Os segredos do JavaScript

APRENDA TÉCNICAS AVANÇADAS DA LINGUAGEM

POR MATHEUS BATTISTI
HORA DE CODAR

SOBRE O AUTOR

Matheus Battisti é desenvolvedor há mais de 6 anos e ama a tecnologia da informação como um todo.

Está sempre em busca de um aprendizado constante e evolutivo.

Atua como desenvolvedor Full Stack, desenvolvendo e-commerces.

É criador do Hora de Codar, que hoje passa de 20 mil alunos.

Python é uma de suas linguagens preferidas, já utilizou para Data Science como também desenvolvimento web.

Seu objetivo é sempre passar o máximo de conhecimento possível, unindo toda a experiência de seus estudos e também de sua carreira como desenvolvedor.

INTRODUÇÃO

Este eBook é destinado a quem deseja melhorar o seu entendimento e código em JavaScript.

Feito para apresentar conceitos e técnicas importantes da linguagem, que não são comuns em outras.

Você vai aprender com exemplos os principais recursos que oferecem uma maior potencialização ao seu software.

Os recursos aqui apresentados podem ser utilizados para garantir uma manutenção melhor no seu código e também uma robustez maior para o seu sistema.

Além de poupar linhas e mais linhas de código, tentando recriar funções similares para atender a alguns problemas que você se depara ao programar em JS.

Está pronto? Vamos começar!

PROTOTYPES

JavaScript é uma linguagem que possui Prototypes, então é de suma importância compreender este tópico.

O Prototype funciona como uma herança, sempre que um objeto é criado, ele também terá a propriedade __proto__ inserida nele.

Podemos adicionar propriedades e métodos no Prototype para que seja acessível em outro objeto.

Veja o Prototype na prática:

```
class Pessoa{}

Pessoa.prototype.falar = function() {
    console.log("Olá");
}

const pedro = new Pessoa;

pedro.falar();
```

Neste caso, qualquer objeto criado a partir da classe Pessoa, terá acesso ao método falar, que está no Prototype da mesma.

CLASSES

No exemplo anterior criamos uma classe no JavaScript, o que não era possível até sua versão ES5.

Agora podemos utilizar este recurso que vem do paradigma de Orientação a Objetos, temos até a possibilidade de utilizar um construtor.

Recursos que faltavam na linguagem em suas versões passadas, veja:

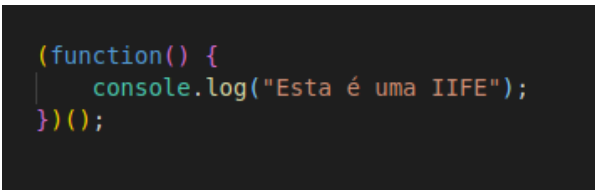
```
class Pessoa{
    constructor(nome, idade) {
    this.nome = nome;
    this.idade = idade;
  }
}


const pedro = new Pessoa("Pedro", 21);

console.log(pedro.nome, pedro.idade);
```

Perceba que no fim do exemplo também é exibida a forma de acessar propriedades de um objeto criado por uma classe.

IIFE

Um recurso pouco explorado pelos iniciantes da linguagem são as IIFE (Immediately Invoked Function Expression).

Ou seja, funções que são executadas automaticamente ao serem identificadas na execução do seu código, veja um exemplo:

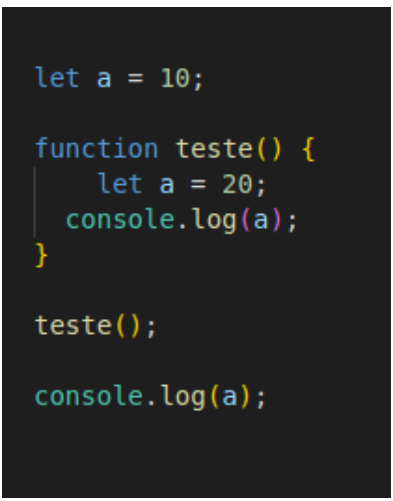


A IIFE é caracterizada por envolver a função a ser executada entre parênteses e depois chamar a mesma com o ();

ESCOPO

Outro conceito de altíssima importância no JS é entender o escopo das expressões que utilizamos no nosso código.

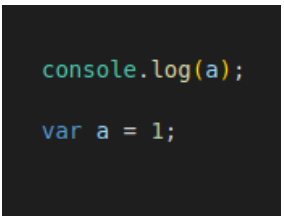
O escopo global é o mais alto de todos, onde executamos o código, porém podemos criar outros escopos como funções, veja um exemplo:



Neste caso a função teste não muda o valor da variável a do escopo global, pois as das variáveis estão em diferentes escopos e possuem diferentes valores.

HOISTING

Mais um conceito importante, o hoisting tem a finalidade de identificar as declarações de variáveis, não inicializá-las.



Neste caso teremos a como undefined, pois o hoisting identificou ela, mas não a inicializou.

Isso por que futuramente ela terá valor atribuído no código.

CALLBACKS

As callbacks no JavaScript servem como funções em argumentos, que serão executadas após uma determinada ação.

Vejamos um exemplo clássico de callback:

```
setTimeout(function() {
    console.log("Ativando a callback")
}, 300);
```

No método setTimeout temos um argumento que é uma função callback.

A mesma será executada ao fim do tempo determinado pelo setTimeout.

OBJETOS DO JAVASCRIPT

Conhecer as APIs do JavaScript vai transformar a sua forma de programar em JS.

Os mais utilizados são:

- Math - operações e cálculos matemáticos;
- Date - manipulação de datas;
- RegExp - expressões regulares;
- JSON - manipulação de dados em JSON;
- Array - métodos para arrays;
- String - métodos para textos;
- Number - métodos para valores númericos;

DESTRUCTURING

O destructuring é um recurso utilizado para desestruturar arrays e objetos, na maioria das vezes.

A partir de uma expressão podemos criar várias variáveis com os valores destes conjuntos de dados, veja:

```
let arr = [1, 2, 3];

[a, b, c] = arr;

console.log(a, b, c);
```

O destructuring em arrays é realizado utilizando o [] e em objetos o {}

OPERADORES DE COMPARAÇÃO

No JavaScrip temos o operador == e === para realizar comparações.

A grande diferença é que o === testa também o tipo da variável, além do valor.

Acostume-se a utilizar o ===, pois ele deixará seus softwares mais confiáveis do que o ==.

A ES6

ES6 é uma versão atual da linguagem JavaScript, vários conceitos deste eBook tiveram origem nesta versão.

Além disso, vários conceitos importantes para a orientação a objetos também chegaram com ela, como:

- Classes;
- constructor;
- Rest e Spread;
- Arrow functions;
- Destructuring;
- Iterators;
- Generators;
- Promises;

É interessante que cada um destes tópicos seja estudado a fundo para o melhor entendimento da linguagem.

Obrigado

Enquanto houver pessoas dispostas
a aprender, eu estarei criando
conteúdo para enriquecer ainda
mais os seus conhecimentos

Matheus Battisti

Hora de Codar