getElementById("userList").appendChild(userTag);
});

//utilizador desconecta
connection.on("UserDisconnected", function (connectionId) {

    connection.invoke("SendMessageToAll", ("" + username + " OFF"));

    //remove User nos User ON
    var userTag = document.createElement("li");
    userTag.textContent = username;
    document.getElementById("userList").appendChild(userTag + "off");
});
```

Funções de controlo dos utilizador que conectam e desconectam

```
//codigo alteracao DOM
$("#sendButton").click(function () {
    var message = $('#message').val();

    if(isEmptyOrSpaces(message))
        message = "-----";
```

```

var groupElement = document.getElementById("group");
var groupValue = groupElement.options[groupElement.selectedIndex].value;

if (groupValue === "All") {
    // var method = groupValue === "All" ? "SendMessageToAll" : "SendMessageToCaller";
    connection.invoke("SendMessageToAll", username + " (" + tempoMsg() + ")-- > " + message).catch(function (err) {
        return console.error(err.toString());
    });
}
else {
    connection.invoke("SendMessageToGroup", groupValue, " (" + tempoMsg() + ")-- > " + message).catch(function (err) {
        return console.error(err.toString());
    });
}
e.preventDefault();
});

```

Envio de mensagens e deteta se é a partir de um grupo privado ou do chat geral

```

$("#joinGroup").click(function (event) {

    var nomeGrupo = $('#nomeGrupo:text').val();

    var option = document.createElement("option");
    option.text = nomeGrupo;
    option.value = nomeGrupo;
    document.getElementById("group").add(option); //grupo de elementos

    connection.invoke("JoinGroup", nomeGrupo).catch(function (err) {
        return console.error(err.toString());
    });
    event.preventDefault();
});

```

Criacao ou juncao de um grupo

Referencias

<https://docs.microsoft.com/en-us/aspnet/signalr/overview/guide-to-the-api/working-with-groups>

<https://docs.microsoft.com/en-us/aspnet/signalr/>

<https://github.com/YarkoKhamar/Chat>

http://www.icons101.com/icon/id_67331/setid_2086/Flat_Round_by_Roundicons/Chat

<https://stackoverflow.com/questions/10232366/how-to-check-if-a-variable-is-null-or-empty-string-or-all-whitespace-in-javascript>