ASP.Net SignalR

Christen Zarif

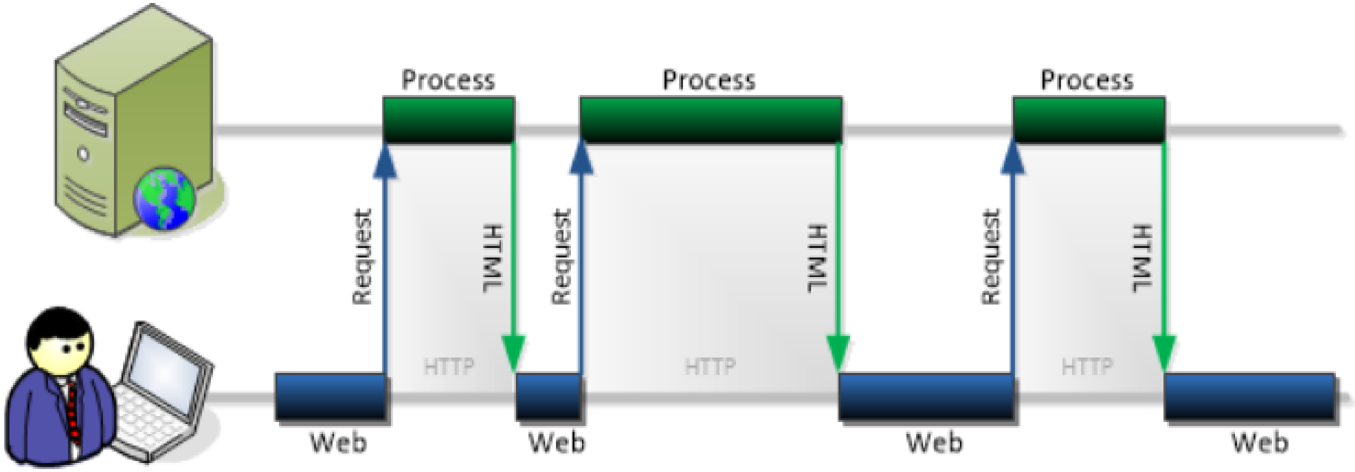
HTTP OPERATION

- HTTP (HyperText Transfer Protocol) is the language in which the client and the server of a web application speak to each other
- Its operation is based on a request-response schema which **is always started by the client** (**Pull model**)

COMMUNICATION STYLE

Pull model

- It is a **synchronous** process:



HTTP communication between a browser and a web server

Pull model (Con.)

- **asynchrony** of modern

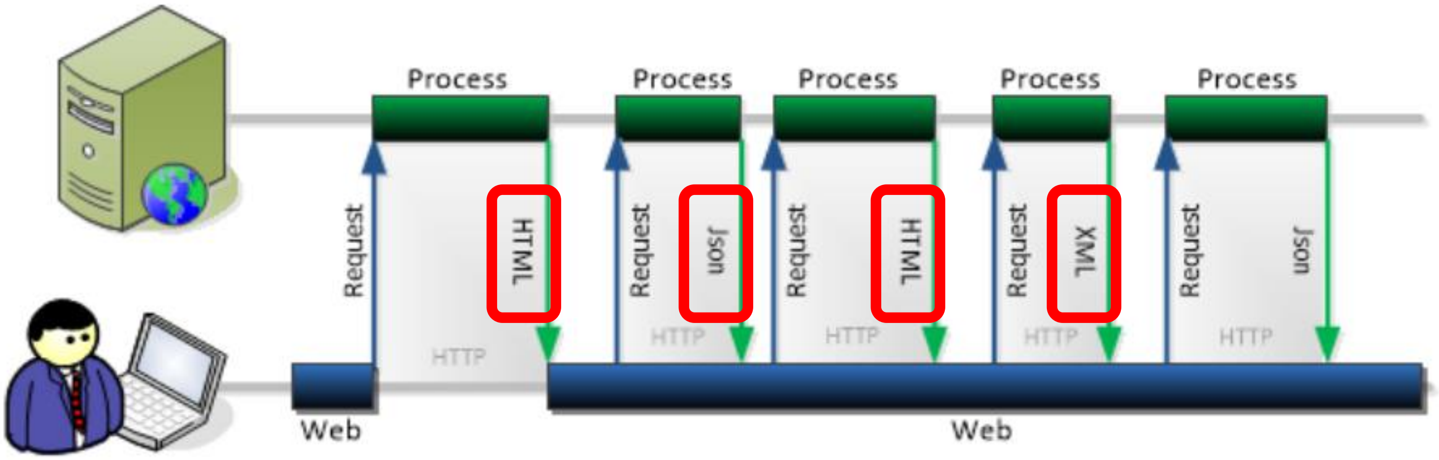


Figure 2. Ajax in a webpage

HTTP Limitation

- **With this protocol it is not easy to**
  - implement instant-messaging applications
  - Chat-rooms,
  - Collaboration tools,
  - multiuser online games
  - real-time information services, even when using asynchrony.
- The reason is simple: **HTTP is not oriented to real time**.

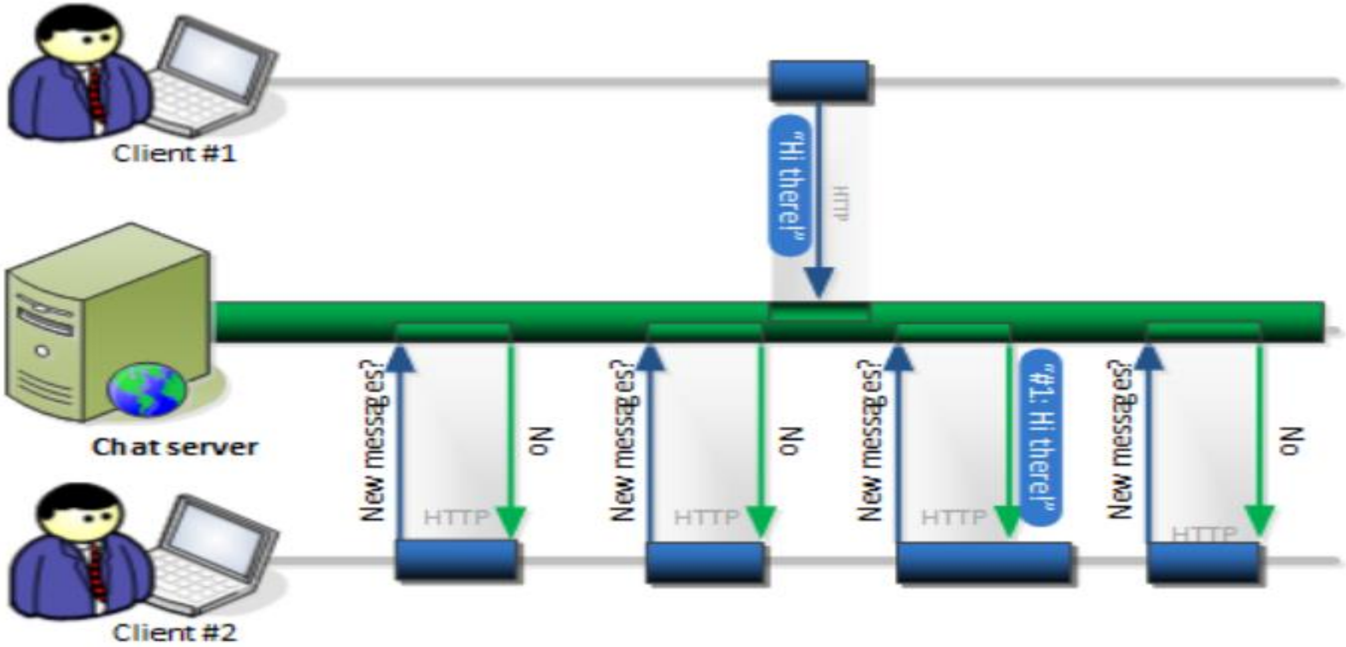
The client is always the one to take the initiative , deciding when to connect to the server

Real Time Application

- Real-time functionality is the ability to have server code **push** content to **connected clients** instantly as it becomes available, rather than having the server wait for a client to request new data.
- Real Time is all about **Pushing** instead of **Pulling**
- Push Technology is completely different from Pull Technology. Its about **getting** told what's new, instead of **asking** for what's new!!!
- Facebook, Twitter, Yahoo Cricket Live, Stock Ticker

POLLING

- Polling basically consists in making **periodic connections** from the client to check whether there is any relevant **update** at the server (30 seconds).



setInterval

Figure 3. Polling in a chat room service

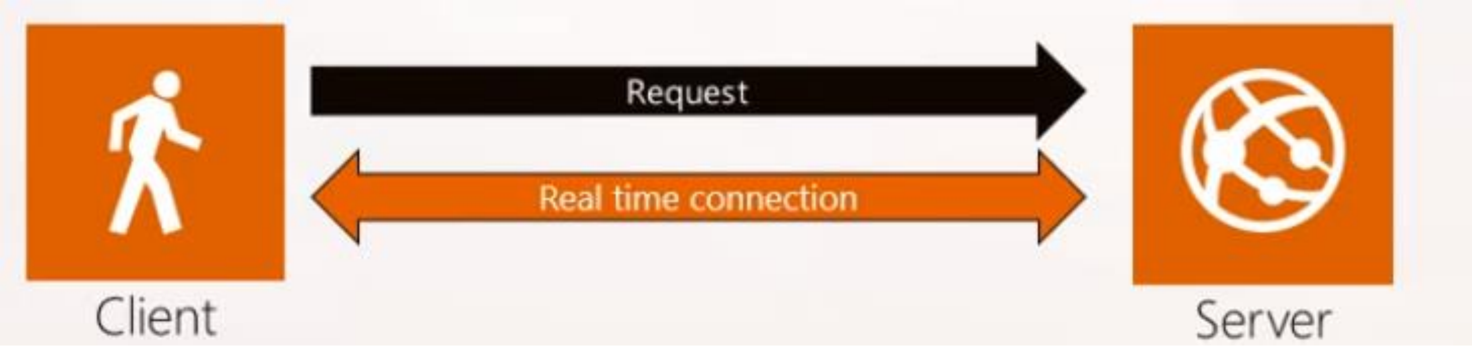
Polling Advantage & Disadvantage

- Easy implementation
- Its universal application: with all browsers and with all servers
- We still use the Pull model
- Cost : Server Load

PUSH METHODS

Push Method - Transport Options

- Modern HTML5 approaches
  - Server Send Event (SSE)
  - WebSocket
- Comet approaches
  - Long Polling
  - Forever Frame



Push Today

Push Today

- They are the most universal ones (they work in all types of client and server systems),
- They are used natively by SignalR,
  - Long Polling
  - Forever Frame

Long Polling

- This Push technique is quite similar to Polling
- The **connection** remains open until the server has something to notify

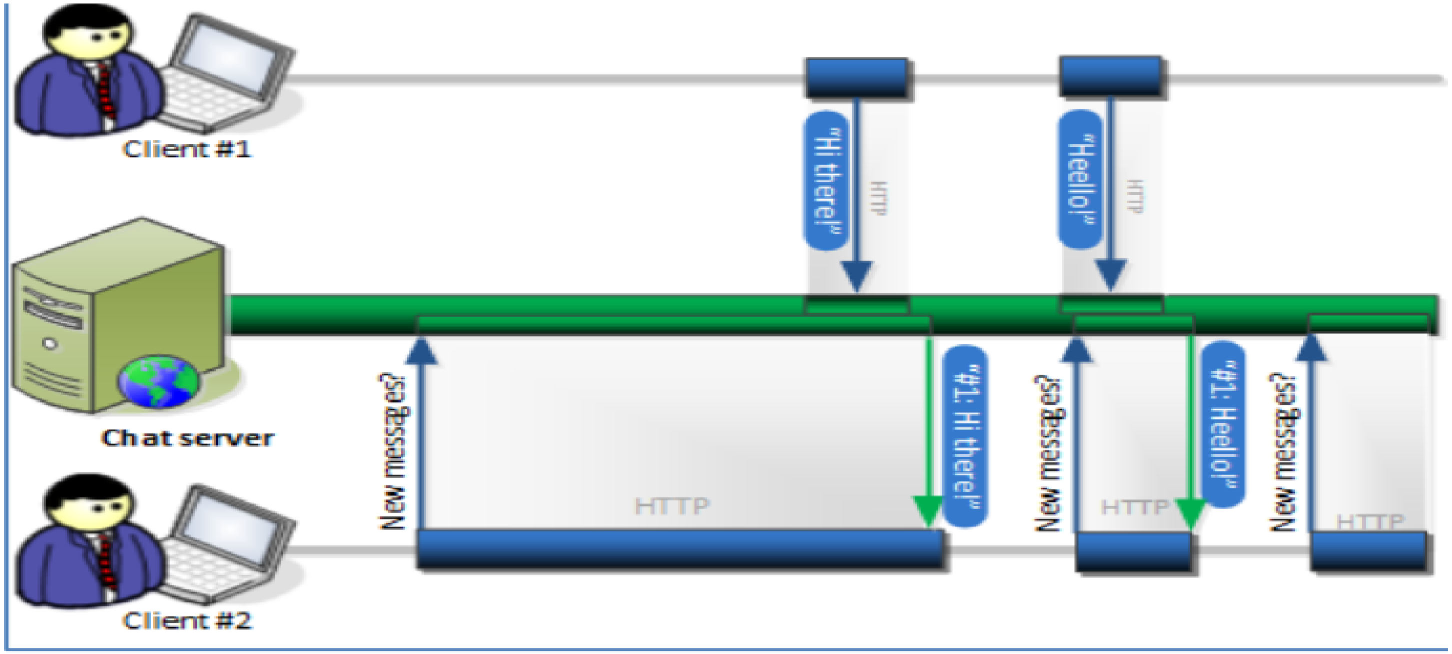


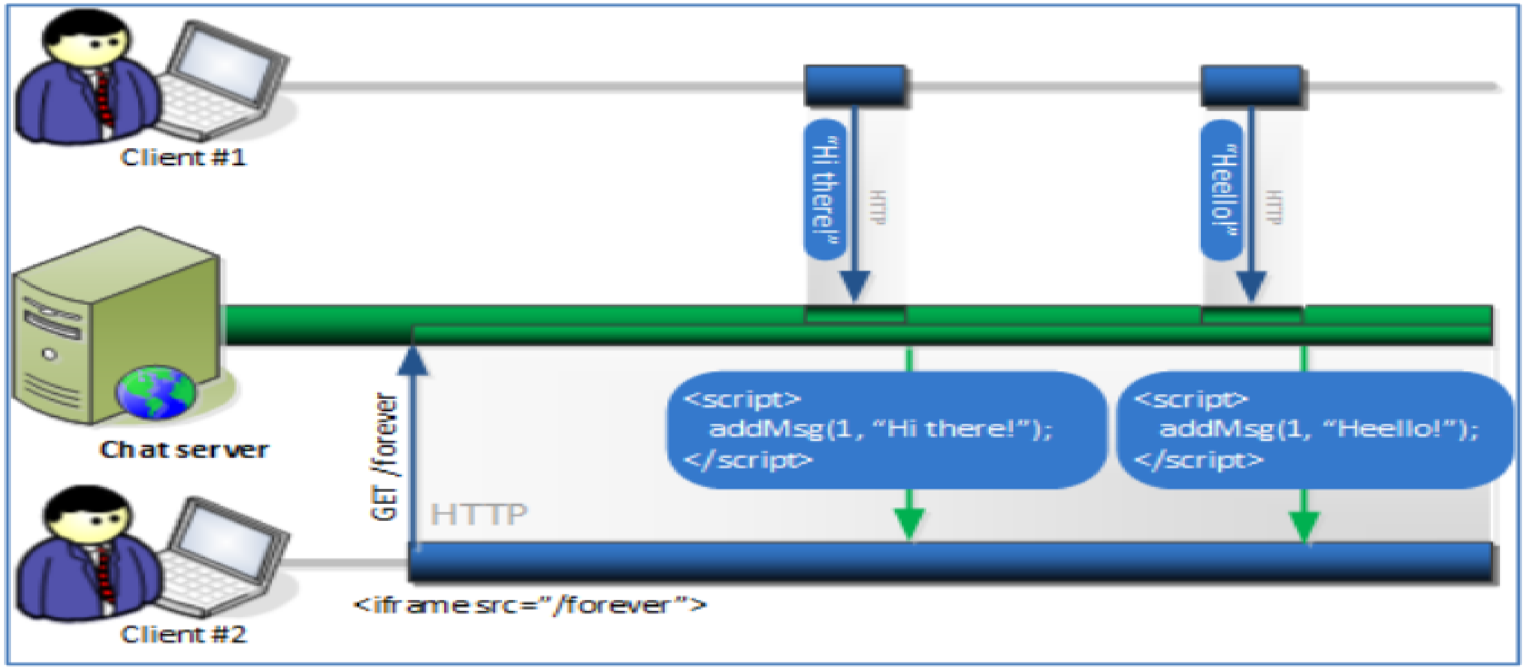
Figure 6. Long polling

Long Polling Feature

- Low delay in updating the client
- Number of connection openings and closures is reduced
- Completely universal (***All Browsers support***)
- But there are still many connection openings and closures if the rate of updates is high

Forever Frame

- also known as ***hidden iframe*** which is an approach specific to Internet Explorer.



```
<script>
  addMsg(1, "Hi there!");
</script>
```

```
<script>
  addMsg(1, "Heello!");
</script>
```

Figure 7. Forever frame

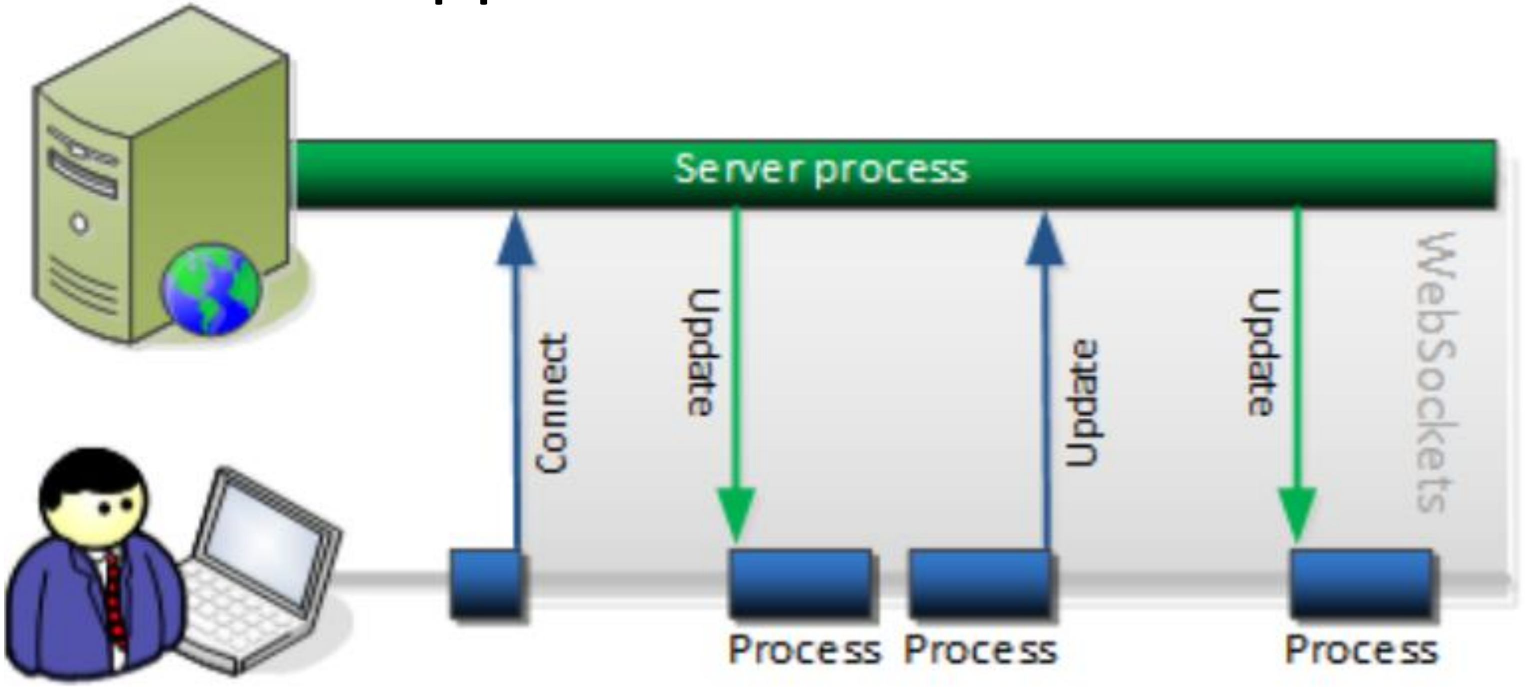
Forever Frame (Con.)

- less common than long polling
- Active components embedded in pages are being eliminated from the Web at a dramatic speed and are being substituted for more modern

Modern HTML5 approaches

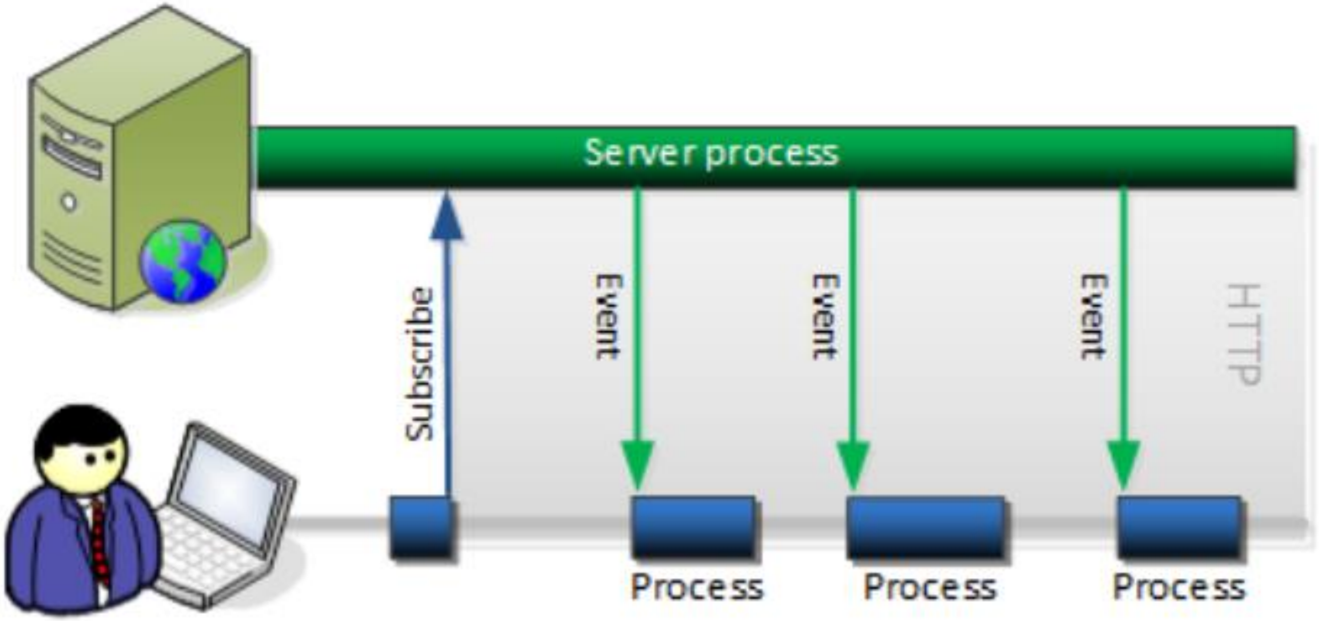
Websockets

- Provides ***two-way communication channels*** over a TCP connection between client and server,
- only recent versions of most common web browsers support it



Server-Sent Event

- *one-directional channel* from the server to the client, but opened by the client.
- Currently, almost all browsers support this standard **except** for IE



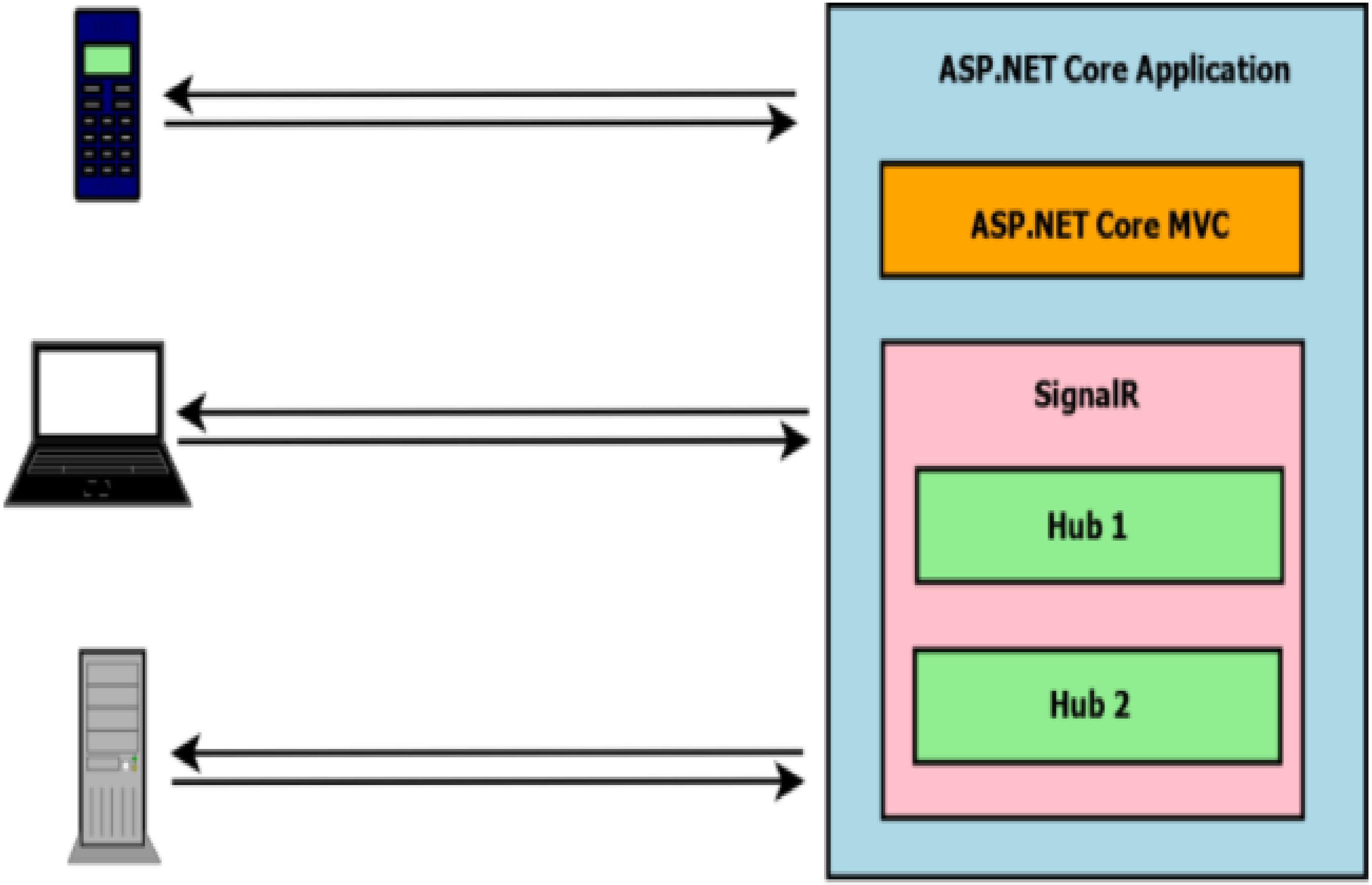
```
var source = new EventSource('/getevents');
source.onmessage = function(event) {
alert(event.data);
};
//----------------------------------------------------------------------------------
---
var ws = new WebSocket("ws://localhost:9998/echo");
ws.onopen = function() {
	// Web Socket is connected, send data using send()
	ws.send("Message to send");
	alert("Message is sent...");
};
ws.onmessage = function(evt) {
	var received_msg = evt.data;
	alert("Message is received...");
};
ws.onclose = function () {
	// websocket is closed.
	alert("Connection is closed...");
};
```

SIGNALR

Asp.Net Core SignalR

- ASP.NET Core SignalR is an open-source library that can be used to develop real-time web applications in ASP.NET Core. SignalR contains an API that allows server-side code to send messages to connected client browsers.

Introduction to SignalR

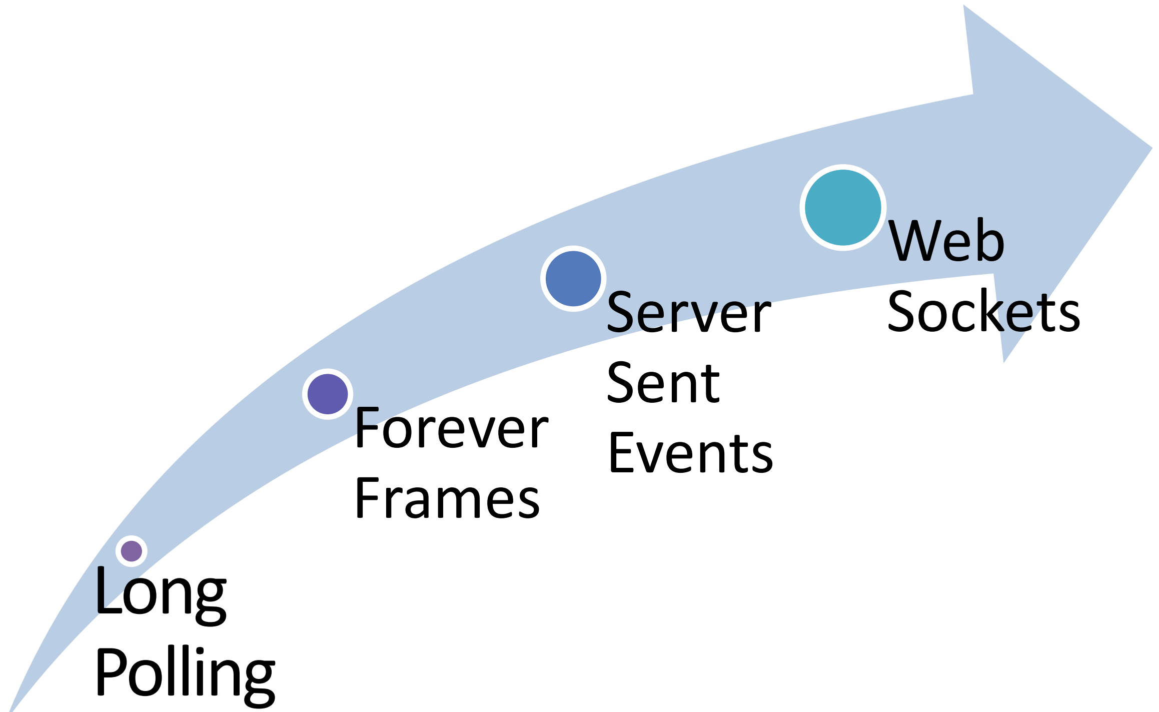


Introduction to SignalR

SignalR in ASP.NET Core uses Remote Procedure Call to enable the server to call a function on the client using underlying transport.

SignalR in ASP.NET core allows running on many platforms, other than windows, which are supported by ASP.NET Core.

SignalR Fallback



Long Polling

Forever Frames

Server Sent Events

Web Sockets

SignalR automatically chooses the best transport method that is within the capabilities of the server and client.

Some SignalR Features

- Handles connection management automatically.
- Sends messages to all connected clients simultaneously. For example, a chat room.
- Sends messages to specific clients or groups of clients.
- Scales to handle increasing traffic.

As developers, we will focus on *programming our services*, and SignalR will be in charge of *managing the connections* and the specifics of client and server software throughout

Transport Negotiation

- After start() method initiates communication with the server, thus beginning the negotiation phase in which the technologies or techniques to be used to maintain the persistent connection will be selected.
  - Websocket
  - Server-Sent Events
  - Forever Frame
  - Long Polling

THANK YOU