Unidade VI: Objetos soulcode.com

**1 Objetos**

Os objetos são uma das características centrais da linguagem JavaScript, per mitindo que você armazene dados e funcionalidades em uma estrutura flexí vel. Usamos os objetos para representar dados complexos.

**1.1 Criação de Objetos**

Um objeto é formado por propriedades e métodos, eles representam as carac terísticas e ações que um objeto pode ter. Podemos criar um objeto podemos usar a forma literal entre {}.

Listagem 1: Sintaxe literal de um objeto

1 const objeto = { // Criacao de um objeto

2 propriedade1: valor1, // Propriedades e valores

3 propriedade2: valor2,

4 metodo: function() { // Metodo dentro do objeto

5 // codigo do metodo

6 }

7 };

Veremos em mais detalhes adiante como definir e manipular as propriedades e métodos de um objeto.

**1.2 Propriedades**

Os objetos JavaScript consistem em pares de propriedade-valor. As proprieda des podem ser acessadas e modificadas usando a notação de ponto (objeto.propriedade) ou a notação de colchetes (objeto['propriedade']).

Listagem 2: Propriedades

1 const pessoa = {

2 nome: 'Joao',

3 idade: 30

4 };

5

6 console.log(pessoa.nome); // Saida: Joao

7 pessoa.idade = 31;

8 console.log(pessoa['idade']); // Saida: 31

9

10 pessoa.salario = 5500; // nova propriedade no objeto

11 pessoa['salario'] += 50; // ou, pessoa.salario += 50

No código acima, podemos ver que é possível alterar o valor específico de uma propriedade usando uma das notações de acesso. Dessa forma, temos uma estrutura altamente flexível e que nos possibilita inserir novas propriedades.

**1.3 Métodos**

Os métodos são funções definidas dentro de um objeto. Eles podem ser cha mados usando a notação de ponto, assim como as propriedades.

1

Unidade VI: Objetos soulcode.com

Listagem 3: Métodos

1 const pessoa = {

2 nome: 'Joao',

3 idade: 30,

4 cumprimentar: function() {

5 console.log("Ola!!");

6 }

7 };

8

9 pessoa.cumprimentar();

Observe a linha 4, definimos a propriedade **cumprimentar**, no entanto seu valor é uma *function expression*. Essa function não possui um nome definido, mas para invocá-la basta fazer como na linha 9.

No geral, os métodos são idênticos as funções estudadas anteriormente, então todas as características estão presentes nela. Podemos aplicar parâ metros:

Listagem 4: Métodos com parâmetros

1 const pessoa = {

2 nome: 'Joao',

3 idade: 30,

4 cumprimentar: function(nome) {

5 console.log("Ola " + nome + "!");

6 }

7 };

8

9 pessoa.cumprimentar("Pedro");

**1.3.1 Uso do this**

Uma característica importante dos objetos, é que podemos referenciar o ob jeto em que o método pertence usando a palavra reservada *this*.

Listagem 5: Uso do this

1 const pessoa = {

2 nome: 'Joao',

3 idade: 30,

4 falar: function() {

5 console.log('Ola, meu nome eh ' + this.nome + ' e eu tenho ' + this.idade + ' anos.');

6 },

7 envelhecer: function() {

8 this.idade++;

9 }

10 };

11

12 pessoa.falar();

13 pessoa.envelhecer();

Com o uso do this, podemos acessar as propriedades e métodos dentro do objeto podendo então fazer algumas manipulações interessantes.

2

Unidade VI: Objetos soulcode.com

O **this** é uma palavra-chave especial que representa o contexto de exe cução atual. Ele é determinado no momento da chamada do método e varia dependendo de como o método é chamado. Em outras palavras, o this está vinculado ao objeto no qual o método é chamado. O exemplo a seguir, mostra que o **this**, pode se "perder".

Listagem 6: Vínculo do this

1 const metodoFalar = pessoa.falar;

2 metodoFalar(); // Saida: Ola, meu nome eh undefined e eu tenho undefined anos.

Nesse caso, o valor de this.nome e this.idade é undefined, porque metodoFa lar é uma função isolada e o this não está mais vinculado ao objeto pessoa.

**2 Construtores**

Você também pode criar objetos usando funções construtoras. Uma função construtora é uma função que define propriedades e métodos para um objeto criado usando o operador **new**.

Listagem 7: Construtor

1 function Pessoa(nome, idade) {

2 this.nome = nome;

3 this.idade = idade;

4 }

5

6 const pessoa1 = new Pessoa('Maria', 25);

7 console.log(pessoa1.nome); // Saida: Maria

8 console.log(pessoa1.idade); // Saida: 25

Nessa sintaxe, a principal diferença é a forma de criar o objeto, nas forma lite ral passamos dentro do objeto as propriedades e valor, já no construtor pas samos as propriedades como parâmetro para que o objeto seja construído.

**3 Sintaxes úteis**

**3.1 Operador spread**

O operador spread é representado por três pontos consecutivos (...). Ele per mite "espalhar"os elementos de um objeto. Em outras palavras, podemos unir objetos em um só.

Listagem 8: Operador spread

1 const pessoa = {nome: "Pedro", idade: 31};

2 const pessoaAtualizada = {...pessoa, idade: 32, telefone: "(00) 00000-0000"}; 3 console.log(pessoaAtualizada); // {nome: 'Pedro', idade: 32, telefone: '(00) 00000-0000'}

3

Unidade VI: Objetos soulcode.com

**3.2 Desestruturação**

A desestruturação é uma forma concisa de extrair valores de objetos ou arrays e atribuí-los a variáveis individuais. Ela permite acessar facilmente elementos aninhados ou propriedades de objetos sem a necessidade de escrever várias linhas de código.

Listagem 9: Desestruturação

1 const pessoa = {

2 nome: 'Maria',

3 idade: 30,

4 cidade: 'Sao Paulo'

5 };

6

7 const { nome, idade, cidade } = pessoa;

8 console.log(nome, idade, cidade); // Saida: Maria 30 Sao Paulo

**3.3 Propriedades computadas**

É possível utilizar expressões para definir o nome das propriedades de um objeto usando colchetes durante a criação ou modificação de propriedades.

Listagem 10: Propriedades computadas

1 const chave = 'nome';

2 const valor = 'Joao';

3

4 const objeto = {

5 [chave]: valor

6 };

7

8 console.log(objeto); // Saida: { nome: 'Joao' }

Nesse exemplo, a propriedade nome é criada dinamicamente usando a va riável chave como o nome da propriedade e a variável valor como o valor atribuído a ela.

**3.4 Métodos abreviados**

Quando uma função é atribuída a uma propriedade de um objeto, você pode usar a sintaxe de método abreviado para definir o método sem usar a palavra chave function.

Listagem 11: Métodos abreviados

1 const objeto = {

2 nome: 'Maria',

3 dizerOla() {

4 console.log('Ola!');

5 }

6 };

7

8 objeto.dizerOla(); // Saida: Ola!

4