

WEBSOCKETY



Peter Gurský, peter.gursky@upjs.sk

Websockets

- Udržiavaný obojsmerný kanál na komunikáciu so serverom
 - nie len request-response ako pri bežnom HTTP
- cieľom je:
 - poskytnúť funkcionality TCP spojení cez HTTP protokol s využitím jeho šifrovania a bezpečnosti cez origin a prípadného proxy cez ten istý port ako bežná HTTP komunikácia
 - umožniť fungovať viacerým službám cez pomenovania
 - komunikácia cez rámce bez obmedzenia veľkosti
 - korektné ukončenie spojenia

Inštalácia

- WebSocket je súčasťou javascriptu, ale príliš nízkoúrovňový (ako TCP)
 - ▣ potrebujeme nadstavbový protokol, ktorý označuje, ktorá správa je aká
 - ▣ STOMP = Streaming Text Oriented Messaging Protocol
 - správy CONNECT, DISCONNECT, SUBSCRIBE, UNSUBSCRIBE, BEGIN, SEND, COMMIT, ABORT, ACK, NACK
- nainštalujeme knižnice
 - ▣ `npm install stompjs net`
 - ▣ `npm install @types/stompjs --save-dev`

Použitie

□ napojenie:

- **socket**: `WebSocket = new WebSocket('ws://localhost:8080/ws');`
- `stompClient`: `Stomp.Client = Stomp.over(socket);`
- `this.stompClient.connect(`
`header, (frame:Stomp.Frame)=>{}, error=>{});`
 - druhý parameter je callback funkcia, zavolaná, keď príde nový rámec s informáciou o úspešnom pripojení
- po napojení je spojenie vytvorené a
 - môžeme sa urobiť subscribe na dané topic-y message brokera
 - posielat' správy vybranému cieľu

□ odpojenie:

- `socket.close()`

počúvanie topic-u a posielanie správ

- `this.stompClient.subscribe('/topic/messages', msg => { .. });`
 - ▣ prvý parameter je meno topic-u na serveri z ktorého prichádzajú správy
 - ▣ druhý parameter je callback keď príde správa
 - ▣ tretí, nepovinný je hlavička

- `this.stompClient.send('/app/hello', {}, 'hello world');`
 - ▣ prvý parameter je cieľ, kam sa posiela správa
 - ▣ druhý parameter je hlavička
 - ▣ tretí parameter je samotná správa