

JavaScript - interwały i czas



Hello

Adrian Kukowski

Senior Frontend Developer at LIBRUS

Agenda

1. Ale jak to działa? Czyli Event Loop
2. Nie blokujący się UI
3. Przykład asynchroniczności
4. setTimeout i setInterval
5. A może własny interwał?
6. RequestAnimationFrame

Ale jak to działa? Czyli Event Loop

Zrobimy mały eksperyment



Trener

$2 + 2$



Dwójka ochotników



$3 + 3$

Ale jak to działa? Czyli Event Loop

The screenshot shows the 'loupe' event loop simulator interface. It features a code editor on the left with the following JavaScript code:

```
1 $.on('button', 'click', function onClick() {  
2   setTimeout(function timer() {  
3     console.log('You clicked the button!');  
4   }, 2000);  
5 });  
6  
7 console.log("Hi!");  
8  
9 setTimeout(function timeout() {  
10   console.log("Click the button!");  
11 }, 5000);  
12  
13 console.log("Welcome to loupe.");
```

Below the code editor is a button labeled "Click me!". To the right of the code editor are three panels: "Call Stack", "Web APIs", and "Callback Queue". The "Web APIs" panel contains the entry "\$.on('button', 'click', ...)". The "Callback Queue" panel is empty. A red circular arrow icon is positioned between the "Web APIs" and "Callback Queue" panels, indicating the flow of the event loop.

Nie blokujący się UI



Przykład asynchroniczności

```
const init = () => {  
  const button = document.querySelector("#button");  
  const handleClick = (event) => {  
    console.log(event);  
  };  
  button.addEventListener("click", handleClick);  
}  
window.onload = init;
```

setTimeout i setInterval

setTimeout - używany do wykonania jakiejś części kodu po określonym czasie

setInterval - używany do wykonania jakiejś części kodu w pętli każda pętla po określonym czasie

A może
własny
interwał?



RequestAnimationFrame



Tworzymy animację i porównamy z setInterval



Dzięki



adrian-kukowski

@ kontakt@kukowskiadrian.pl

kukowskiadrian.pl