Введение в «SignalR»

Александр Кузнецов

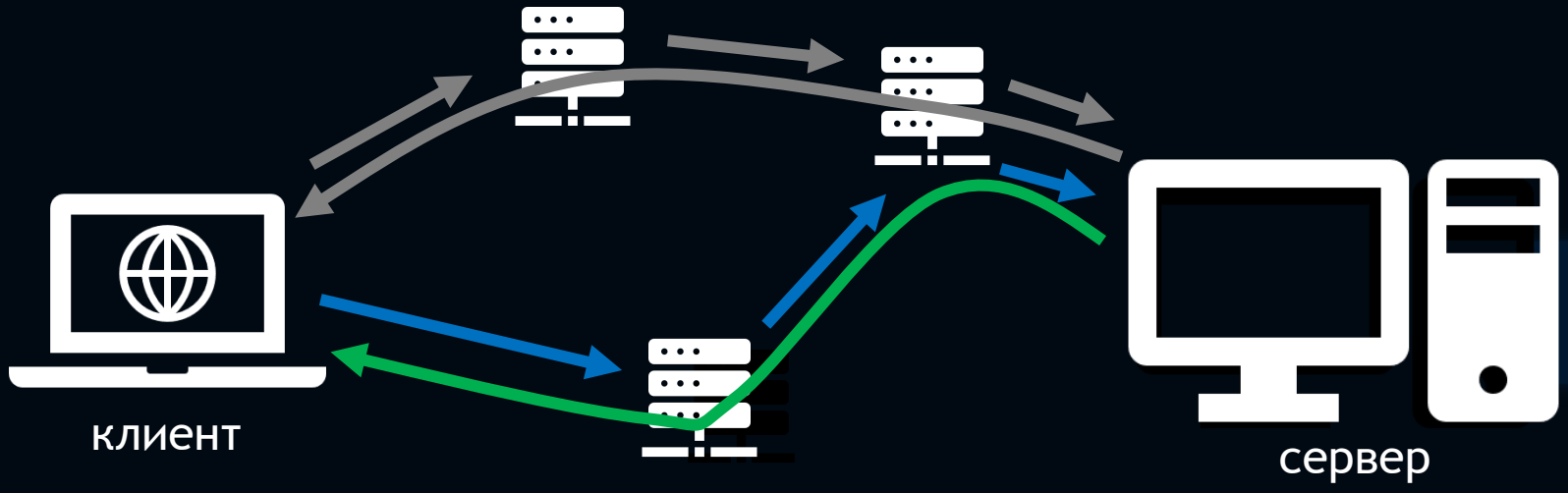
СКБ Контур

DOTNEXT

Принципы работы веб-приложений

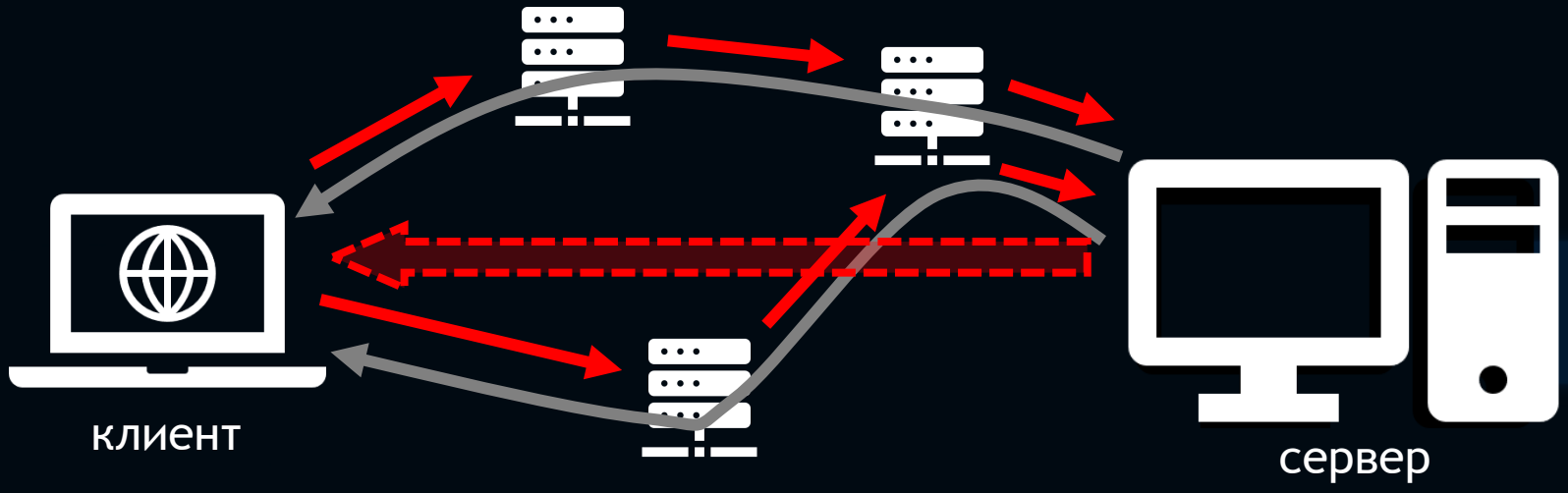
Клиент связывается с сервером через цепочку промежуточных узлов и получает ответ

Принципы работы веб-приложений



Новый запрос может пойти через другую цепочку узлов

Проблемы



1. Установка соединения не быстрая операция
2. Сервер не может инициировать запрос

И решения

1. Постоянное двухстороннее соединение

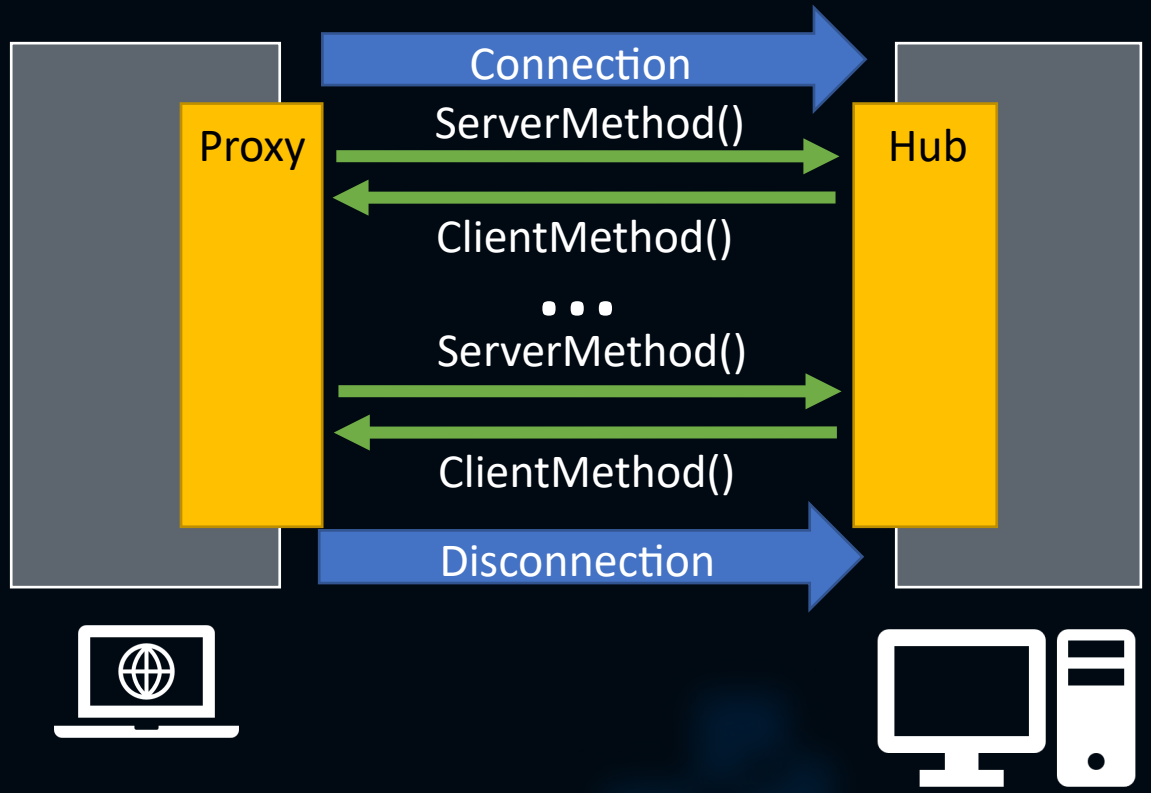
И решения

1. Постоянное двухстороннее соединение
2. Технологии:
   - Long polling,
   - Server Sent Events,
   - Forever Frame,
   - WebSocket

И решения

1. Постоянное двухстороннее соединение
2. Технологии:
   - Long polling,
   - Server Sent Events,
   - Forever Frame,
   - WebSocket
3. Библиотека:
   - Microsoft SignalR

Принцип работы



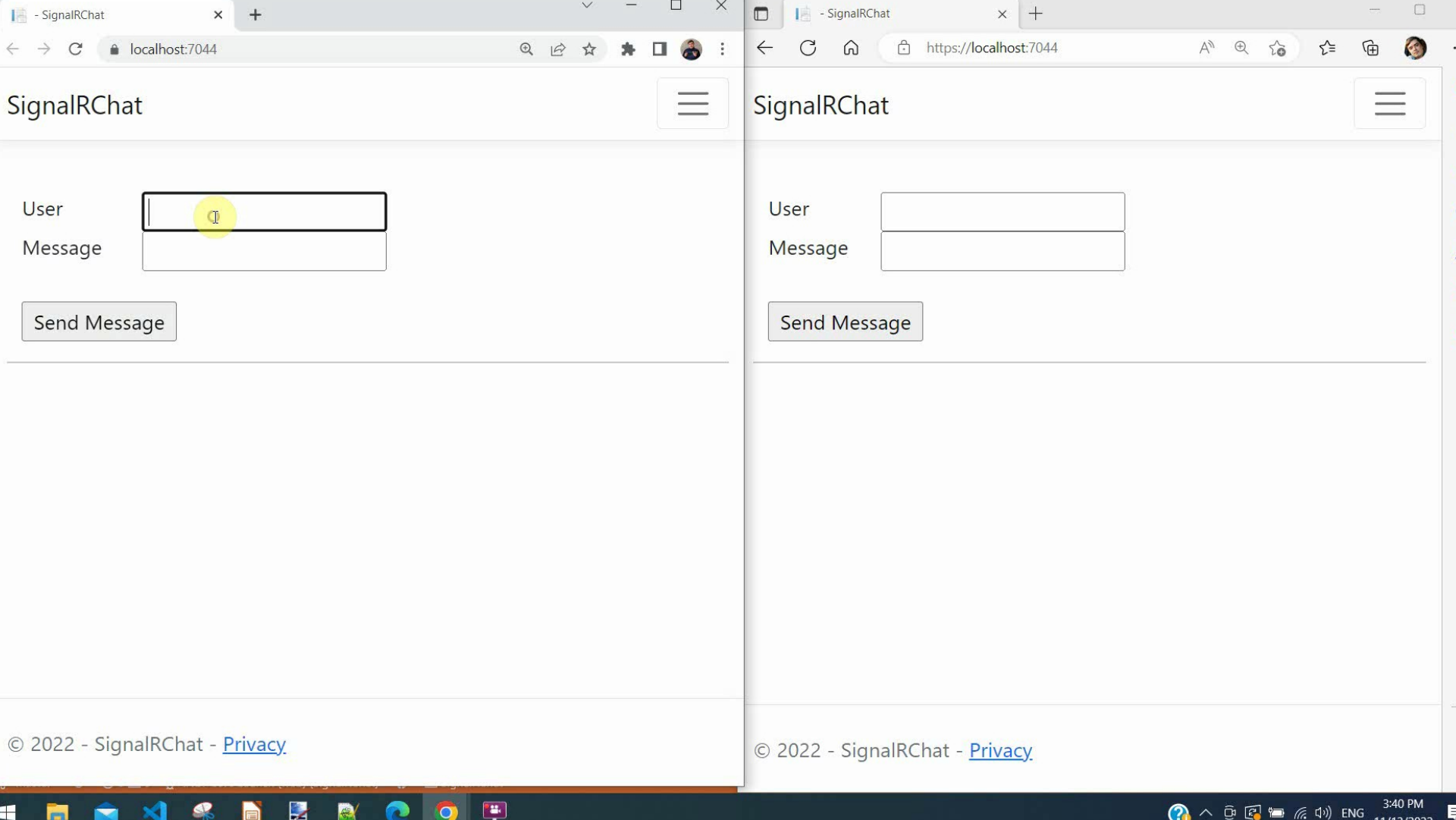
От теории к практике

Версии ПО:

- Visual Studio Code 1.73.0
- .NET 7.0.100 SDK
- SignalR for ASPNetCore
- JQuery 3.5.1
- Sqlite 3.35.5

https://learn.microsoft.com/ru-ru/aspnet/core/tutorials/signalr

Принцип работы чата


- SignalRChat
localhost:7044
SignalRChat
User
Message
Send Message
© 2022 - SignalRChat - Privacy
- SignalRChat
https://localhost:7044
SignalRChat
User
Message
Send Message
© 2022 - SignalRChat - Privacy
ENG
3:40 PM

Код хаба

Hubs/ChatHub.cs

```
using Microsoft.AspNetCore.SignalR;

namespace SignalRChat.Hubs;

1 reference
public class ChatHub : Hub
{
    0 references
    public async Task SendMessage(string user, string message)
    {
        await Clients.All.SendAsync("ReceiveMessage", user, message);
    }
}
```

Program.cs

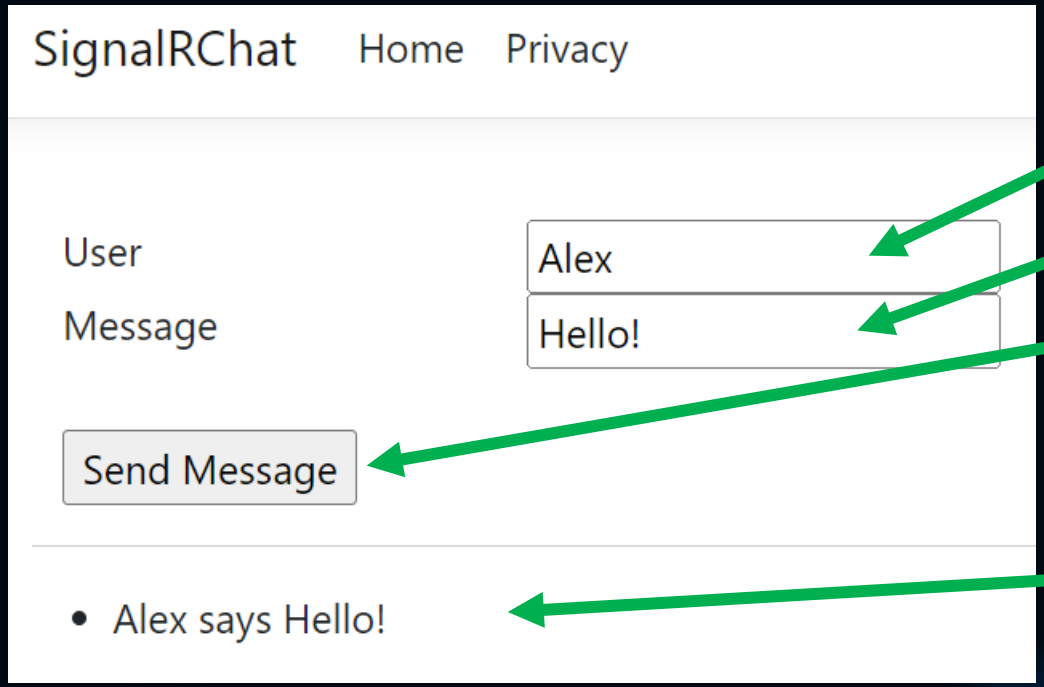
```
app.MapHub<ChatHub>("/chatHub");
```

Строго типизированные хабы

```
1 reference
public interface IChatClient
{
    1 reference
    Task ReceiveMessage(string user, string message);
}
```

```
1 reference
public class ChatHub : Hub<IChatClient>
{
    0 references
    public async Task SendMessage(string user, string message)
    {
        await Clients.All.ReceiveMessage(user, message);
    }
}
```

Клиент



#userInput

#messageInput

#sendButton

#messagesList

Клиент, установка

```
dotnet tool install -g Microsoft.Web.LibraryManager.Cli

libman install @microsoft/signalr@latest -p unpkg -d wwwroot/js/signalr
libman install jquery@latest -p unpkg
```

```
<script src="~/js/signalr/dist/browser/signalr.js"></script>
<script src="~/js/chat.js"></script>
```

Клиент, установка соединения

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


connection.start().then(function () {
    document.getElementById("sendButton").disabled = false;
}).catch(function (err) {
    return console.error(err.toString());
});
```

Клиент, обработчик вызова сервера

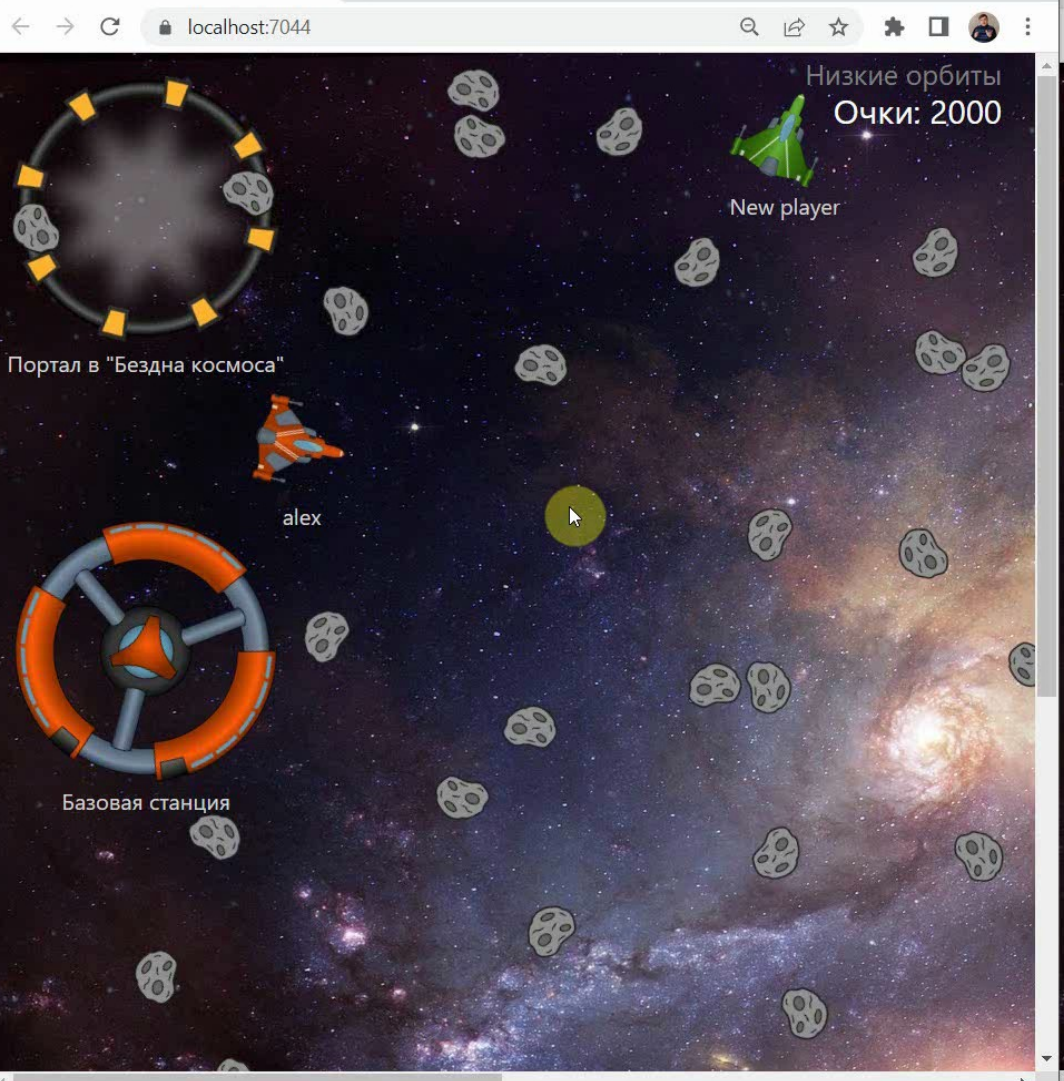
```
connection.on("ReceiveMessage", function (user, message) {
    var li = document.createElement("li");
    document.getElementById("messagesList").appendChild(li);
    li.textContent = `${user} says ${message}`;
});
```

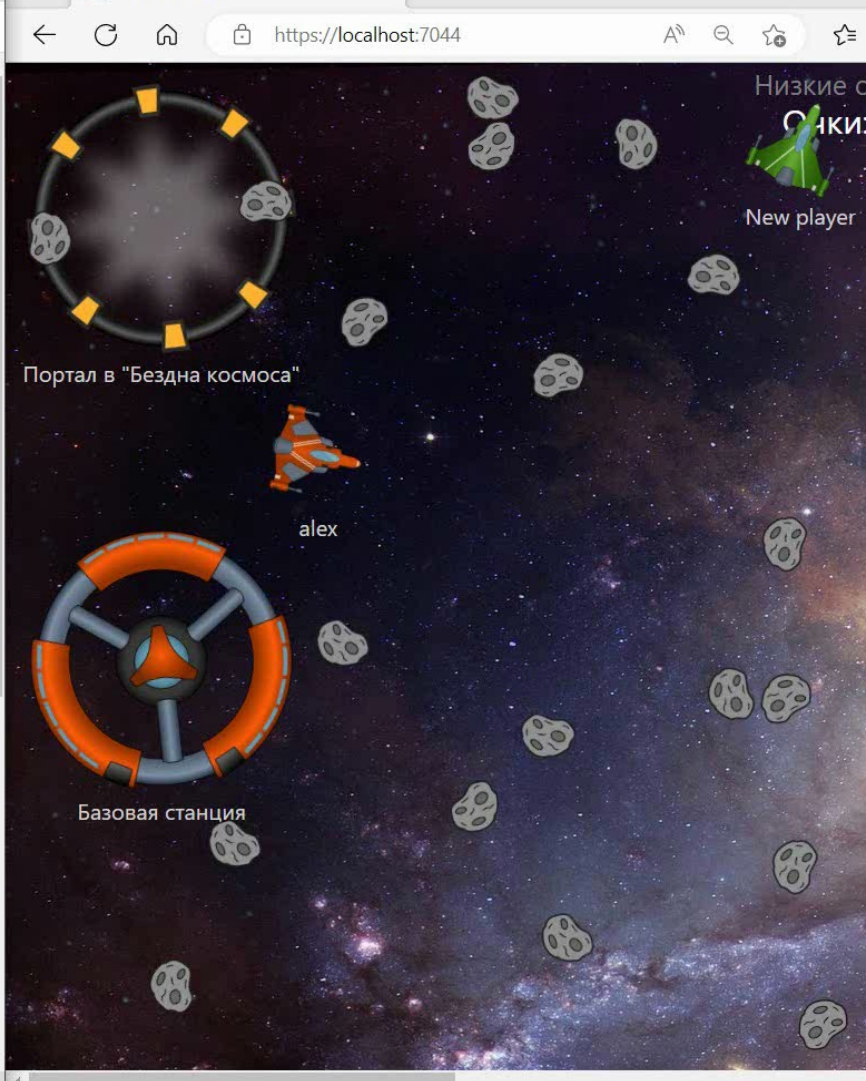
Клиент, вызов сервера

```
document.getElementById("sendButton")
    .addEventListener("click", function (event) {
        var user = document.getElementById("userInput").value;
        var message = document.getElementById("messageInput").value;
        connection.invoke("SendMessage", user, message)
            .catch(function (err) {
                return console.error(err.toString());
            });
        event.preventDefault();
    });
```

Пример 2. Игра

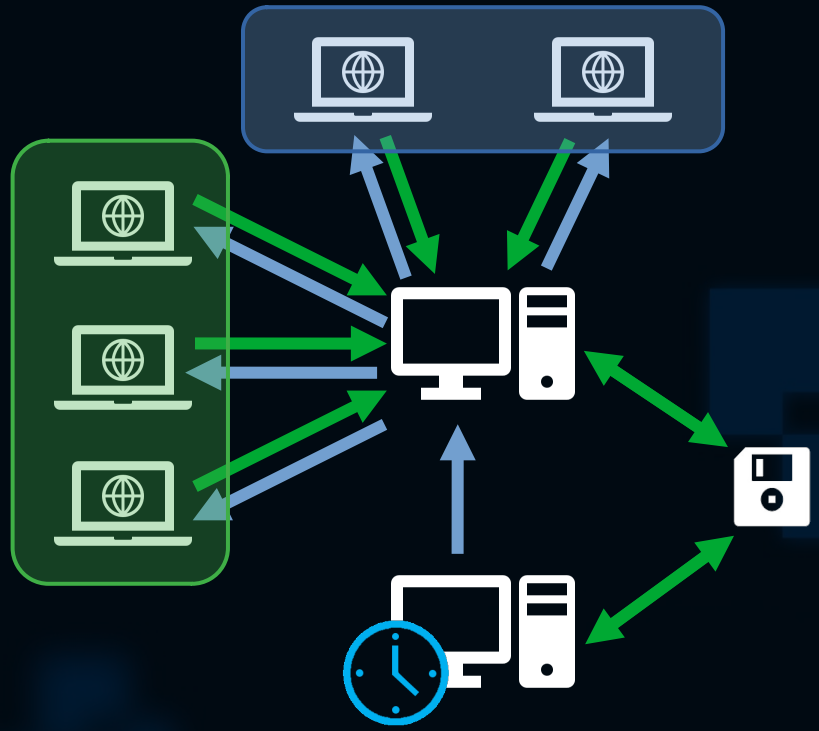
- Корабли летают и подбирают астероиды
- За подбор астероидов идёт набор очков
- Подобранные астероиды периодически обновляются
- Можно перемещаться по игровым уровням


localhost:7044
Низкие орбиты
Очки: 2000
New player
Портал в "Бездна космоса"
alex
Базовая станция


https://localhost:7044
New player
Портал в "Бездна космоса"
alex
Базовая станция

Особенности архитектуры

- Клиент передаёт действия пользователя и показывает результат
- Все расчёты делает сервер
- Начальный объём данных может быть большим
- Обновлением информации и перерасчётами занимается отдельное приложение



Соединения, группы и не только...

```
await Clients.All.SendAsync("ReceiveMessage", user, message);
```

Соединения, группы и не только...

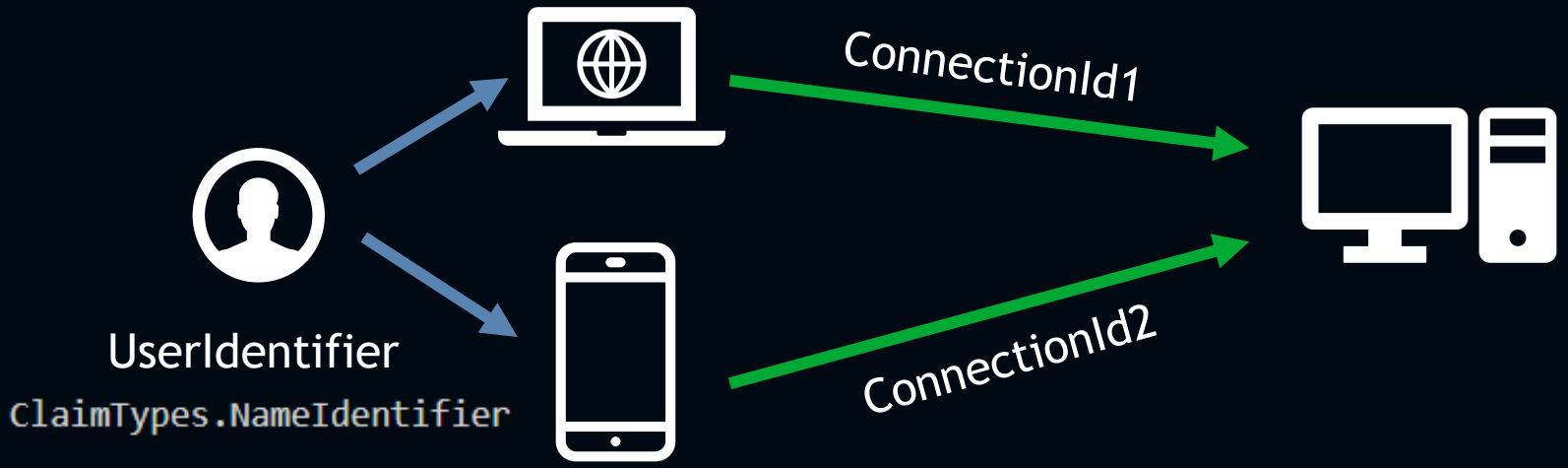
```
await Clients.All.SendAsync("ReceiveMessage", user, message);
```

```
Context.ConnectionId
```

```
await Clients.AllExcept(Context.ConnectionId).ReceiveMessage(user, message);
await Clients.Caller.ReceiveMessage(user, message);
await Clients.Others.ReceiveMessage(user, message);
await Clients.Clients(Context.ConnectionId).ReceiveMessage(user, message);
```

Соединения, группы и не только...

```
await Clients.Users(Context.UserIdentifier).ReceiveMessage(user, message);
```

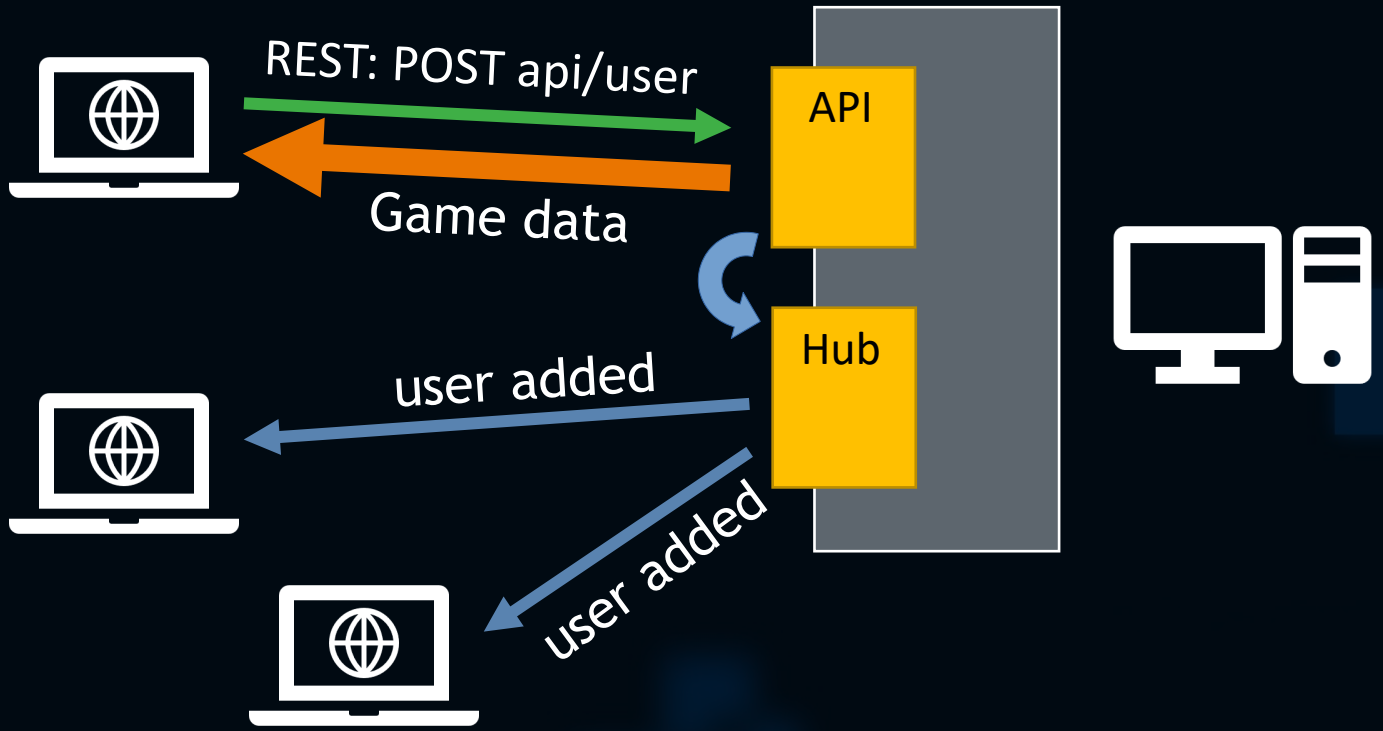


Соединения, группы и не только...

```
await Groups.AddToGroupAsync(Context.ConnectionId, "GroupName");
await Groups.RemoveFromGroupAsync(Context.ConnectionId, "GroupName");
```

```
await Clients.Group("GroupName").ReceiveMessage(user, message);
await Clients.OthersInGroup("GroupName").ReceiveMessage(user, message);
await Clients.GroupExcept("GroupName", Context.ConnectionId)
    .ReceiveMessage(user, message);
```

Вызов методов хаба из контроллера



Вызов методов хаба из контроллера

```
public class UserController : ControllerBase
{
    4 references
    private readonly IUserService _userService;
    3 references
    private readonly IHubContext<GameHub> _gameHub;
    0 references
    public UserController(IHubContext<GameHub> gameHub, IUserService userService)
    {
        _gameHub = gameHub;  _userService = userService;
    }
    [HttpPost]
    0 references
    public async Task<User> CreateAsync([FromBody]User user)
    {
        var createdUser = await _userService.CreateUserAsync(user);
        await _gameHub.Clients.Group(user.Ship.LevelId.ToString()).SendAsync("NewUser", user);
        return createdUser;
    }
```

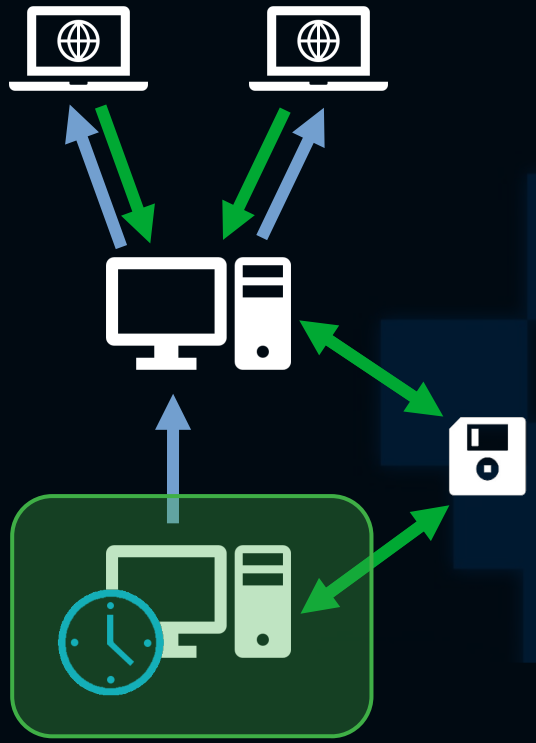
IHubContext<GameHub> gameHub

.NET client

```
dotnet add package Microsoft.AspNetCore.SignalR.Client
```

```
HubConnection connection;
connection = new HubConnectionBuilder()
        .WithUrl("https://localhost:7001/gameHub")
        .WithAutomaticReconnect()
        .AddJsonProtocol(opt =>
        {
            opt.PayloadSerializerOptions.ReferenceHandler =
                ReferenceHandler.IgnoreCycles;
        })
        .Build();
```

6

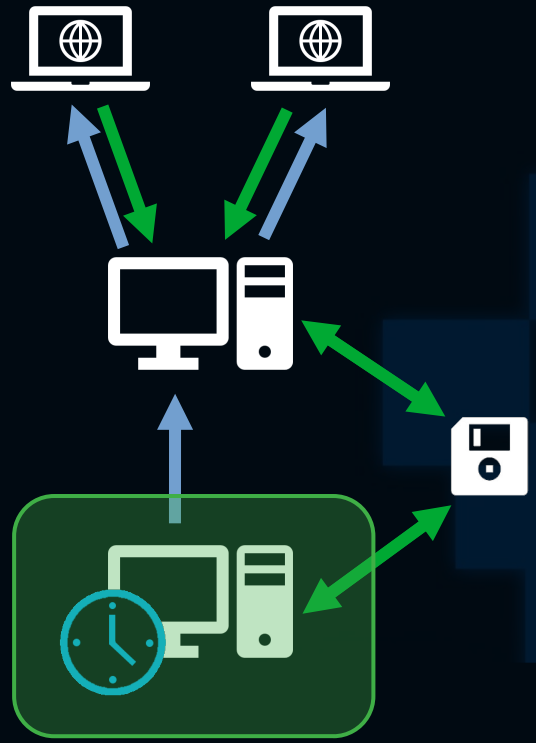


.NET client

```
await connection.StartAsync()
    .WaitAsync(TimeSpan.FromSeconds(5));
if(connection.State != HubConnectionState.Connected)
    throw new TimeoutException();
```

```
connection.On<string>("MethodName",
    (p) => { Console.WriteLine(p); });
```

```
await connection.InvokeAsync("AsteroidAddedAsync",
    asteroid);
```



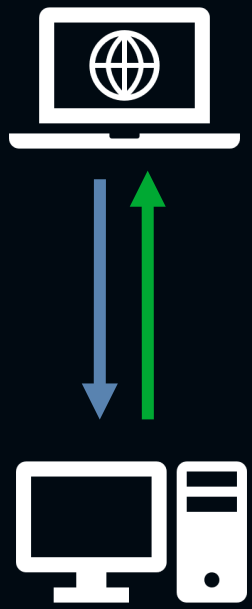
Ограничения SignalR

Данные от клиента

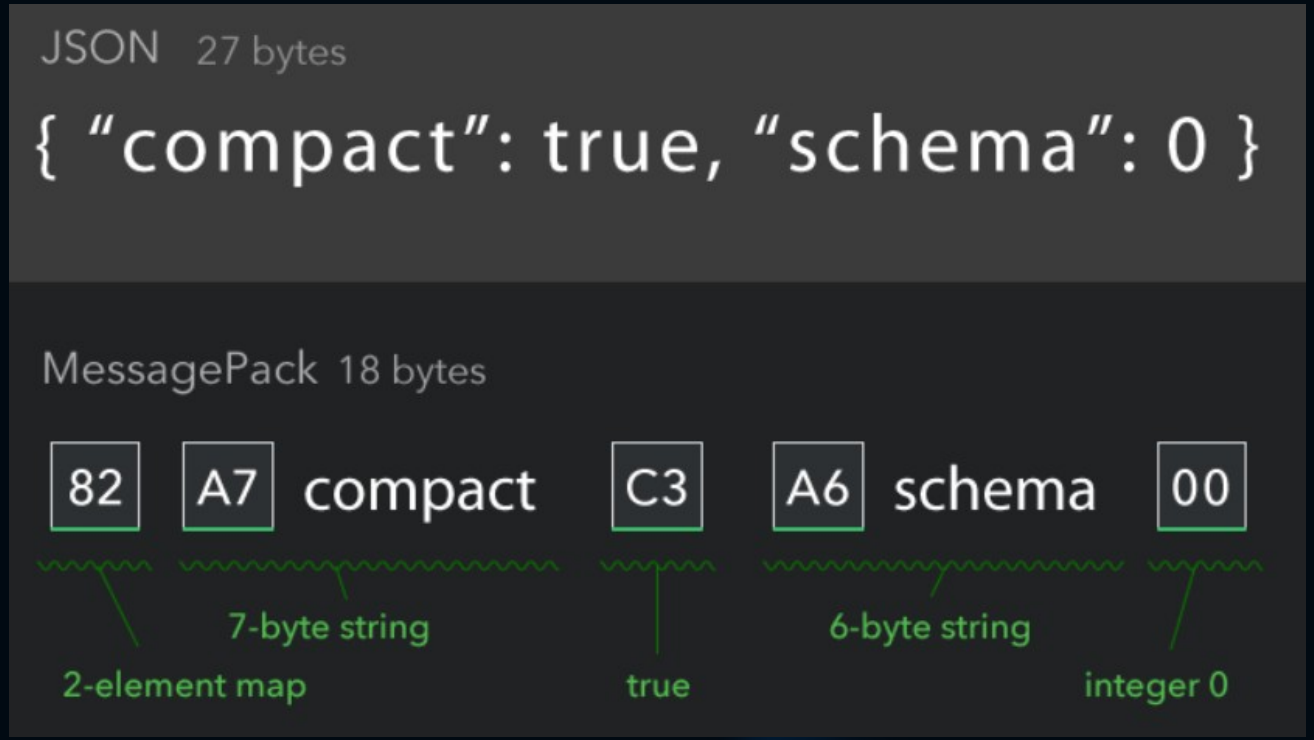
- ApplicationMaxBufferSize = 32 kb;

От сервера в буферы и т.д.

- TransportMaxBufferSize = 32 kb;

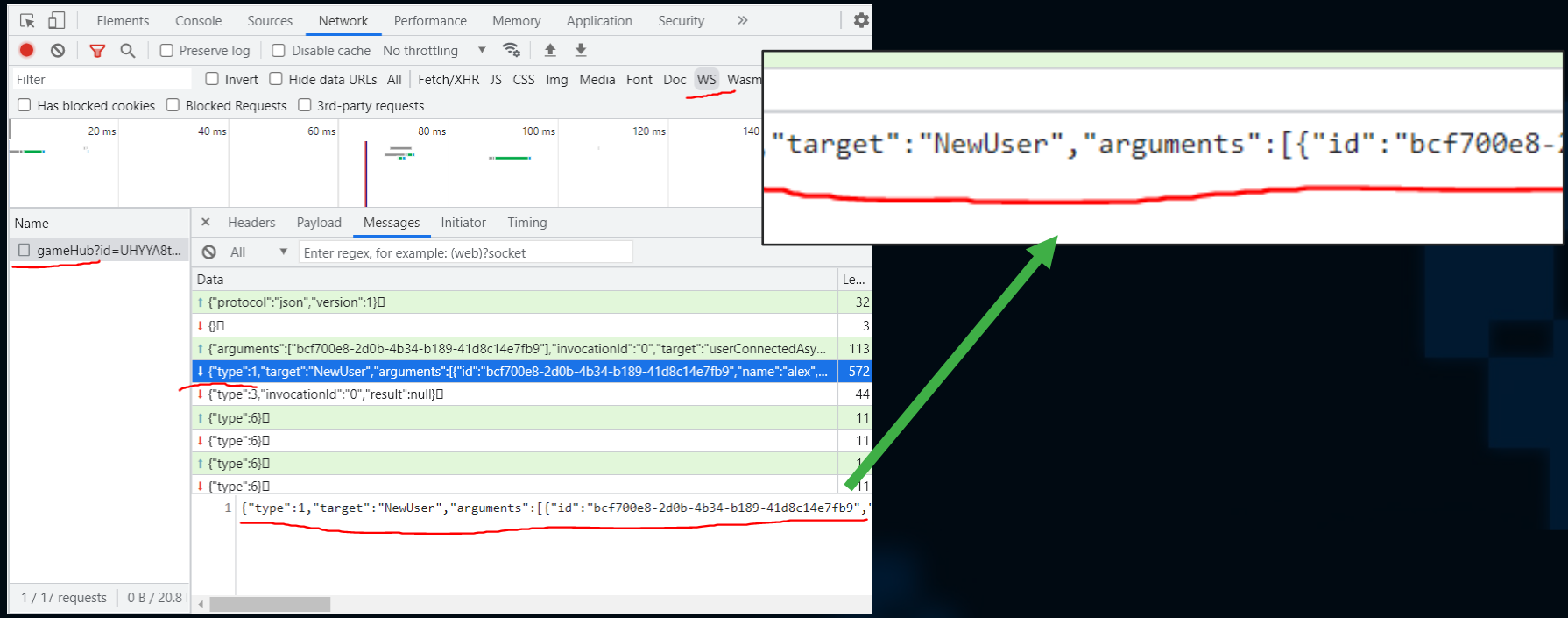


Протокол MessagePack

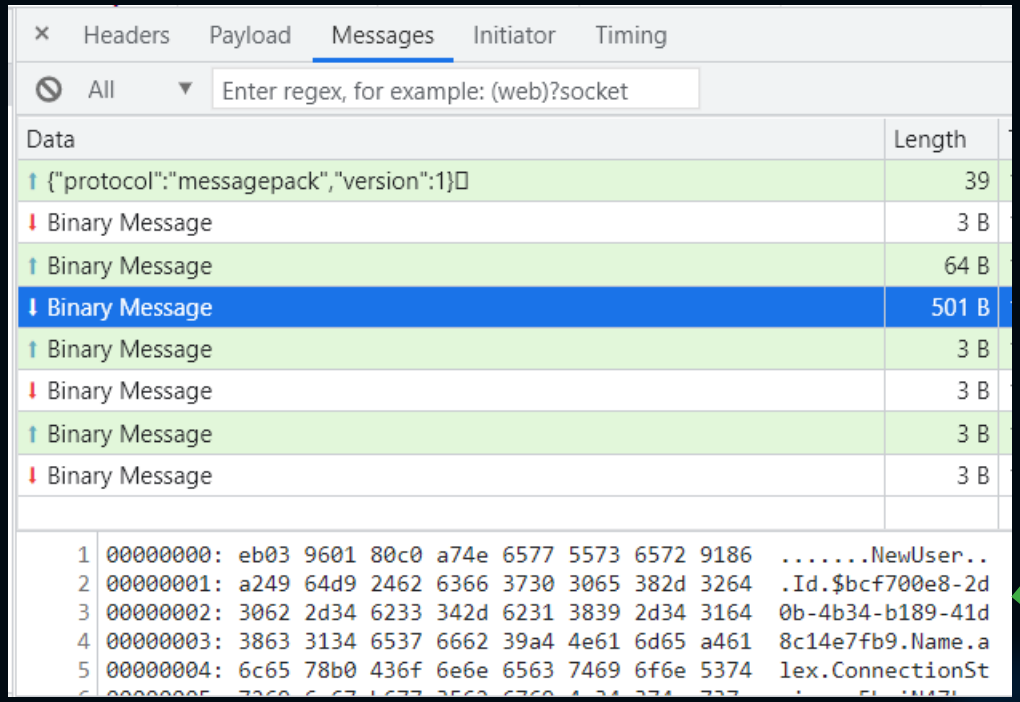


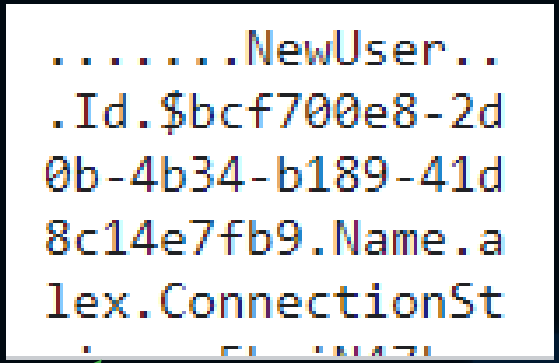
https://msgpack.org/index.html

Принцип работы



Принцип работы






```
Message Pack
        Duration: 70
        Length: 5853
JSON
        Duration: 577
        Length: 9647
```

Переключение протоколов

```
dotnet add package Microsoft.AspNetCore.SignalR.Protocols.MessagePack
```

```
libman install @microsoft/signalr-protocol-msgpack -p unpkg
```

Server

```
builder.Services.AddSignalR()
    .AddMessagePackProtocol();
```

Net client

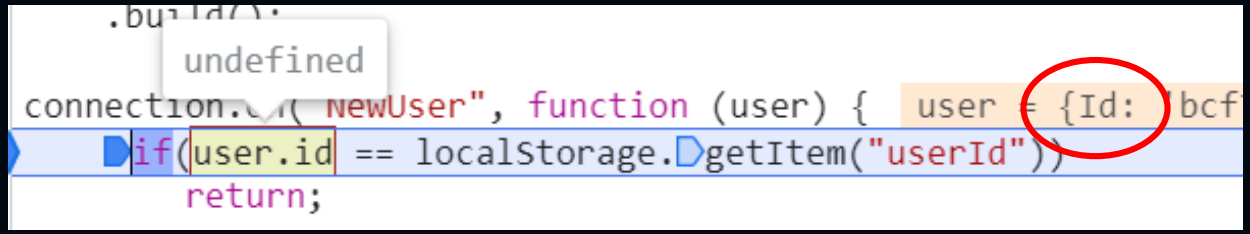
```
_connection = new HubConnectionBuilder()
        .WithUrl($"{serverPath}/gameHub")
        .WithAutomaticReconnect()
        .AddMessagePackProtocol()
        .Build();
```

JS Client

```
var connection = new signalR.HubConnectionBuilder()
    .withHubProtocol(new signalR.protocols.msgpack.MessagePackHubProtocol())
    .withUrl(serverPath + "/gameHub")
    .build();
```

И неочевидные нюансы)

Чувствительность к регистру



И неочевидные нюансы)

Чувствительность к регистру

```
"bcf700e8-2d0b-4b34-b189-41d8c14e7fb9"
connection.on( 'NewUser' , function (user) {   user = {id: 'bcf7
    if(user.id == localStorage.getItem("userId"))
        return;
```

```
[DataContract]
18 references
public class User
{
    [Key]
    [DataMember(Name = "id")]
    4 references
    public Guid Id { get; set; }
```

Потоковая передача данных

Создание потока

```
var channel = Channel.CreateUnbounded<GameObject>();
await _connection.SendAsync("MovedStream", channel.Reader);
```

Запись данных

```
foreach(var movedObject in movedObjects)
{
    await channel.Writer.WriteAsync(movedObject);
}
```

Закрытие потока

```
channel.Writer.Complete();
```

Потоковая передача данных

Получение данных

```
public async Task MovedStream(IAsyncEnumerable<GameObject> stream)
{
    await foreach (var obj in stream)
    {
        await Clients.Group(obj.LevelId.ToString())
            .GameObjectMoved(obj);
    }
}
```

Client result

Сервер (до .NET 7)

```
public async Task GameObjectClickAsync(Guid userId, Guid objectId)
{
    var gameObject = await _userService.HandleObjectClickAsync(userId, objectId);

    var user = await _userService.GetUserAsync(userId);

    if(gameObject != null)
        await Clients.Group(user.Ship.LevelId.ToString()).RemoveObject(gameObject);

    await Clients.Caller.UpdateScore(user.Score);
}
```

Client result

Клиент

```
function onGameObjectClick(id) {
    let userId = localStorage.getItem("userId");
    connection.invoke("gameObjectClickAsync", userId, id)
        .then((res) => updateScore(res));
}
```

Сервер

```
public async Task<int> GameObjectClickAsync(Guid userId, Guid objectId)
{
    var user = await _userService.GetUserAsync(userId);
    // user take asteroid logic
    return user.Score;
}
```

Возможности SignalR

Компонент	Server	.NET	JS	Java
Azure SignalR Service Support	2.1.0	1.0.0	1.0.0	1.0.0
Server-to-client Streaming	2.1.0	1.0.0	1.0.0	1.0.0
Client-to-server Streaming	3.0.0	3.0.0	3.0.0	3.0.0
Automatic Reconnection	3.0.0	3.0.0	3.0.0	✕
WebSockets Transport	2.1.0	1.0.0	1.0.0	1.0.0
Server-Sent Events Transport	2.1.0	1.0.0	1.0.0	✕
Long Polling Transport	2.1.0	1.0.0	1.0.0	3.0.0
JSON Hub Protocol	2.1.0	1.0.0	1.0.0	1.0.0
MessagePack Hub Protocol	2.1.0	1.0.0	1.0.0	5.0.0
Client Results	7.0.0	7.0.0	7.0.0	✕

Исходные коды

https://gitlab.com/akuzn1/dotnext2022-signalr

Спасибо за внимание

Александр Кузнецов

СКБ Контур

EMAIL akuzn1@gmail.com

DOTNEXT