Desenvolvimento de Sistemas : Encapsulamento

Prof. Ovídio Francisco

Plano de aula

Introdução

Introdução

O contexto

Na POO, encapsulamento significa esconder atributos da classe de forma que somente ela pode acessá-lo. Essa classe pode fornecer métodos que permitem acesso controlado ao atributo.

Esse recurso é utilizado para proteger dados alterações indevidas, mas permitindo acesso por intermédio de métodos.

O contexto

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

p = new Pessoa("Francisco");

p.nome = 123.456;
```

O contexto

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

p = new Pessoa("Francisco");

p.nome = 123.456;
```

Não faz sentido,
mas nada impede

Atributos privados

O contexto

Por padrão, em Javascript, todos os atributos são **públicos**, o que significa que **podem ser acessados** livremente de **fora da classe**, como no exemplo anterior.

Atributos **privados**, por outro lado, **não** pode ser acessados fora da classe. Ou seja, **somente a própria classe** tem domínio sobre um atributo privado.

O contexto

Em Javascript, atributos privados são declarados com um **#** e somente são acessíveis dentro do escopo onde foram declarados.

Tipicamente, **criamos um atributo privado ao declará-lo dentro do escopo da classe**.

Qualquer tentativa de acessá-lo **fora da classe**, resulta em um **erro de sintaxe**;

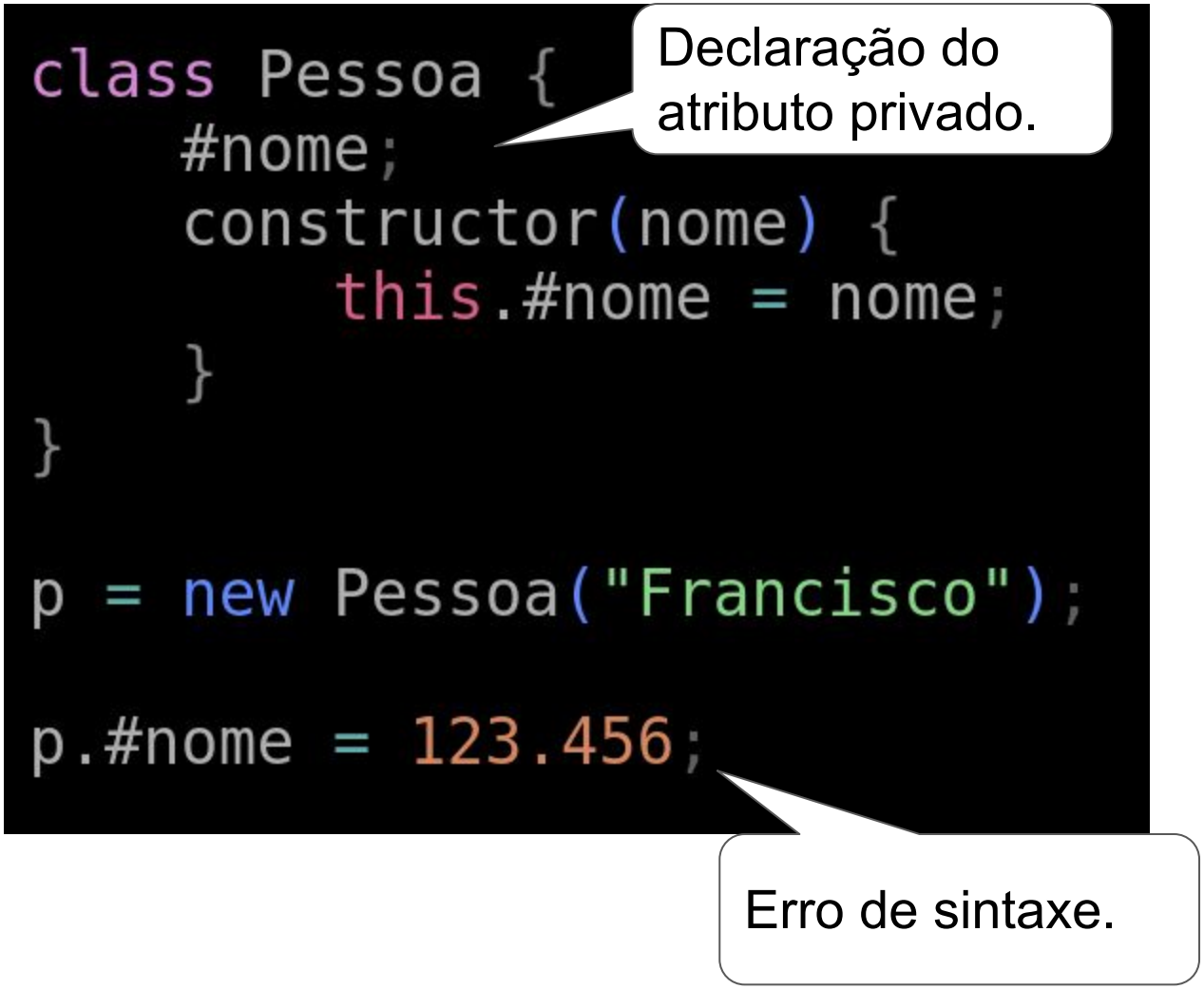
O contexto

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

p = new Pessoa("Francisco");

p.#nome = 123.456;
```

O contexto


class Pessoa {
#nome;
constructor(nome) {
this.#nome = nome;
}
}
p = new Pessoa("Francisco");
p.#nome = 123.456;
Declaração do atributo privado.
Erro de sintaxe.

O contexto

Com isso o atributo fica protegido de acessos externos.

Mas como utilizar esse atributo além de dentro da própria classe?

Métodos Acessores

Metodos acessores

Para fornecer acesso controlado aos atributos encapsulados, podemos criar métodos que intermediam leituras e alterações.

Os métodos que acessam atributos privados, para leitura ou escrita, são comumente chamados de *getters* e *setters*.

O contexto

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

    get nome() {
        return this.#nome;
    }

    set nome(nome) {
        this.#nome = nome
    }
}

p = new Pessoa("Francisco");

p.nome = "Chico";
```

O contexto

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

    get nome() {
        return this.#nome;
    }

    set nome(nome) {
        this.#nome = nome
    }
}

p = new Pessoa("Francis

p.nome = 123.456;
```

Falta resolver o problema de valores indevídos.

O contexto

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

    get nome() {
        return this.#nome;
    }

    set nome(nome) {
        if (typeof nome != 'string')  {
            throw "O nome deve ser uma string";
            return;
        }

        this.#nome = nome.tim();
    }
}

p = new Pessoa("Francisco");

p.nome = " Chico   ";
```

O contexto

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

    get nome() {
        return this.#nome;
    }

    set nome(nome) {
        if (typeof nome != 'string')  {
            throw "O nome deve ser uma string";
            return;
        }

        this.#nome = nome.tim();
    }
}

p = new Pessoa("Francisco");

p.nome = " Chico   ";
```

Validação dos dados

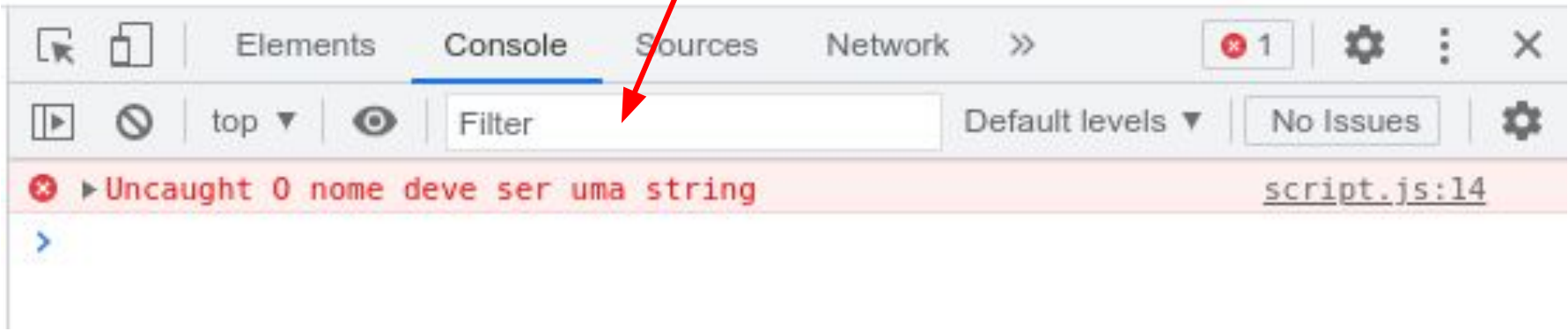
Tratamento (remove espaços extras)

O contexto

As tentativas de alteração com dados inválidos resultarão em um erro.

```
p = new Pessoa("Arnaldo");

p.nome = 678.987;
```



Para saber mais...

- https://itnext.io/controlling-access-to-data-in-javascript-903e80213168
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Classes/Private_class_fields
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_With_Private_Class_Features
- https://stackoverflow.com/questions/55611/javascript-private-methods
- https://stackify.com/oop-concept-for-beginners-what-is-encapsulation/
- https://www.w3schools.com/js/js_object_accessors.asp
- https://bobbyhadz.com/blog/javascript-check-if-variable-is-string#:~:text=Use%20the%20typeof%20operator%20to,the%20typeof%20of%20a%20value.
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/throw

Exercícios

Exercícios

Crie um classe Filme que contém os atributos título, duração em minutos e gênero. Essa classe deve encapsular e validar todos os atributos.

O título não pode estar vazio.

A duração deve ser maior que zero.

O gênero de ser Romance, Terror ou Comédia;

Exercícios

Crie uma classe conta bancária da qual o atributo saldo é encapsulado e seu valor será acessado por meio dos métodos, depósito, saque e consulta.

Para cada depósito deve ser cobrado uma taxa de 1%.

Para cada saque deve ser cobrado uma taxa de 0,5%.

A cada 5 consultas, será cobrado uma taxa de 0,10 centavos.