Seção 06: Javascript Objetos e prototypes (avançado)

* Aula 70 Revisando Obetos

const pessoa = {

    nome: 'Luiz',

    sobrenome: 'Otávio'

};

console.log(pessoa['nome']);

console.log(pessoa.sobrenome);

forma de chamar

const pessoa = {

    nome: 'Luiz',

    sobrenome: 'Otávio',

    idade: 30

};

// formas de chamar um atributo

console.log(pessoa['nome']);

console.log(pessoa.sobrenome);

// chave dinamica

const chave = 'idade';

console.log(pessoa[chave]);

console.log('#############################################################');

// new ArraY(); ou []

const pessoa1 = new Object();

pessoa1.nome = 'Luiz';

pessoa1.sobrenome = 'Otávio';

pessoa1.falarNome = function(){

    return `${this.nome} está falando seu nome.`;

};

pessoa1.idade = 30;

pessoa1.getDataNascimento = function() {

    const dataAtual = new Date();

    return dataAtual.getFullYear() - this.idade;

};

const pessoa2 = {

    nome: 'Luiz',

    sobrenome: 'Otávio'

};

// delete pessoa2.nome;

// console.log(pessoa1);

// console.log(pessoa2);

// pessoa1.falarNome()

// console.log(pessoa1.getDataNascimento());

for (let chave in pessoa1) {

    console.log(chave, pessoa1[chave]);

}

console.log('#############################################################');

// factory functions / Constructor functions // Classes

// Factory function

function criaPessoa(nome, sobrenome) {

    return {

        nome,

        sobrenome,

        nomeCompleto() {

            return `${this.nome} ${this.sobrenome}`

        }

    };

};

const p1 = criaPessoa('Luiz', 'Otávio');

console.log(p1.nomeCompleto());

// Constructor functions

function Pessoa(nome, sobrenome) {

    this.nome = nome,

    this.sobrenome = sobrenome;

    Object.freeze(this);  // Nenhum objeto poderar ser alterado seu valor ou criado

}

// {} <- this -> retornando esse this

const pe1 = new Pessoa('Maria', 'Joaquina');

Object.freeze(pe1);   // para travar o objeto

pe1.nome = 'Monica';  // iria aterar o objeto caso não estivesse travado

const pe2 = new Pessoa('João', 'Pedro');

console.log(pe1);

console.log(pe2);

* Aula 71 Object.defineProperty() e Object.defineProperties()

// defineProperty - defineProperties

function Produto(nome, preco, estoque) {

    this.nome = nome;

    this.preco = preco;

    Object.defineProperty(this, 'estoque', {

        enumerable: true,     // mostra a chave

        value: estoque,       // valores que vai receber

        writable: false,      // controla se o valor pode ou não ser alterado

        configurable: false   // se pode reconfigurar a propriedade (ex: delete p1.estoque)

    });

}

const p1 = new Produto('Camiseta', 20, 3);

console.log(p1);

console.log(Object.keys(p1)); [ 'nome', 'preco', 'estoque' ]

para mostrar as chaves do seu objeto

Se a opção de enumerable for false, essa chave não aparecerá.

 Object.defineProperty(this, 'estoque', {

        enumerable: false,     // mostra a chave

    });

console.log(Object.keys(p1)); [ 'nome', 'preco' ]

essa foi a defineProperty, que serve para pegar so um valor.

Agora vamos ver a defineProperties para pegar mais de uma propriedade

function Produto2(nome, preco, estoque) {

    Object.defineProperties(this, {

        nome: {

            enumerable: true,

            value: nome,

            writable: true,

            configurable: true

        },

        preco: {

            enumerable: false,

            value: preco,

            writable: true,

            configurable: true

        },

        estoque: {

            enumerable: true,

            value: estoque,

            writable: true,

            configurable: true

        }

    });

}

const p2 = new Produto2('Camiseta', 20, 3);

console.log(p2);

você vai interando com cada chave do seu objeto particularmente, um a um

* Aula 72 Getters e Setters

// defineProperty -> Getters e Setters

// Get - Obter valor

// Set - setar esse valor

function Produto(nome, preco, estoque) {

    this.nome = nome;

    this.preco = preco;

    let estoquePrivado = estoque

    Object.defineProperty(this, 'estoque', {

        enumerable: true,

        configurable: false,

        get: function() {    // Pegar e mostrar o valor

            return estoquePrivado;

        },

        set: function(valor) {

            if(typeof valor !== 'number') {

                console.log('Bad value');

                return

            }

            estoquePrivado = valor;

        }

    });

}

const p1 = new Produto('Camisa', 20, 3);

//console.log(p1);

p1.estoque = 500;

console.log(p1.estoque);

agora com function factory

// Function Factory get e set

function criaProduto(nome) {

    return {

        get nome(){

            return nome;

        },

        set nome(valor){

            nome = valor;

        }

    };

}

const p2 = criaProduto('camiseta');

console.log(p2.nome);