

# JavaScript: Interval e Timeout



DEVinHouse

Parcerias para desenvolver a sua carreira

**SENAI**

<LAB365>

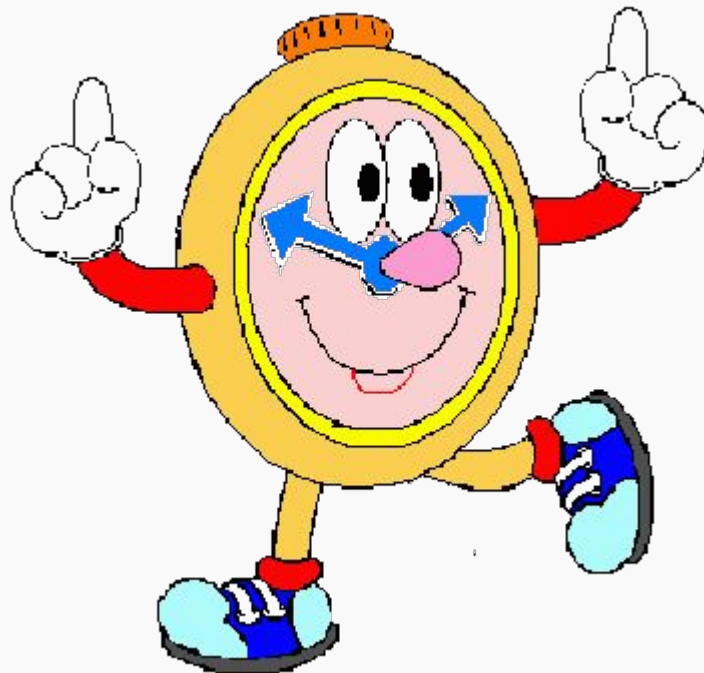
# Timeout e Interval

- Interval
- Timeout

## Porque estudar?

Em algumas situações precisamos realizar uma **ação** no código:

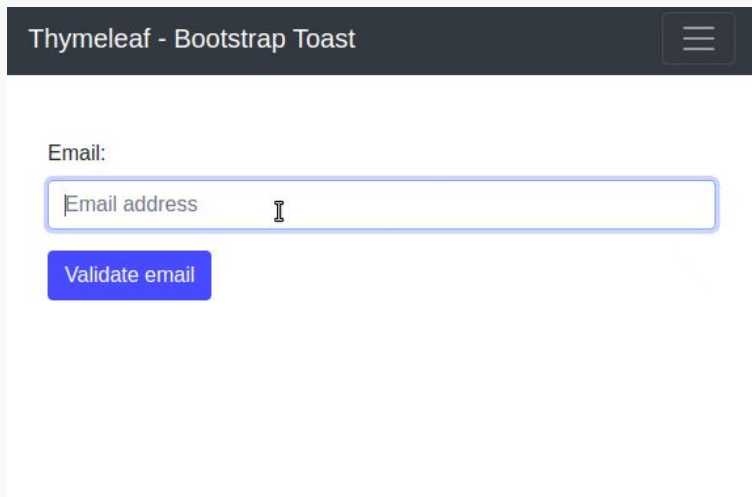
- depois de um tempo específico  
(como mostrar uma mensagem de erro ou sucesso na tela e depois tirá-la da tela)
- ou realizar uma ação repetidas vezes  
(como mudar o horário de um relógio a cada 1 segundo)



# TIMEOUT (pt-BR: tempo limite)

Usamos a função **setTimeout()** para executar um **bloco específico** de código **uma vez**, depois de um **determinado tempo**.

Por exemplo: mostrar uma mensagem de *sucesso* ou *erro* na tela após determinado tempo



The screenshot shows a web form titled "Thymeleaf - Bootstrap Toast". It features a label "Email:" above a text input field. The input field has a placeholder "Email address" and a cursor. Below the input field is a blue button labeled "Validate email".

## Sintaxe

**setTimeout()** é uma função que, normalmente recebe 2 parâmetros:

- outra função
- intervalo de tempo em que será executada (**1 só vez**)

```
setTimeout(function() {  
    ...  
}, interval)
```

## Exemplos

Podemos declarar a função recebida de forma:

- nomeada

```
setTimeout(function oi() {  
    alert("Olá! De uma função com nome!");  
}, 2000);
```

- não nomeada

```
setTimeout(function() {  
    alert("Olá, Mundo!");  
}, 2000);
```

Ou ainda escrever uma **função externa** (nomeada) e inseri-la como 1º parâmetro

```
function ola() {  
    alert("Olá! De uma função definida separadamente!");  
}  
  
setTimeout(ola, 5000);
```

## + parâmetros

setTimeout() pode receber um **sequência de parâmetros** que serão passados para a função a ser executada após o tempo.

Por exemplo:

```
function dizerOla(quem) {  
  alert("Olá, " + quem + "!");  
}  
  
setTimeout(dizerOla, 2000, "Mundo");
```

## clearTimeout()

Apesar de na prática não ser muito usado, pode acontecer de você precisar **limpar o setTimeout()** para que não seja chamado.

Por exemplo:

```
let numero = 0  
const apareceConsoleDoValorDeX = function() {  
  console.log('O valor do número é ')  
}  
const meuSetTimeout = setTimeout(apareceConsoleDoValorDeX, 1500)  
  
numero = 5  
if (numero !== 0) {  
  clearTimeout(meuSetTimeout)  
}
```

Construa um site com  
um **botão** que, ao ser **clicado**,  
mostre um **texto na tela após 3 segundos**.

Opcional: e após 5 segundos a informação deve desaparecer da tela.

# INTERVALO DE AULA

## **DEV!**

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

**Início:** 20:20

**Retorno:** 20:40





# INTERVAL

Usamos a função **setInterval()** para executar um **bloco específico** de código **repetidamente**, em um **intervalo fixo de tempo** entre cada chamada.

Por exemplo: mudar os números de um relógio a cada 1 segundo



00:00

## Sintaxe

`setInterval()` também **é uma função** que, normalmente recebe 2 parâmetros:

- **outra função**
- **intervalo** de tempo **entre cada chamada** (*várias vezes*)

```
setInterval(() => {  
    // Run every 100 milliseconds  
}, 100);
```

## + parâmetros

Mas o `setInterval()` também pode receber um ***sequência de parâmetros*** que serão passados para a função a ser executada entre cada chamada.

Por exemplo:

```
function fuiChamada(quem) {  
    console.log(`Fui chamada por ${quem}`)  
}  
  
setInterval(fuiChamada, 5000, "Rosana")
```

## clearInterval()

Como o setInterval() é chamado continuamente, ele pode **comprometer a memória do computador do usuário** caso a função invocada seja complexa e/ou o intervalo de tempo entre as chamadas sejam curtos.

Por essa razão, é possível usar o **clearInterval()** para **parar o programa** assim que desejado.

Por exemplo:

```
function fuiChamada(quem) {  
  console.log(`Fui chamada por ${quem}`)  
}  
  
// setInterval  
const chamarFuncao = setInterval(fuiChamada, 5000, "Rosana")  
  
// clearInterval  
const botao = document.getElementById("botao");  
botao.addEventListener("click", function() {  
  clearInterval(chamarFuncao)  
})
```

Construa um site com um **relógio** que atualize automaticamente o horário na tela.

Opcional: a cada segundo a cor do horário deve alternar entre verde e rosa.

Dicas: pesquise por ***new Date()*** para facilitar a resolução.

- [setTimeout\(\) - Web APIs | MDN](#)
- [WindowOrWorkerGlobalScope.setInterval\(\) - APIs da Web | MDN](#)

## AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

[Clique aqui](#) ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.





# DEVinHouse

Parcerias para desenvolver a sua carreira

**OBRIGADO!**



<LAB365>