14 - Arrow Functions

Vamos se aprofundar sobre Arrow Functions em JavaScript. Esse é um tópico super legal e importante porque as Arrow Functions não só simplificam a sintaxe, mas também têm algumas peculiaridades que você precisa entender.

Introdução às Arrow Functions

Arrow Functions, introduzidas no ES6, são uma forma mais curta de escrever funções em JavaScript. Elas são especialmente úteis para funções anônimas e têm uma sintaxe mais enxuta.

Sintaxe Básica

```
let somar = (a, b) => a + b;
console.log(somar(2, 3)); // 5
```

Aqui, `somar` é uma Arrow Function que recebe dois parâmetros `a` e `b` e retorna a soma deles. Note que não usamos a palavra-chave `function` e o retorno é implícito.

Funções Sem Parâmetros

```
let saudar = () => console.log("Olá, mundo!");
saudar(); // Olá, mundo!
```

Se a função não tem parâmetros, ainda precisamos de parênteses vazios.

Funções com Corpo de Bloco

```
let multiplicar = (a, b) => {
  let resultado = a * b;
  return resultado;
};
console.log(multiplicar(2, 4)); // 8
```

Para funções com mais de uma expressão, usamos chaves e a declaração `return`.

Características Importantes das Arrow Functions

Arrow Functions não são apenas uma sintaxe mais curta. Elas têm características únicas que as diferenciam das funções tradicionais.

`this` nas Arrow Functions

```
function Pessoa() {
  this.idade = 0;

  setInterval(() => {
    this.idade++;
    console.log(this.idade);
  }, 1000);
}

new Pessoa(); // Incrementa a idade a cada segundo
```

Em Arrow Functions, `this` é léxico, ou seja, refere-se ao contexto no qual a função foi criada, ao contrário das funções tradicionais.

Não Podem Ser Construtores

Arrow Functions não podem ser usadas como funções construtoras. Tentar fazer isso resultará em um erro.

```
let Pessoa = () => {};
let p = new Pessoa(); // TypeError: Pessoa is not a constructor
```

Usos Práticos das Arrow Functions

As Arrow Functions são especialmente úteis em callbacks e funções que exigem contexto léxico de `this`.

Em Métodos de Array

```
let numeros = [1, 2, 3, 4, 5];
let dobrados = numeros.map(numero => numero * 2);
console.log(dobrados); // [2, 4, 6, 8, 10]
```

Arrow Functions são ótimas para operações em arrays, como `map`, `filter` e `reduce`.

Callbacks

```
setTimeout(() => console.log("Olá depois de 1 segundo"), 1000);
```

Aqui, usamos uma Arrow Function como callback para `setTimeout`.

Exercícios

1. **Calculadora Simples: Crie Arrow Functions para as operações básicas de matemática: soma, subtração, multiplicação e divisão. Cada função deve aceitar dois parâmetros e retornar o resultado da operação.**
2. **Função de Saudação Personalizada: Escreva uma Arrow Function que recebe um nome como parâmetro e retorna uma saudação personalizada.**
3. **Contador com Arrow Function: Use uma Arrow Function dentro de um `setInterval` para criar um contador que imprime um número a cada segundo.**
4. **Filtrar Números Pares: Dado um array de números, use uma Arrow Function com o método `filter` para criar um novo array apenas com números pares.**
5. **Conversor de Temperatura: Crie uma Arrow Function que converte a temperatura de Celsius para Fahrenheit.**
6. **Função de Ordenação: Implemente uma Arrow Function que recebe um array de strings e retorna um novo array com os itens ordenados alfabeticamente.**