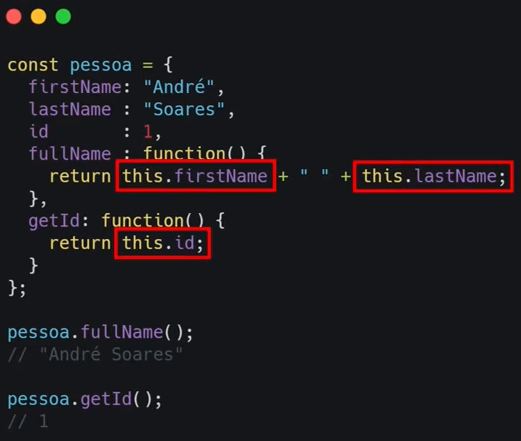
**FUNCOES**

**This**

* a palavra reservada **this** é uma referência de contexto
* no exemplo abaixo, this refere-se ao objeto pessoa



fullName é um método

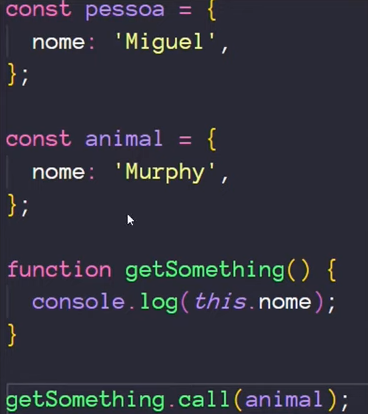
Quando se utilizar **this** dentro de um método, ele vai se referir ao objeto pai desse método

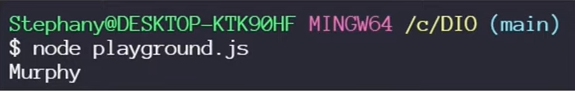
* Seu valor pode mudar de acordo com o lugar no código onde foi chamada



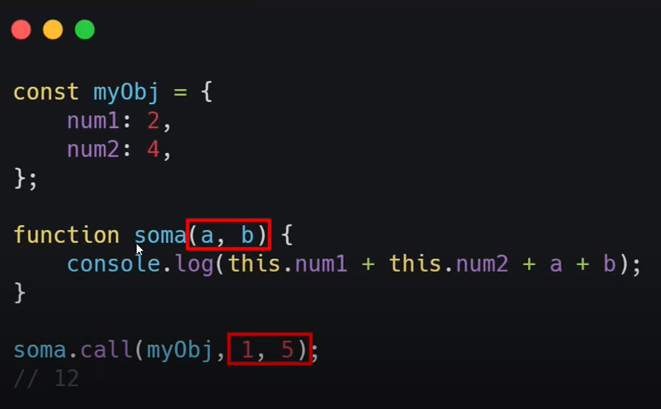
**Manipulando seu valor**

**Call**





* É possível passar parâmetros para essa função separando-os por vírgulas



**Apply**

const pessoa = {

    nome: 'Miguel',

};

const animal = {

    nome: 'Godi',

};

function getSomething() {

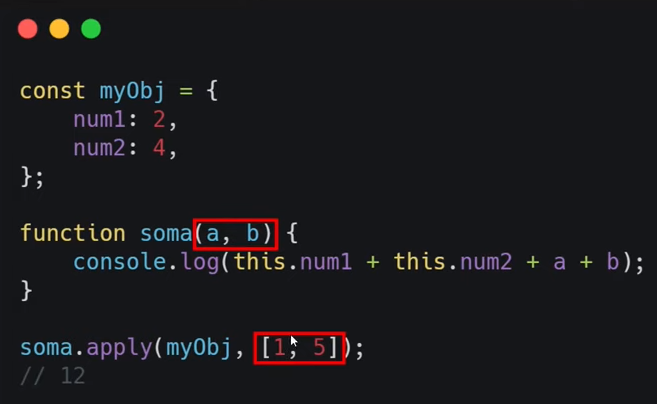
    console.log(*this*.nome);

}

getSomething.apply(animal);

//resultado: Godi

* É possível passar parâmetros para essa função dentro de um array



**Bind**

* clona a estrutura da função onde é chamada e aplica o valor do objeto passado como parâmetro
* const retornaNomes = function () {
* return *this*.nome;
* };
* let bruno = retornaNomes.bind({ nome: 'Bruno'});
* bruno();

**Arrow Functions**

**Sintaxe:**

const helloWorld = () => {

    return "Hello World";

}

const helloWorld = () => "Hello World"; //forma resumida

* Caso existe apenas uma linha, pode dispensar as chaves e o return
* Caso exista apenas um parâmetro, pode dispensar os parênteses
* const soma = (*a*, *b*) => *a* + *b*;
* soma (4,6);
* //10
* const soma = *a* => *a*;
* soma (4);
* //4

**Arrow function NÃO faz hoisting!**

**Outras restrições:**

* “this” sempre será o objeto global. Métodos para modificar seu valor não irão funcionar;
* não existe o objeto “arguments”;
* o construtur (ex: new meuObjeto() ) também não pode ser utilizado