



SIG WebSockets

**Nathan Breuring
Mark Mudde**

24 September 2019

Overview

17:30 – 18:00

Short introduction: **WebSockets 101**

Some background, history and the inner workings of WebSockets

18:00 – 18:45

Dinner Break

A fine selection of delicious pizzas!

18:45 – 19:00

Short introduction: **Socket.io**

A JS library that makes it easy to use WebSockets

19:00 – 21:00

Hands-On

Creating your own chat app using Socket.io -- using either React or Angular

WebSockets 101

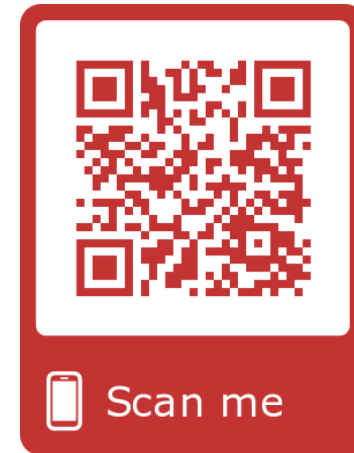


A short introduction

WebSockets 101

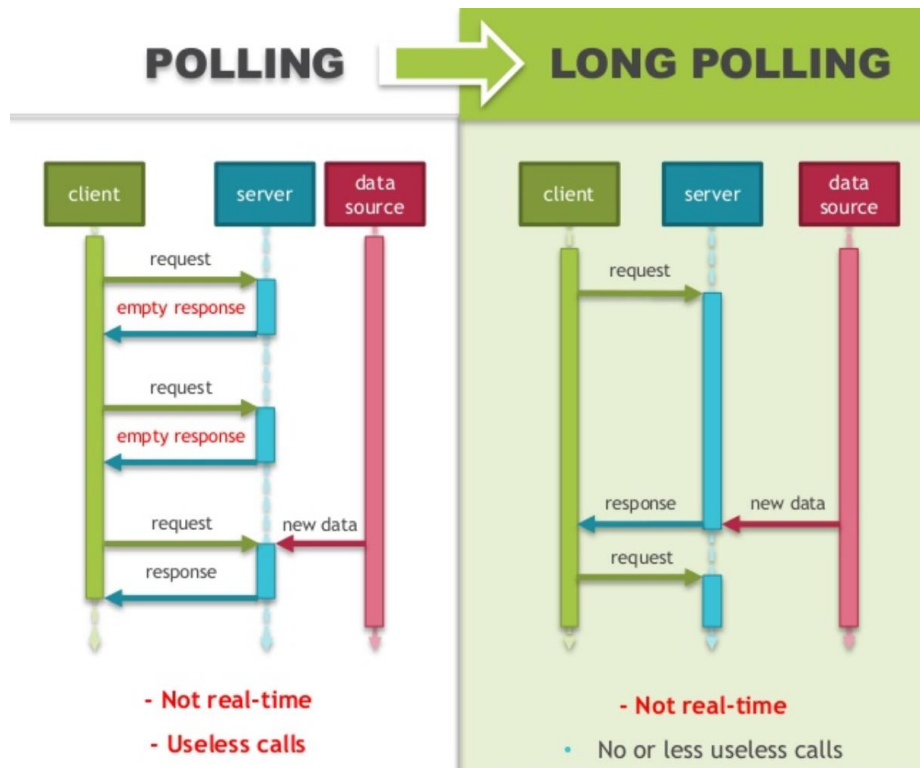
Why WebSockets?

- Session is 'INITIATED'
- HTTP: request and response
- Polling – every three seconds
- Session is 'CONNECTED'



WebSockets 101

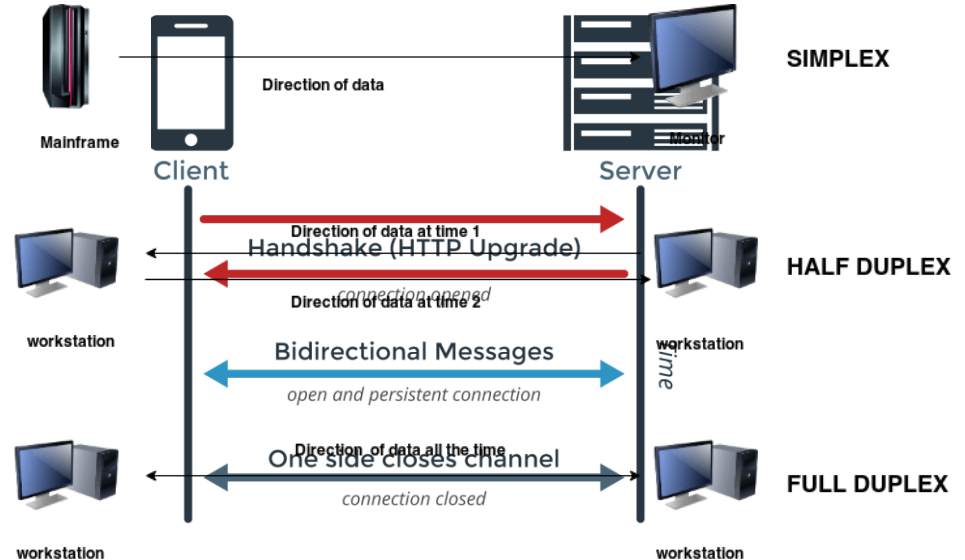
Polling vs Long Polling



WebSockets 101

Enter: WebSockets

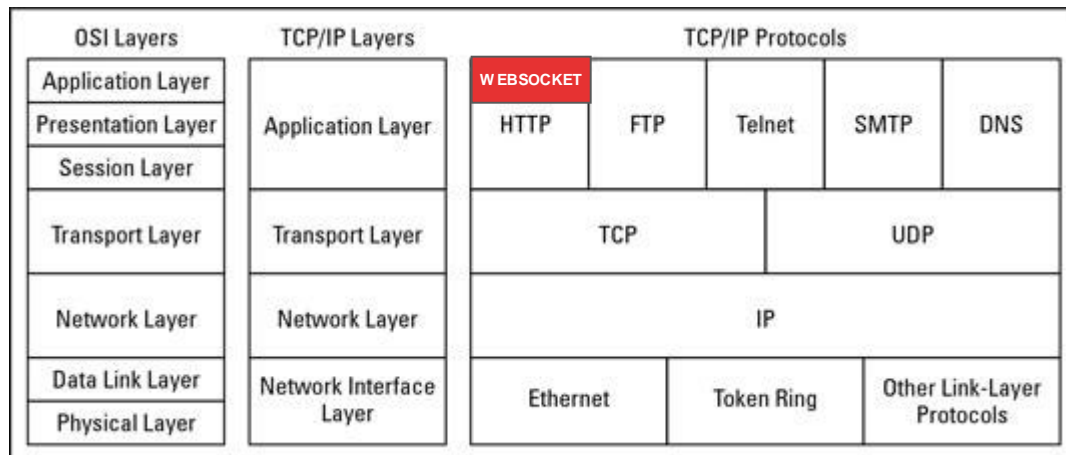
- Standardized in 2011 by IETF
(RFC 6455 - <https://tools.ietf.org/html/rfc6455>)
- WebSockets: *full-duplex*
- Realtime updates
- Using existing TCP connection
- Lifecycle: Handshake – Messages sent back and forth – Closing connection



WebSockets 101

Short technical background

- Protocol
- Internet Protocol
- Internet Protocol Suite (TCP/IP)
- Transmission Control (TCP)
- User Datagram (UDP)
- HyperText Transfer (HTTP)
- WebSocket



WebSockets 101

(not so secret) Handshake

- Client HTTP upgrade request
- Successful: 101 Switching Protocols
- Unsuccessful: 400 Bad Request
- Switches to ws:// or wss://
- Connected and able to send and receive
- Will remain open until explicitly closed

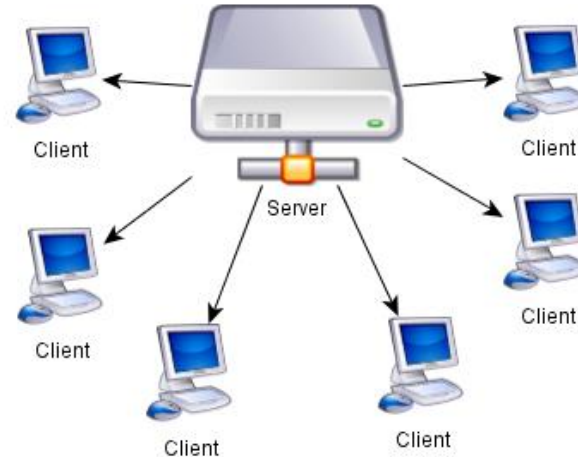
```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: x3JJHMbDL1EzLkh9GBhXDw==
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
Origin: http://example.com
```

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept: HSmrc0sMlYUkAGmm5OPpG2HaGWk=
Sec-WebSocket-Protocol: chat
```


WebSockets 101

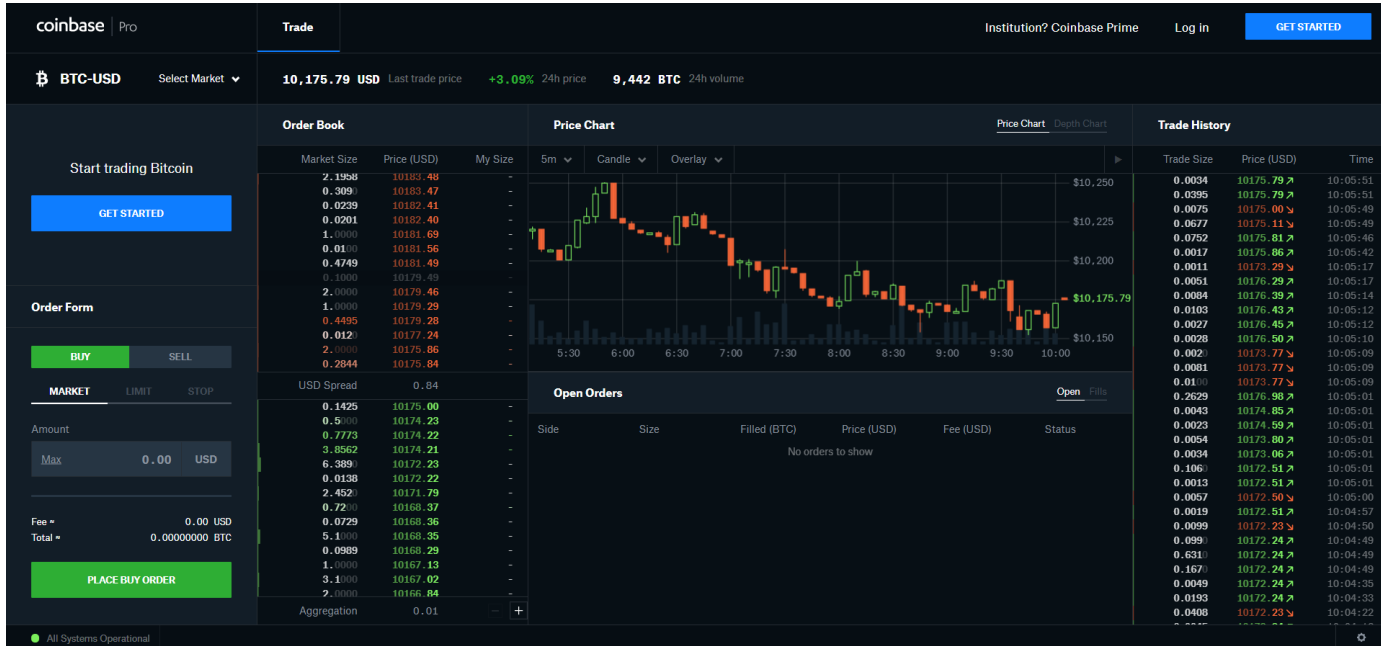
Examples

- Multiplayer browser games
- Collaborative code editing
- Live text for sports/news/finance websites
- Social media feed
- Real-time to-do apps with multiple users



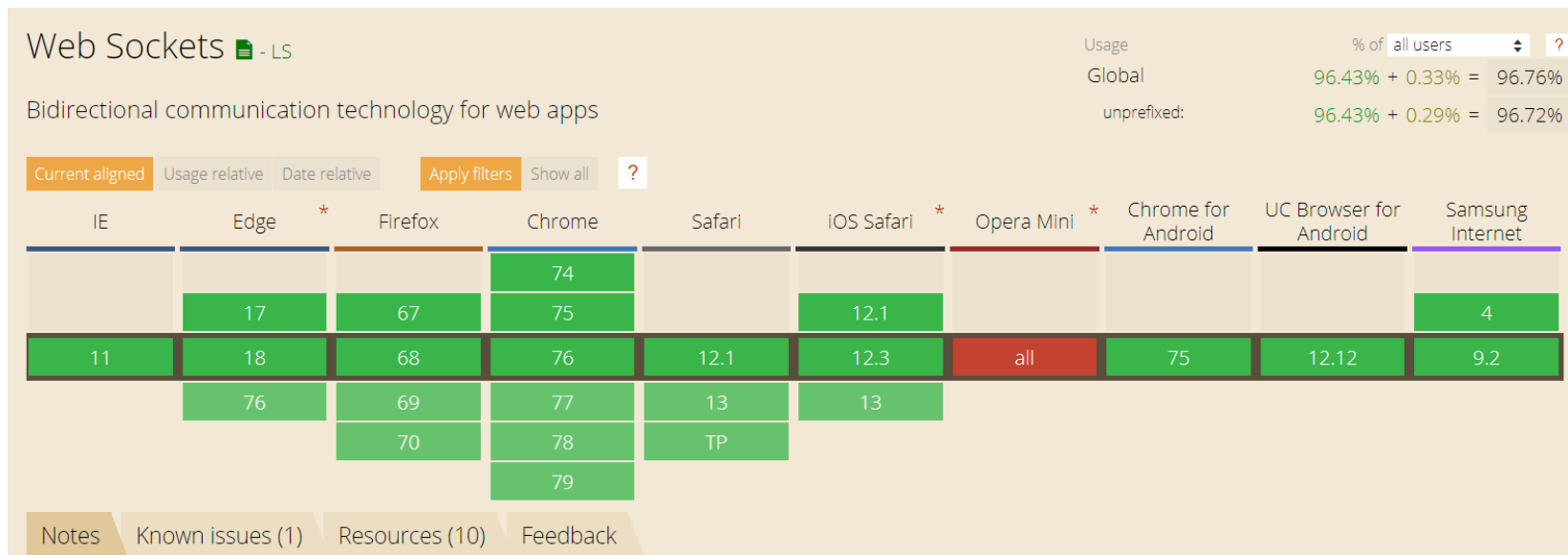
WebSockets 101

WebSockets in action



WebSockets 101

Browser Support



WebSockets 101

When not to use WebSockets

- If data only has to be collected once
- If realtime updates are not the main concern

THANK YOU FOR YOUR ATTENTION

Any Questions?

We will start with the **hands-on** here at

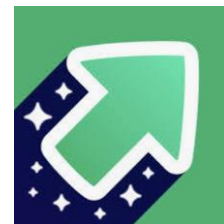
18:45

Enjoy your meal!!















101

- A library that enables real-time, bidirectional and event-based communication between browser and server
- Consists of 2 packages
 - NodeJS server <https://socket.io/docs/server-api/>
 - JS client <https://socket.io/docs/client-api/>
- Features:
 - Reliable connections
 - Auto-reconnection support
 - Disconnection detection
 - Room support



Under the hood

- Implementation of Engine.IO
 1. Establish long-polling connection
 2. Tries to upgrade to transports like WebSocket
- WebSockets have trouble connecting in the presence of
 - Proxies
 - Load balancers
 - Personal firewalls
 - Antivirus software

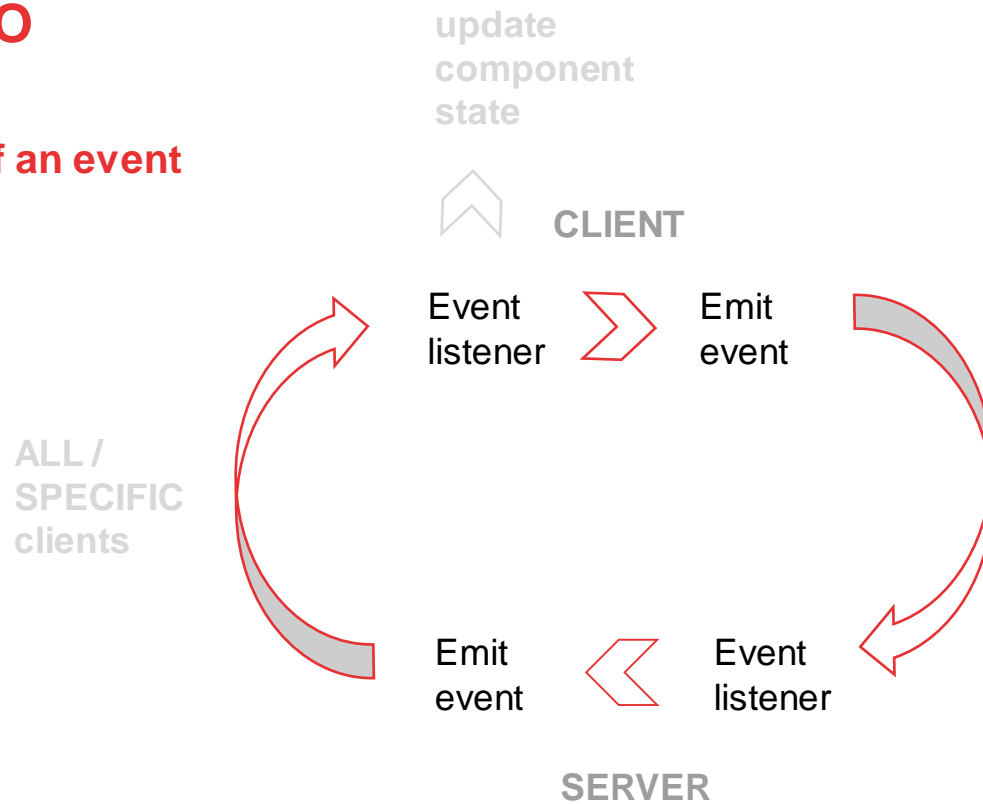
Name	× Headers Messages Timing
 localhost	<div>▼ General</div> <div>Request URL: ws://localhost:3100/socket.io/?EIO=3&transport=websocket&sid=DvqT17bkG9C5WLbAAAC</div> <div>Request Method: GET</div> <div>Status Code:  101 Switching Protocols</div> <div>► Response Headers (4)</div> <div>▼ Request Headers view source</div> <div>Accept-Encoding: gzip, deflate, br</div> <div>Accept-Language: en-GB,en-US;q=0.9,en;q=0.8</div> <div>Cache-Control: no-cache</div>
 bundle.js	
 0.chunk.js	
 main.chunk.js	
 ?EIO=3&transport=polling&t=MrTrh0m	
 ?EIO=3&transport=polling&t=MrTrh1L&sid=DvqT17bkG...	
 ?EIO=3&transport=websocket&sid=DvqT17bkG9C5WL...	
 manifest.json	
 favicon.ico	
 info?t=1569237413980	
 websocket	

Socket.IO

DEMO

Socket.IO

Lifecycle of an event



Docs

- <https://socket.io/docs/>
- <https://socket.io/docs/emit-cheatsheet/>
- Event emitters `.emit()`
- Event listeners `.on()`
- Use listeners to
 - callback emits (mostly done server side):
 - callback (component) state updates

```
`client.on("message", function(payload) {  
  io.emit("message", payload);  
});`
```

Overview

- Exercises and skeleton project
 - <https://github.com/AMIS-Services/sig-socket-io>
- Examples
 - React
 - <https://github.com/nathanbreuring/socket-chat>
 - Angular
 - <https://github.com/MarkMudde/sig-socket-angular>

(socket)Handler

- ...