Cognizant
Softvision

SIGNALR WITH .NET CORE

COGNIZANT SOFTVISION ARGENTINA .NET COMMUNITY

Matías Baldi
Community Manager
matias.baldi@softvision.com

Guillermo Murano
Tech Lead
guillermo.murano@softvision.com

Cristian Piqué
Tech Lead
cristian.pique@softvision.com

Romualdo Lanzon
Tech Lead
romualdo.lanzon@softvision.com

Nicolás Granata
Software Engineer
nicolas.granata@softvision.com

Javier Paez
Tech Lead
javier.paez@softvision.com

Cognizant Softvision

AGENDA

- What is SignalR
- When to use SignalR
- Communication types
- Hubs
- WebSockets in .NET Core
- Examples

Cognizant
Softvision

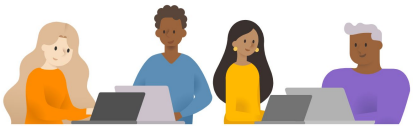
What is SignalR

WHAT IS SIGNALR

SignalR in ASP.NET Core is an open-source library that simplifies adding real-time web functionality to apps, wrapping the complexity of several web transports.

Fast and scalable

Open source and open protocol, available on GitHub



Connect from different clients, not only web

Cognizant
Softvision

When to use SignalR

WHEN USE SIGNALR

SIgnalR is a good candidate for real time applications, when we need to get the most up-to-date real-time information on a web page, without constantly refresh that page manually.

- Keep user engage
- Email clients
- Social networking/media
- Web documents / collaboration
- Auctions, gambling
- Gaming
- Stock quotes/trading
- Sensors
- Monitoring, tracking, job progress updates
- Sport events
- Real-time forms

Cognizant
Softvision

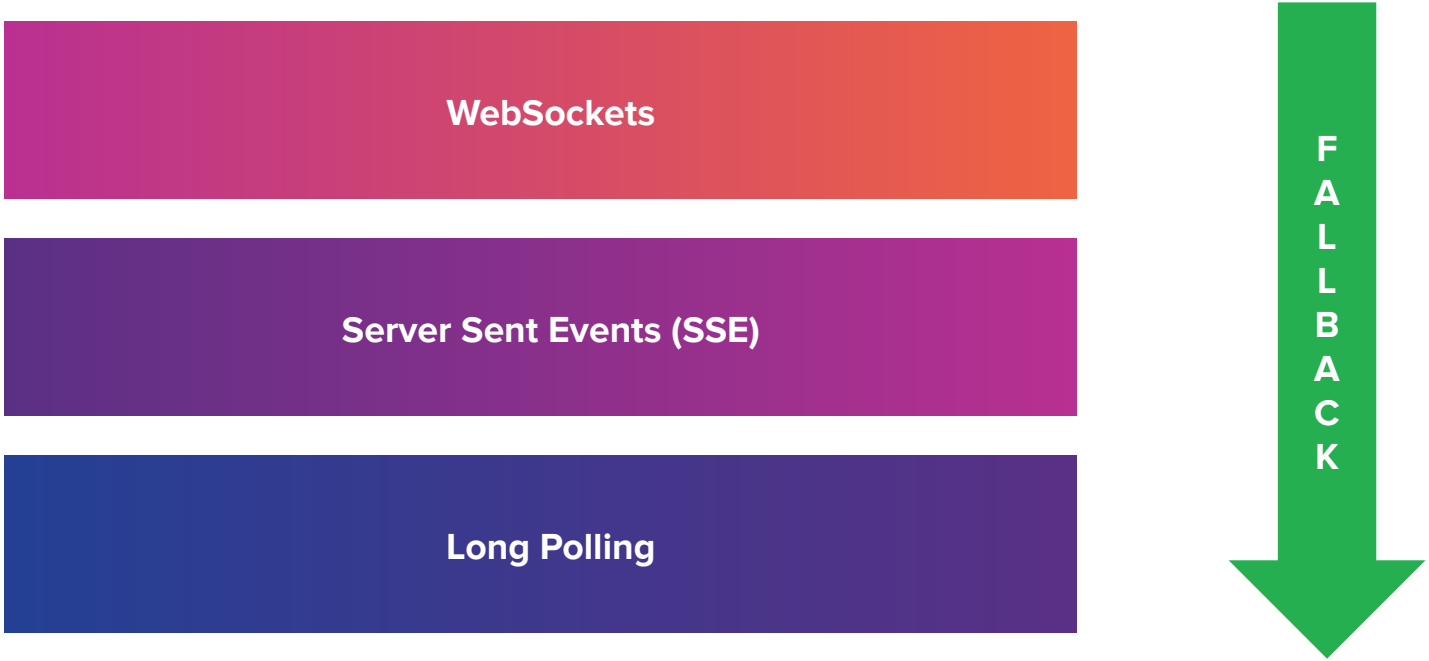
COMMUNICATION TYPES
SignalR transports

PULL & PUSH MODELS

In the traditional model, a client initiates a connection to request the desired information hosted by a server using the HTTP protocol. The server processes this request, returns the resource and the connection is instantly closed.

Sometimes, we need the server to take the initiative and be capable of sending information to the client exactly when a relevant event occurs, instead of waiting for the client to request it.

SIGNALR TRANSPORTS



Cognizant Softvision

FALLBACK MECHANISM

01 **Web Sockets**

On the client, a modern browser is required. No support for versions prior to IE 10. On the server side, when using IIS or HttpSysServer, Windows 8 / Windows Server 2012 or newer is required on the client.

02 **Server Send Events**

It is not supported on the Internet Explorer browsers.

03 **Long polling**

This protocol works with all versions of all browsers and is considered a fallback solution.

Cognizant
Softvision

HUBS

HUBS AND CONNECTIONS

Hubs are the pipeline classes used to communicate between clients and server. In client side we use connections to achieve the communication with the server

From Client Side (Connections)

- Defines methods called from the server
- Relies on the client side code (Javascript, Typescript, C#, etc) to define a contract
- Can catch errors from the server

From server side (Hubs)

- Defines methods called from the client
- It is mapped to a specific Url in the Startup
- Could define a strongly typed client hub or rely on magic strings

SERVER HUB: CLIENTS & CONTEXT

Context is a property in the hub that gives you user information

- ConnectionId
- User
- Items

Clients lets us called methods on specific clients or groups

Groups property let us add or remove clients from groups

STREAMING

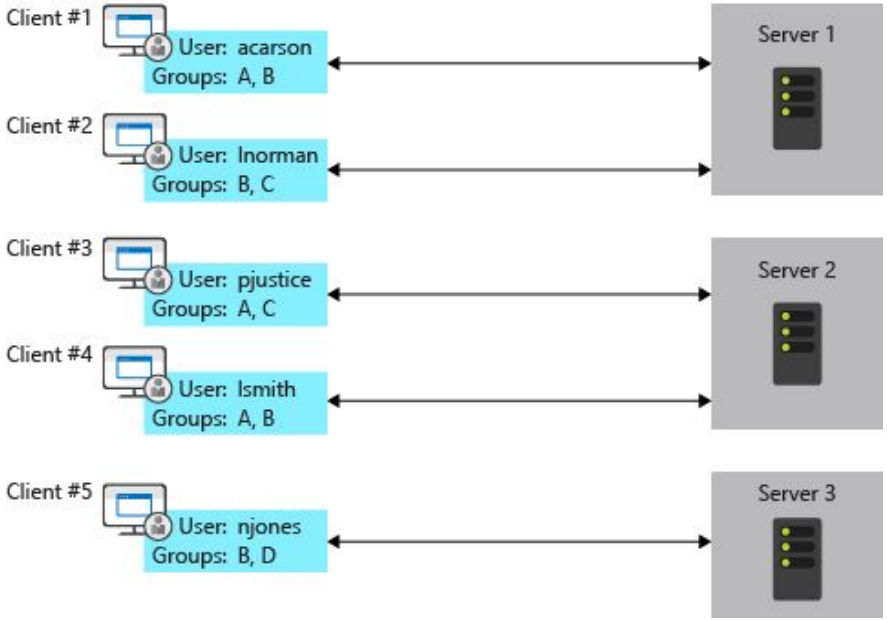
Streaming transmission is a technology that transfers data in the form of stable continuous flow

When the return value of a Hub method is ChannelReader or Task <ChannelReader> SignalR converts the Hub method into a streaming method automatically.

- Gaming
- Video
- Music
- Live events

SCALING

In a server farm, SignalR on each server is unaware of the connections on the other servers



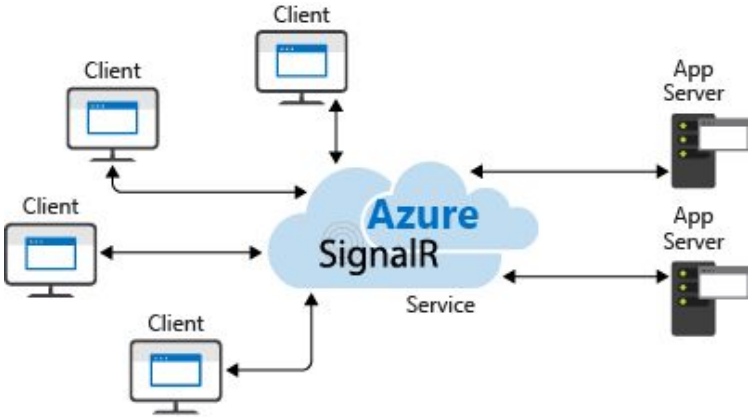
Cognizant Softvision

SCALING

The options for solvinhg this problem are:

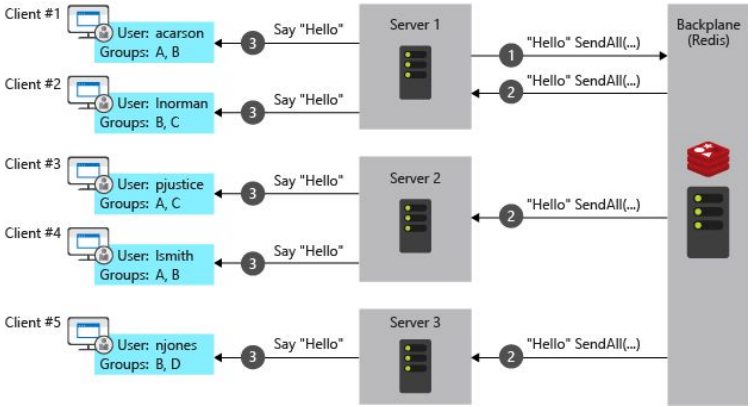
1

Azure SignalR Service



2

Redis backplane



Cognizant
Softvision

Cognizant
Softvision

WebSockets In netcore

DIFFERENCES VS SIGNALR

Advantages of WebSockets

- Full duplex communication model
- Efficient and easy to use
- Faster than HTTP. It uses TCP

Disadvantages of WebSockets

- Not universally supported across all browser and server stacks.
- Takes over the communication protocol between client and server for a specific connection.

SignalR provides all of the functionality you would need to write yourself for a Websocket implementation.

You don't need to worry about updates to WebSocket, since SignalR will continue to be updated to support changes in the underlying transport.

Using SignalR enable support for old browsers

HOW TO USE IT

MIDDLEWARE + SERVICE CONFIG

```
app.UseWebSockets();
```

```
app.Use(async (context, next) =>
{
    if (context.Request.Path == "/ws")
    {
        if (context.WebSockets.IsWebSocketRequest)
        {
            WebSocket webSocket = await context.WebSockets.AcceptWebSocketAsync();
            await Echo(context, webSocket);
        }
        else
        {
            context.Response.StatusCode = 400;
        }
    }
    else
    {
        await next();
    }

});
```

USING THE SOCKET

```
private async Task Echo(HttpContext context, WebSocket webSocket)
{
    var buffer = new byte[1024 * 4];
    WebSocketReceiveResult result = await webSocket.ReceiveAsync(new ArraySegment<byte>(buffer),
        CancellationToken.None);
    while (!result.CloseStatus.HasValue)
    {
        await webSocket.SendAsync(new ArraySegment<byte>(buffer, 0, result.Count),
            result.MessageType,
            result.EndOfMessage,
            CancellationToken.None);

        result = await webSocket.ReceiveAsync(new ArraySegment<byte>(buffer),
            CancellationToken.None);
    }

    await webSocket.CloseAsync(result.CloseStatus.Value,
        result.CloseStatusDescription,
        CancellationToken.None);
}
```

Cognizant
Softvision

SHOW ME THE CODE!

Cognizant
Softvision

Q&A

THANK YOU!

Repo:

https://github.com/RmldLnzn/meetupSignalRCore2