

Bitdefender

GravityZone Business Security

Proteja 15 dispositivos por menos de US \$ 375 por ano

Compre online até 100 endpoints

ECONOMIZE

30%

AJA AGORA

# Função de seta JavaScript

< Anterior

Próximo >

As funções de seta foram introduzidas no ES6.

As funções de seta nos permitem escrever uma sintaxe de função mais curta:

Antes:

```
hello = function() {  
  return "Hello World!";  
}
```

Tente você mesmo "

Com função de seta:

```
hello = () => {  
  return "Hello World!";  
}
```

Tente você mesmo "

Ele fica mais curto! Se a função tiver apenas uma instrução e a instrução retornar um valor, você pode remover os colchetes e a `return` palavra-chave:

Funções de seta Valor de retorno por padrão:

```
hello = () => "Hello World!";
```

Tente você mesmo "

**Observação:** isso funciona apenas se a função tiver apenas uma instrução.

If you have parameters, you pass them inside the parentheses:

Arrow Function With Parameters:

```
hello = (val) => "Hello " + val;
```

Try it Yourself »

In fact, if you have only one parameter, you can skip the parentheses as well:

Arrow Function Without Parentheses:

```
hello = val => "Hello " + val;
```

Try it Yourself »

## What About `this`?

The handling of `this` is also different in arrow functions compared to regular functions.

In short, with arrow functions there are no binding of `this`.

In regular functions the `this` keyword represented the object that called the function, which could be the window, the document, a button or whatever.

With arrow functions the `this` keyword *always* represents the object that defined the arrow function.

Let us take a look at two examples to understand the difference.

Both examples call a method twice, first when the page loads, and once again when the user clicks a button.

The first example uses a regular function, and the second example uses an arrow function.

The result shows that the first example returns two different objects (window and button), and the second example returns the window object twice, because the window object is the "owner" of the function.

Example

With a regular function `this` represents the object that *calls* the function:

```
// Regular Function:  
hello = function() {  
  document.getElementById("demo").innerHTML += this;  
}  
  
// The window object calls the function:  
window.addEventListener("load", hello);  
  
// A button object calls the function:  
document.getElementById("btn").addEventListener("click", hello);
```

Try it Yourself »

Example

With an arrow function `this` represents the *owner* of the function:

```
// Arrow Function:  
hello = () => {  
  document.getElementById("demo").innerHTML += this;  
}  
  
// The window object calls the function:  
window.addEventListener("load", hello);  
  
// A button object calls the function:  
document.getElementById("btn").addEventListener("click", hello);
```

Try it Yourself »

Remember these differences when you are working with functions. Sometimes the behavior of regular functions is what you want, if not, use arrow functions.

## Browser Support

A tabela a seguir define as primeiras versões do navegador com suporte total para funções de seta em JavaScript:

				
Chrome 45	Edge 12	Firefox 22	Safari 10	Opera 32
Sep, 2015	Jul, 2015	May, 2013	Sep, 2016	Sep, 2015

< Anterior

Próximo >


PROPAGANDA



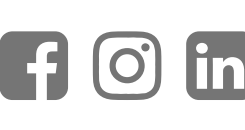


Start Now


COLOR PICKER



COMO NÓS



Obtenha a certificação completando um curso hoje!



iniciar

JOGO DE CÓDIGOS



Jogar um jogo



Reportar erro

Fórum

Cerca de

Comprar

Principais tutoriais

Tutorial HTML Tutorial  
CSS Tutorial  
JavaScript  
Como fazer Tutorial  
SQL Tutorial  
Python Tutorial  
W3.CSS Tutorial  
Bootstrap Tutorial  
PHP Tutorial  
Java Tutorial  
C ++ Tutorial  
jQuery Tutorial

Referências principais

HTML Reference  
CSS Reference  
JavaScript Reference  
SQL Reference  
Python Reference  
W3.CSS Reference  
Bootstrap Reference  
PHP Reference  
HTML Colors  
Java Reference  
Angular Reference  
jQuery Reference

Top Examples

HTML Examples  
CSS Examples  
JavaScript Examples  
How To Examples  
SQL Examples  
Python Examples  
W3.CSS Examples  
Bootstrap Examples  
PHP Examples  
Java Examples  
XML Examples  
jQuery Examples

Web Courses

HTML Course  
CSS Course  
JavaScript Course  
Front End Course  
SQL Course  
Python Course  
PHP Course  
jQuery Course  
Java Course  
C++ Course  
C# Course  
XML Course

Get Certified »

