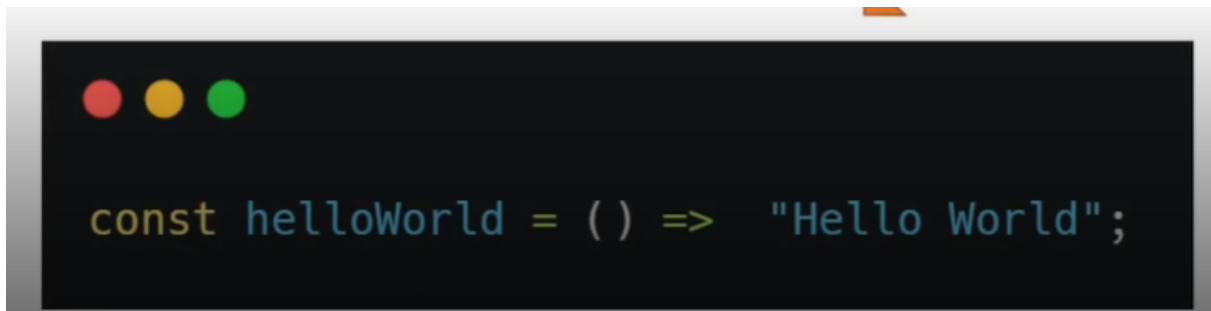


Uma Arrow Function ela é representada por => por isso que ela se chama Arrow Function



```
const helloWorld = () => "Hello World";
```

Os códigos abaixo fazem a mesma coisa:



```
const helloWorld = function(){  
  return "Hello World";  
}
```

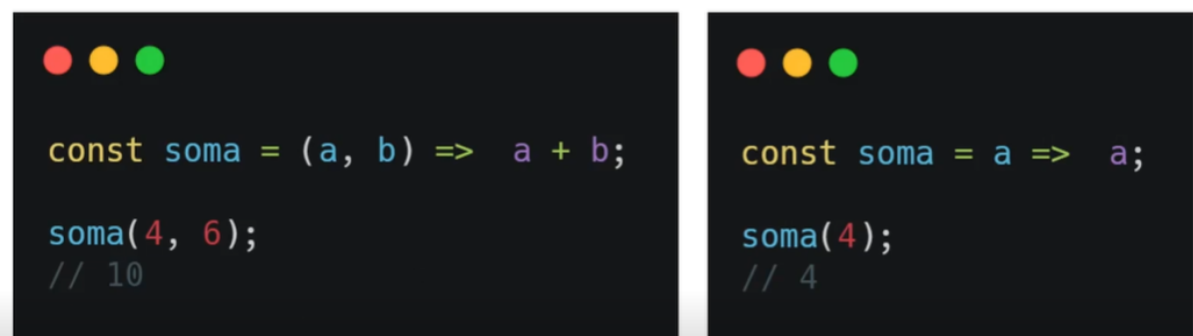
→

```
const helloWorld = () => {  
  return "Hello World";  
}
```

A primeira é o que chamamos de **Função Anônima**

Arrow Functions tem um recurso muito interessante que é o fato de você não precisar, quando ela tem 1 linha só, você não precisa escrever o return ele vai já presumir que você quer retornar aquilo que vem logo após a =>. Então quando a sua Arrow Function for uma função muito pequena e só tiver 1 linha você não precisa nem do return e nem das {}.

Você só coloca do lado => a sua operação ou o que você quiser retornar e ela SEMPRE está aliada a uma variável, ela sempre terá o valor armazenado.



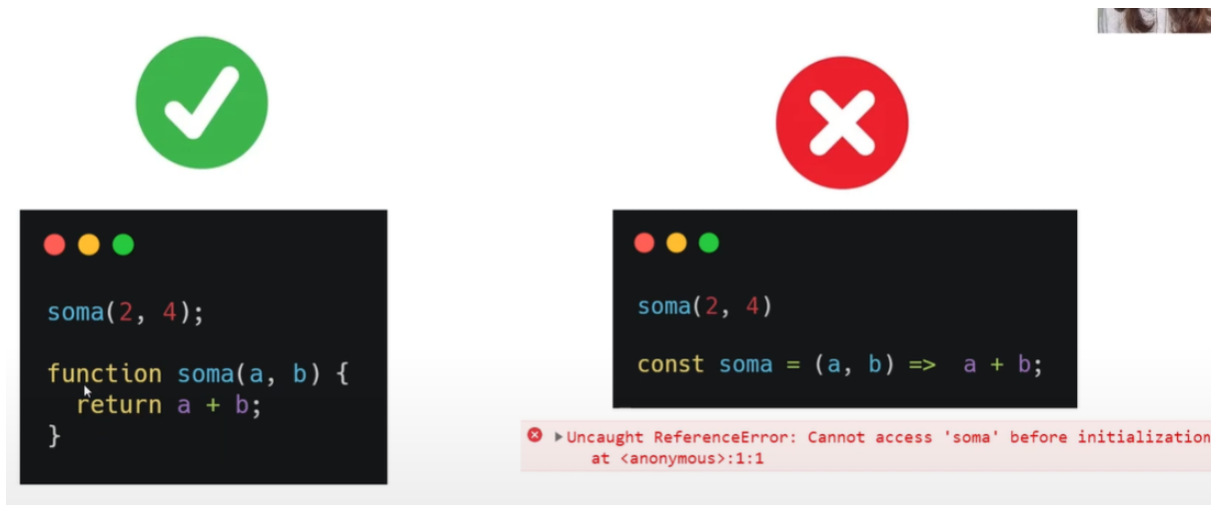
```
const soma = (a, b) => a + b;  
soma(4, 6);  
// 10
```

```
const soma = a => a;  
soma(4);  
// 4
```

Aqui a gente tem algumas regrinhas para você seguir da Arrow Function, na verdade são algumas possibilidades:

- Caso exista apenas 1 linha, pode dispensar as chaves e o return

- Caso exista apenas 1 parâmetro, pode dispensar os parâmetros
Arrow Function NÃO faz Hoisting



Quando a gente falou no curso de Variáveis e Tipos, a gente viu que a palavra reservada **var** ela faz hoisting que é basicamente a variável ou a função antes de você declarar ela, porque o JS vai entender que isso tem que ser colocado pra cima então ele vai mandar pra cima automaticamente quando executar o código e você vai poder chamar a sua função. Então funções comuns fazem hoistings, mas a Arrow Function justamente por ela sempre ter o valor armazenado dentro de uma constante **const** ela não faz, então você sempre tem que declarar ela antes de chamá-la.

Etapa 2: Outras restrições

Por mais que as Arrow Functions equivalem a funções normais, na verdade algumas coisas que a gente faz em funções normais não podem ser feitas em Arrow Functions.

- O valor **this** sempre vai ser do Objeto Global, você não vai poder utilizar os métodos que nós aprendemos anteriormente na Arrow Function, não vai funcionar o `call`, `apply` e o `bind` no Arrow Function, ele tem que ser utilizado em uma função comum;
- Não existe o objeto “arguments” que é o objeto que vai juntar dentro de uma lista [] todos os argumentos que você mandou pra função;
- O construtor que você declarar variáveis ex: `new MeuObjeto()` também não pode ser utilizado.

Uma coisa que a gente sempre diz é: Não faça Arrow Functions sem serem métodos de objetos, sempre utilize uma função comum.