Web Sockets

O http é um protocolo muito conhecido e difundido na web, seu grande foco sempre foi permitir ao usuário fazer requisições ao servidor e este por sua vez responde com aquilo que o cliente deseja.

O http em si, possui uma conexão unidirecional, mas em 2005 surge o AJAX que permitia comunicações bidirecionais, mas o grande problema é que a atualização ainda dependia de interações do usuário, ou requisições constantes (são requisições “automatizadas”, pode ser uma função que requisita o dado acada 5 segundos por exemplo) para manter aquele dado atualizado.

Requisições frequentes apenas para manter um determinado dado atualizado pode sobrecarregar o servidor, a rede, entre outros. Surge então uma API chamada WebSocket

Como funciona o Websocket?

É uma API que estabelece conexões de “soquete” entre um browser e um servidor. Em resumo, podemos dizer que existe uma conexão persistente entre eles, onde qualquer um pode enviar dados quando desejar

Para começar uma conexão Websocket o cliente envia uma solicitação http chamada de **handshake**, basicamente podemos dizer que ela informa ao servidor que vamos começar uma conexão websocket.

Abrindo uma conexão webSocket do lado do cliente:

```
var socket = new WebSocket('ws://minha.url.com');
```

onde *minha.url.com* é o caminho que será aberto a conexão WebSocket

Transferência dos dados

Dados são transferidos com **mensagens**, as mensagen são divididas em **frames (quadros)**, e cada quadro é prefixado com 4 a 12 bytes sobre a carga

útil. O cliente só será notificado de uma nova mensagem após a chegada de todos os quadros e a mensagem tiver sido reconstruída

enviando mensagens

```
var socket = new Websocket('ws://websocket.example.com');
socket.onopen = function(event){
   socket.send('A mensagem');
};
```

`socket.onopen` → aqui estamos apenas dizendo que a conexão websocket está aberta

`socket.send(data)` → onde data é os dados que desejamos enviar para o servidor, que podem ser dos seguintes tipos:

- *string* → Uma sequência de textos
- *ArrayBuffer*
- *Blob*
- *TypedArray* ou um *DataView*

Recebendo mensagens do lado do cliente

```
socket.onmessage = function(event) {
     var message = event.data;
     console.log(message);
};
```

`socket.onmessage` → Sempre quando novos dados chegam para o cliente, é disparado o evento message, que por sua vez chamará a função `function(event)`

`event.data` → event é o que recebemos como parâmetro da função, ele possui algumas propriedades, entre elas o **`.data`** que nos retorna o conteúdo da mensagem que recebemos

Encerrando uma conexão websocket

```
If (socket.readyState === Websocket.OPEN){
    socket.close();
}
```