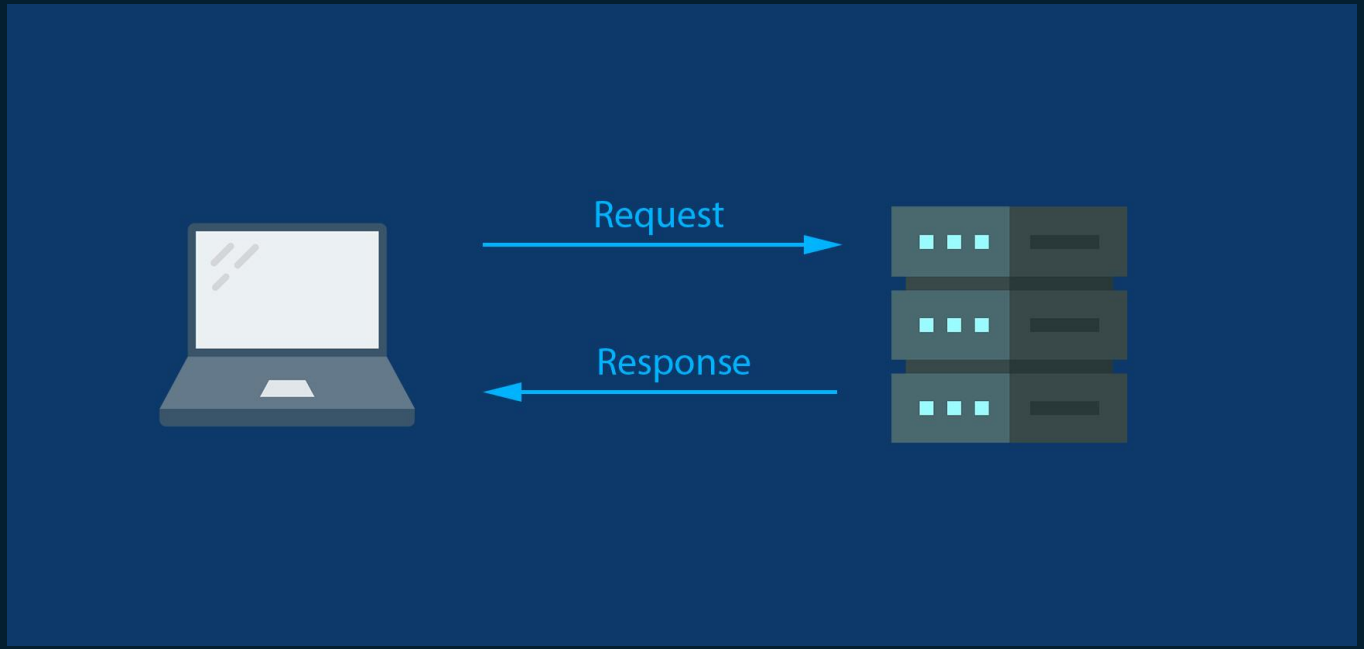
Web sockets

Life before websockets.

1. Request - response cycle
2. Every request has one response
3. Server can only communicate with client while client has made a request



Polling

Polling is the technique when client asking server for data regularly / with some interval

Long polling

Long polling is essentially a more efficient form of the original polling technique. Making repeated requests to a server wastes resources, as each new incoming connection must be established, the HTTP headers must be parsed, a query for new data must be performed, and a response (usually with no new data to offer) must be generated and delivered. The connection must then be closed and any resources cleaned up.

Official Long polling documentation

https://tools.ietf.org/html/rfc6202#section-2.2

Server-Sent Events (server push)

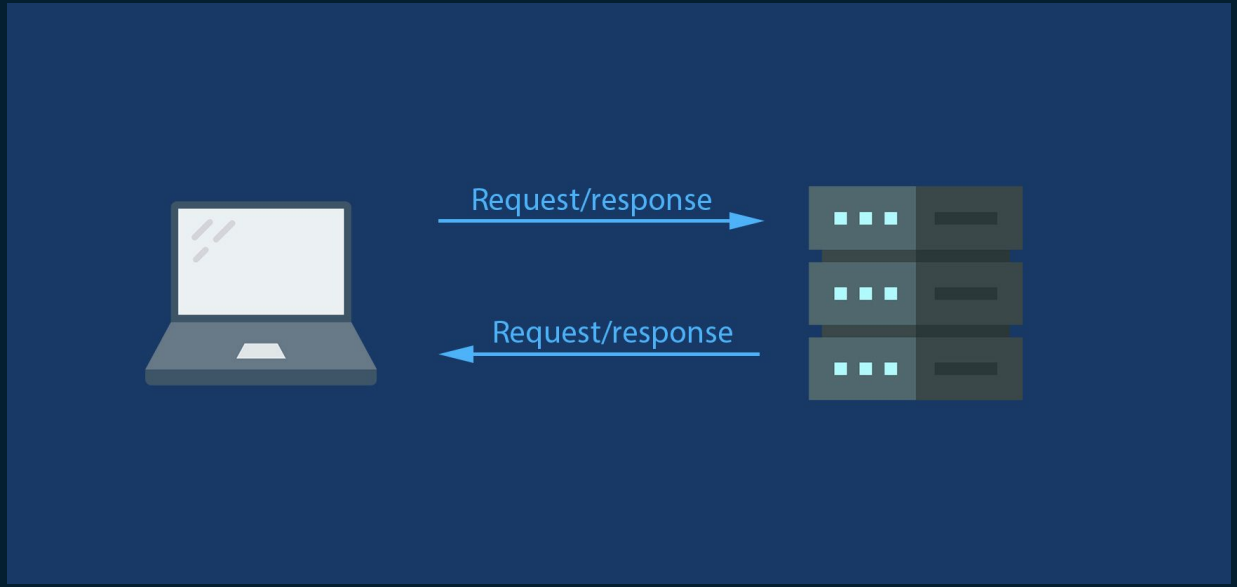
Server-side event allows the server push(asynchronously) the data to the client once the client-server connection is established. The server can then send data whenever a new piece of data is available. It can be called as a one-way publish-subscribe model.

Web sockets

The WebSocket API is an advanced technology that makes it possible to open a two-way interactive communication session between the user's browser and a server. With this API, you can send messages to a server and receive event-driven responses without having to poll the server for a reply.

- MDN

1. Client and server connect
2. Client and server can send messages
3. Uses TCP/IP (Transmission Control Protocol/Internet Protocol)
4. Whenever the server receives new information, it automatically sends it to the client



WS libraries by language

Socket.io
for NodeJS

websockets
for Python

SignalR
for .NET

Full list of WS libraries

https://github.com/facundofarias/awesome-websockets

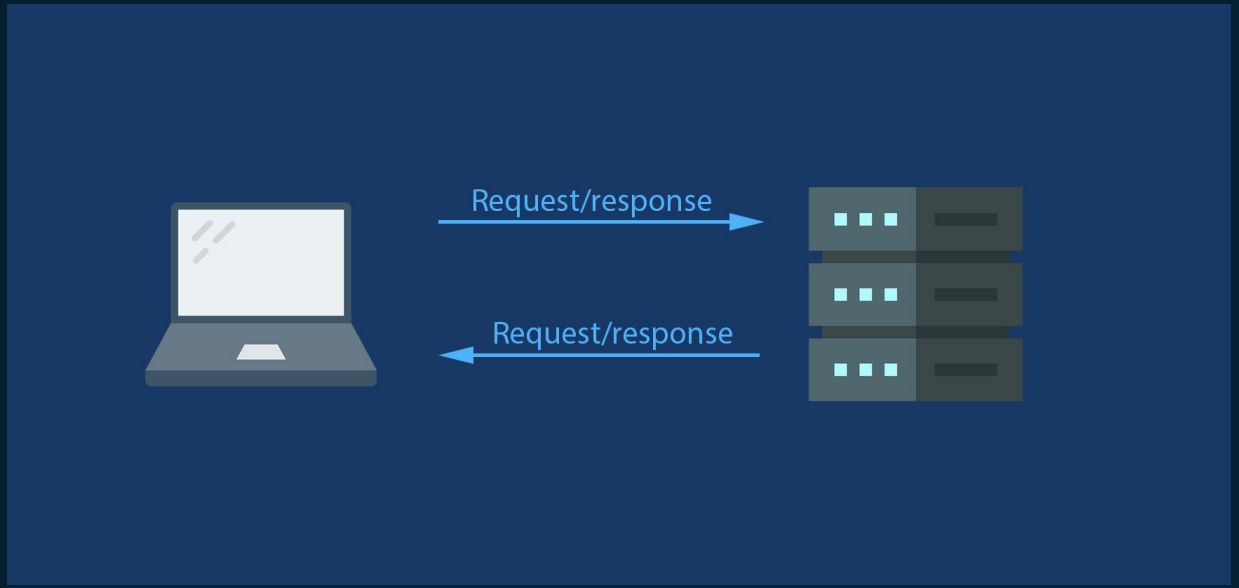
Good to read

https://codeburst.io/polling-vs-sse-vs-websocket-how-to-choose-the-right-one-1859e4e13bd9

Setup

What we need?

1. *Client side*
2. *Server side*



Client side

```
1 // HTML file
2 <script src="/socket.io/socket.io.js"></script>
3 // JS file
4 const socket = io(); // initializing a socket
5 socket.emit("eventName", message); // to send a message
6 socket.on("eventName", message); // to recieve a message
```

Server side

```
const express = require("express");
const app = express();
const http = require("http");
const server = http.createServer(app);
const io = require("socket.io")(server);

io.on("connection", (socket) => {
  console.log("connection:", socket.id);
});

server.listen(3000, () => {
  console.log("Listening on port 3000");
});

```

To code editor - build P2P chat