13

As múltiplas facetas do this

Vejamos o seguinte código!

```
class Pessoa {

    constructor(nome) {
        this.nome = nome;
    }
}

function exibeNome() {
    alert(this.nome);
}

let pessoa = new Pessoa('Salsifufu');

exibeNome('Lampreia'); // PRIMEIRA CHAMADA <=============

exibeNome = exibeNome.bind(pessoa);

exibeNome(); // SEGUNDA CHAMADA <=============
```

O que será exibido na PRIMEIRA e SEGUNDA chamada respectivamente?

- ✓ `undefined` e `Salsifufu`
- B `Lampreia` e `undefined`
- C `undefined` e `Lampreia`.
- D `Salsifufu` e `undefined`

A resposta correta é **undefined** e **Salsifufu**.

Na primeira chamada, o parâmetro `Lampreia` é ignorado pela função, pois a função não recebe parâmetros. Não acontece nenhum erro, mesmo a função não recebendo o parâmetro, uma característica do JavaScript. Sendo assim, quando `exibeNome('Lampreia')` é chamado, o `this` na verdade é `window`, o escopo global, e nele não temos o valor `nome`.

Contudo, quando executamos a linha:

```
exibeNome = exibeNome.bind(pessoa);
```

Estamos recebendo uma referência para uma nova função, que passa a ter o objeto `pessoa` como `this`. Ou seja, a função `bind`, presente em todas as funções, permite indicar qual será o valor de `this` quando ela for executada, em nosso caso `pessoa`. Como `pessoa` possui a propriedade `nome`, será exibido no alerta o valor **Salsifufu**.

É por isso que fizemos ao longo deste capítulo:

```
let $  = document.querySelector.bind(document);
```

O retorno da função `bind` é a função `querySelector`, que tem como contexto o `document`, ou seja, seu `this` será `document`. Se tivéssemos feito apenas:

```
let $  = document.querySelector;
```

A variável `$` passa a ser `querySelector`, mas seu `this` deixará de ser `document`, porque estará sendo executada fora deste contexto.

PRÓXIMA ATIVIDADE