

Classes

Parte 01

Professor Arthur Giangiarulo

POO

- Paradigma de Programação que busca representação através do nosso mundo, dia a dia, objetos físicos.

Nome

CPF

Paradigma:

Significa modelo ou padrão, correspondendo a algo que vai servir de modelo ou exemplo a ser seguido em determinada situação. São as normas orientadoras de um grupo que estabelecem limites e que determinam como um indivíduo deve agir dentro desses limites.



POO

- Reusabilidade de código
- Facilidade na Manutenção
- Maior integridade
- Maior segurança

Nome

CPF

POO



Nome

CPF

POO

Nome

CPF

Nome
CPF



POO

Nome



CPF



Nome
CPF



Nome
CPF

POO

Nome



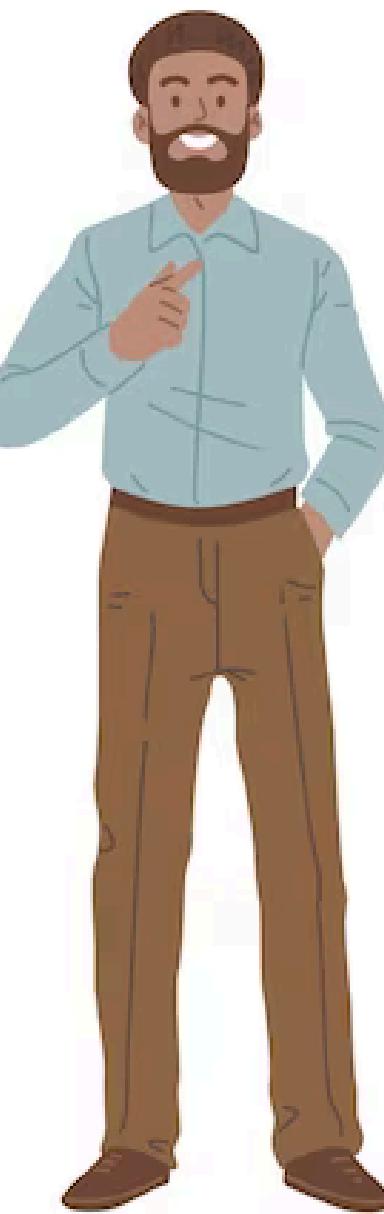
CPF



Nome
CPF



Nome
CPF



POO

Nome

CPF

Nome
CPF

Nome
CPF

Nome
CPF

Nome
CPF



POO

Objeto: Pessoa
Nome

CPF

Nome
CPF

Nome
CPF

Nome
CPF

Nome
CPF



O que é POO

- Abstração: capacidade de focar nas características essenciais de um objeto.
- Ocultação: ocultar detalhes internos e mostrar apenas o necessário.
- Modularidade: separação do código em blocos distintos e independentes.
- Polimorfismo: um objeto pode assumir muitas formas.
- Herança: reutilização de código e formação de hierarquias.

História

- Programação de baixo nível:
Linguagem de máquina,
assembly.
- Programação Procedural:
Programação mais abstrata
focada em funções e
procedimentos. Modularização.



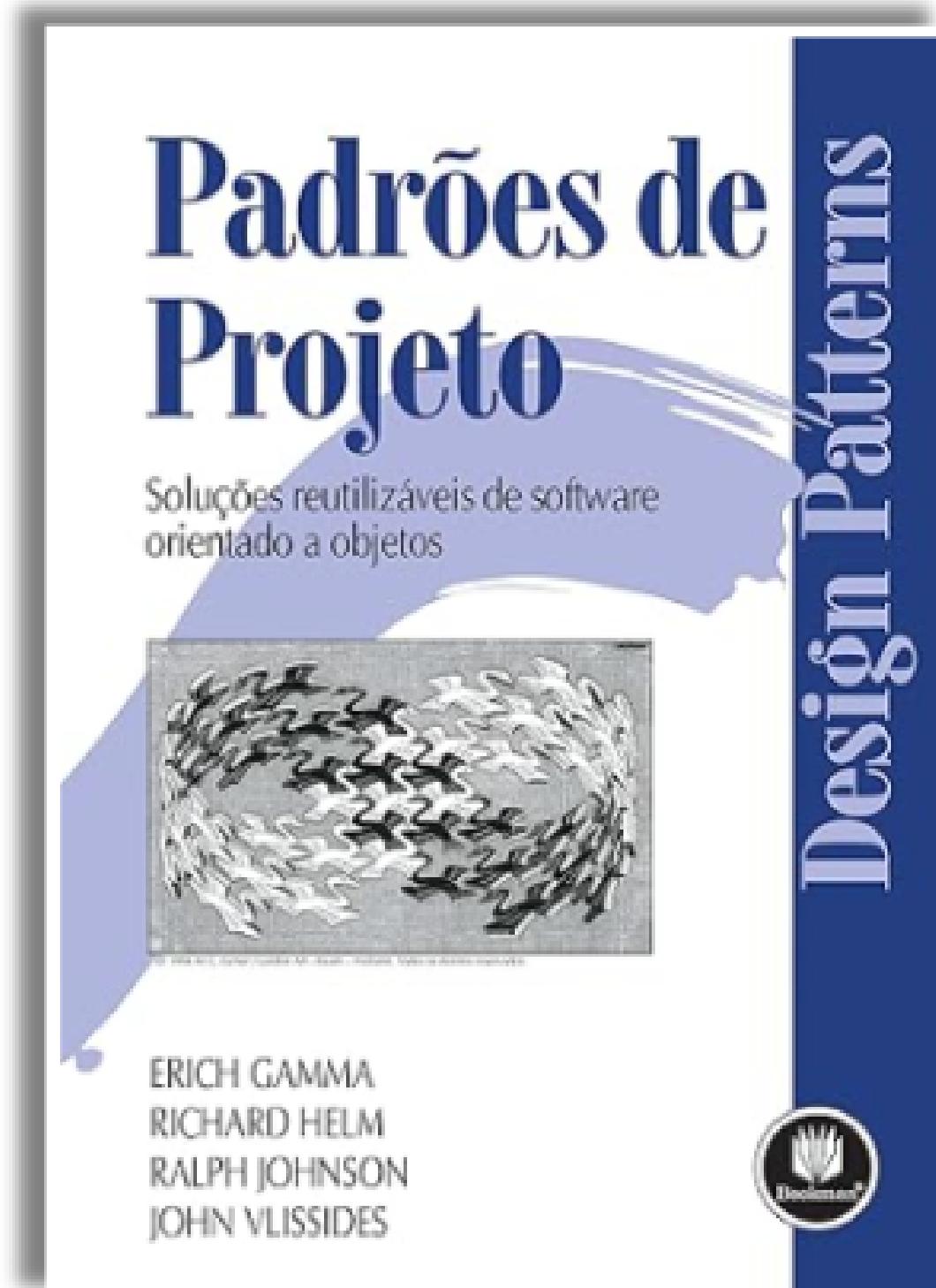
Motivações por trás da POO

- Mapeamento Natural: O mundo é composto por objetos que interagem entre si. A POO permite que programadores modelizem e simulem esse comportamento de maneira intuitiva.
- Abstração e Modularidade: Isolamento das partes do software, facilitando a manutenção, testes e reusabilidade.
- Desenvolvimento Incremental: Permitindo a expansão e evolução dos sistemas sem a necessidade de grandes reestruturações.

Impacto na Indústria

- Frameworks e Bibliotecas: O OOP facilitou a criação de bibliotecas e frameworks reutilizáveis, como o .NET, Spring e muitos outros.
- Padrões de Projeto: A POO levou ao desenvolvimento de padrões de projeto que resolvem problemas comuns de design de software, promovidos pelo famoso livro "Design Patterns" de Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides (o "Gang of Four").

Impacto na Indústria

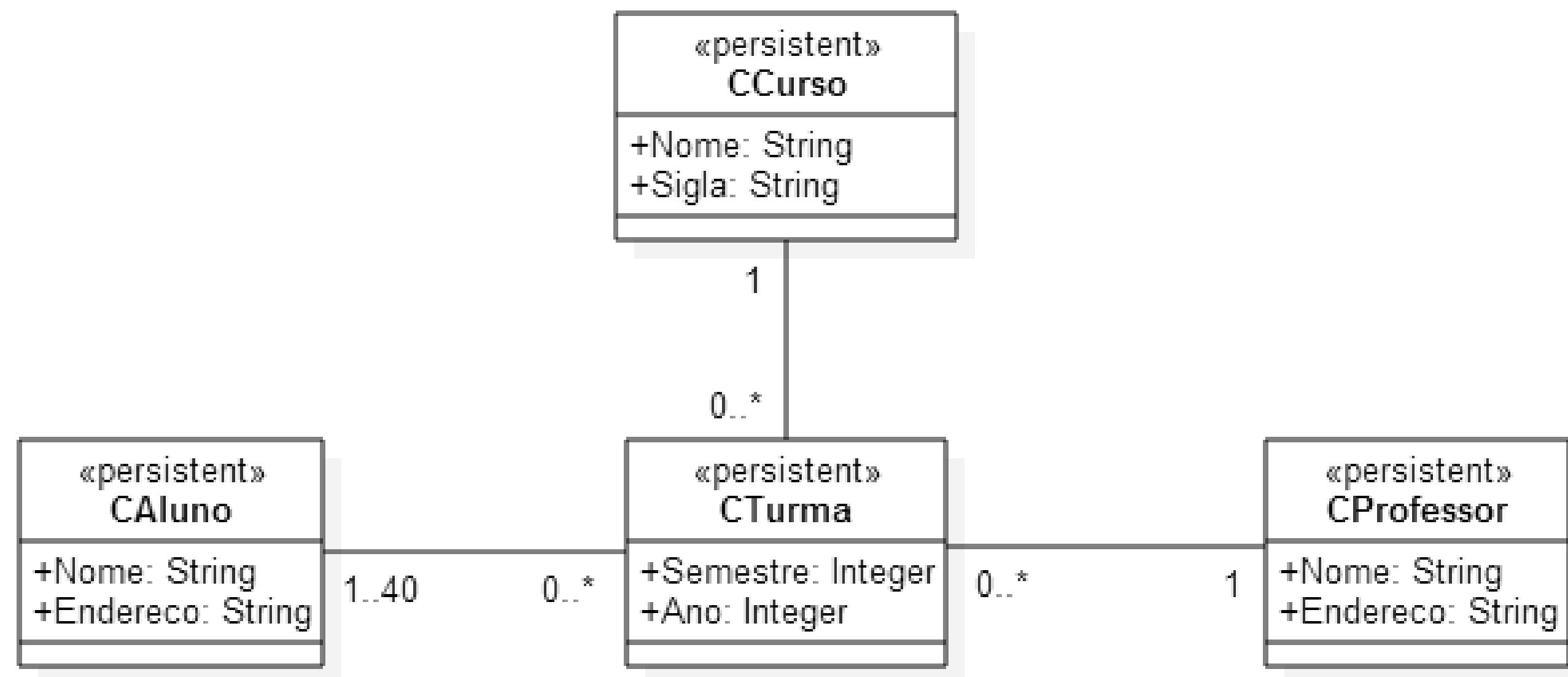


Simulação do Mundo Real

```
public class Carro {  
    private String modelo;  
    private int ano;  
    private String fabricante;  
    private String proprietario;  
    private double valor;  
    private String placa;
```



Introdução aos Conceitos Básicos

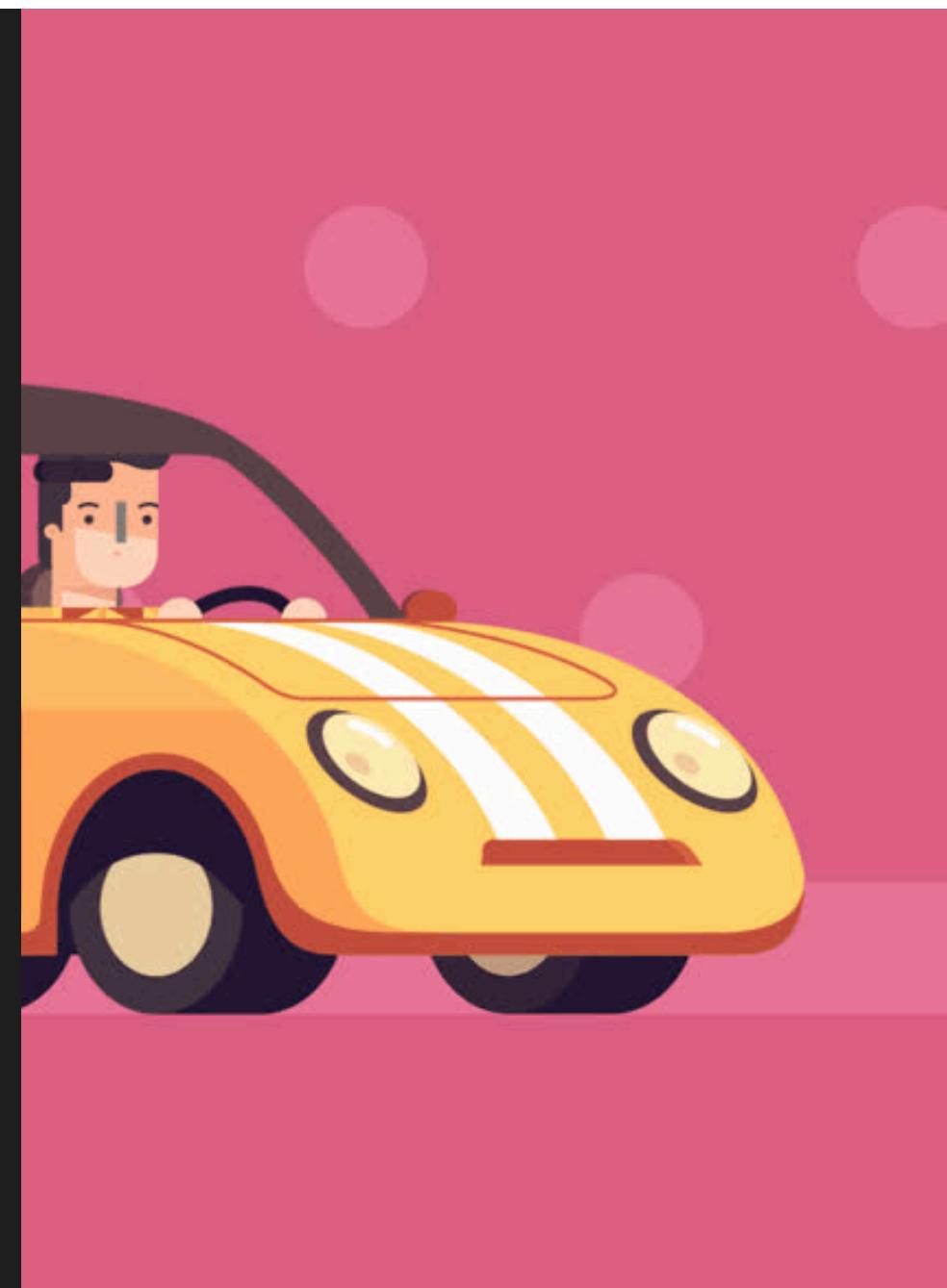


Introdução aos Conceitos Básicos

- Classe: Como uma receita que define os atributos e métodos de um objeto.
- Objeto: Uma instância de uma classe. É a realização concreta da "classe".
- Atributos: Características ou propriedades de um objeto. Eles definem o que o objeto "é" ou "tem".
- Métodos: Funções ou procedimentos associados a um objeto. Eles definem o que o objeto pode "fazer".

Simulação do Mundo Real

```
public class Exemplos {  
  
    class Carro {  
  
        // Atributos  
        private String modelo;  
        private int ano;  
  
        // Métodos  
        public void acelerar() {  
            System.out.println("O carro " + modelo + " está acelerando.");  
        }  
    }  
}
```



Métodos x Funções

Funções

- Em Java, o termo "função" não é comumente usado no contexto de desenvolvimento orientado a objetos, já que tudo é associado a classes. No entanto, podemos nos referir aos métodos static em classes como o equivalente mais próximo de funções, já que eles pertencem à classe e não a uma instância específica da classe.
- Eles não operam em instâncias específicas e não têm acesso a membros não estáticos da classe.

Métodos x Funções

Métodos

- É uma função associada a uma instância de uma classe.
- Em Java, métodos são sempre definidos dentro de classes.
- Eles podem operar sobre os atributos (ou membros) do objeto.

Métodos x Funções

```
public class Exemplo {  
    private int valor;  
  
    public Exemplo(int valor) {  
        this.valor = valor;  
    }  
  
    public int dobrarValor() {  
        return this.valor * 2;  
    }  
}
```

```
public class Utilidade {  
    public static int dobrar(int valor) {  
        return valor * 2;  
    }  
}
```

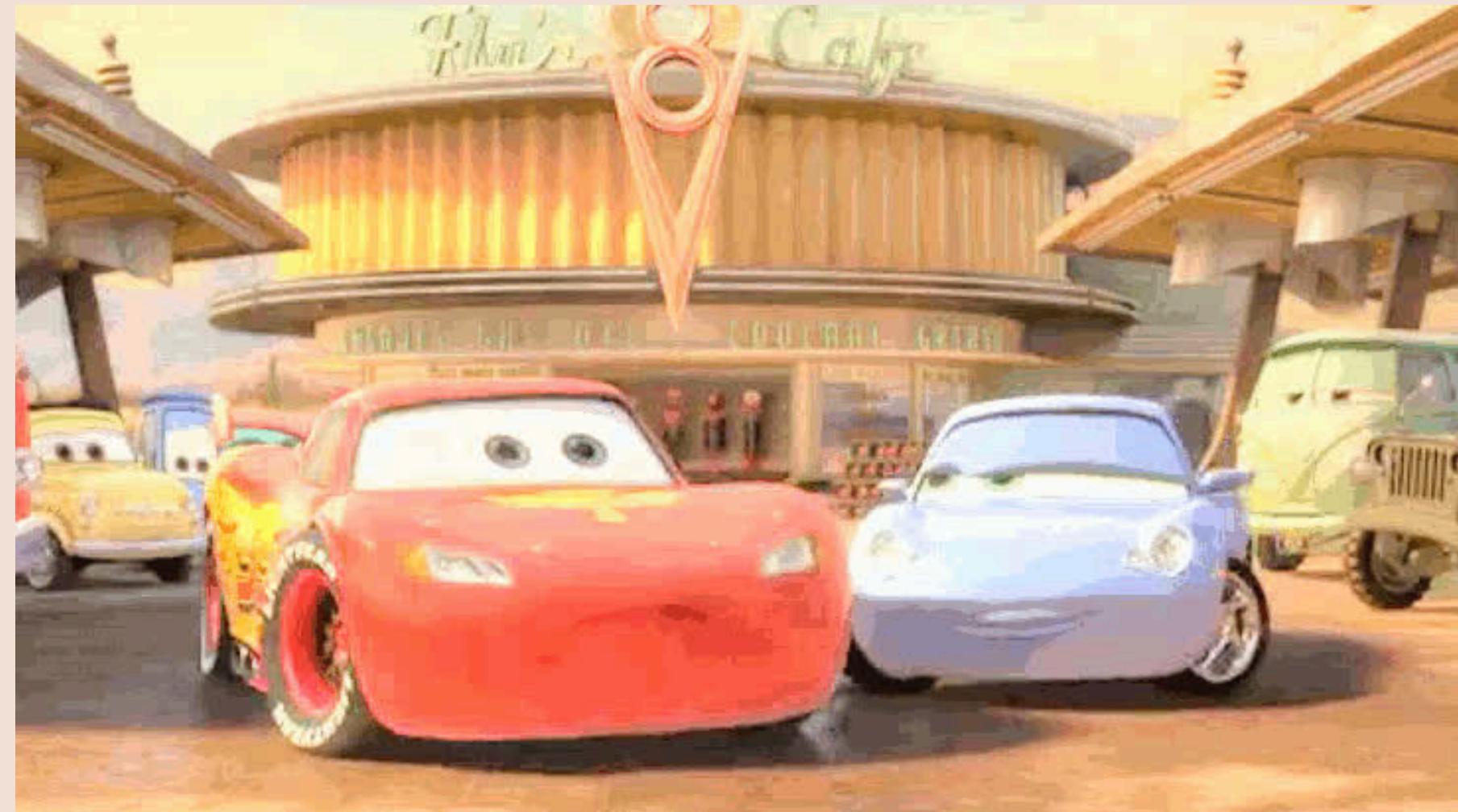
Classes

- Em Programação Orientada a Objetos, a classe é um dos principais pilares.
- Ela serve como um molde para criar objetos. Pode-se pensar em uma classe como um esboço que descreve os detalhes técnicos do que o objeto deve ser, mas sem ser o objeto em si.
- Em Java, uma classe não só define atributos (dados), mas também comportamentos (métodos).



Objeto

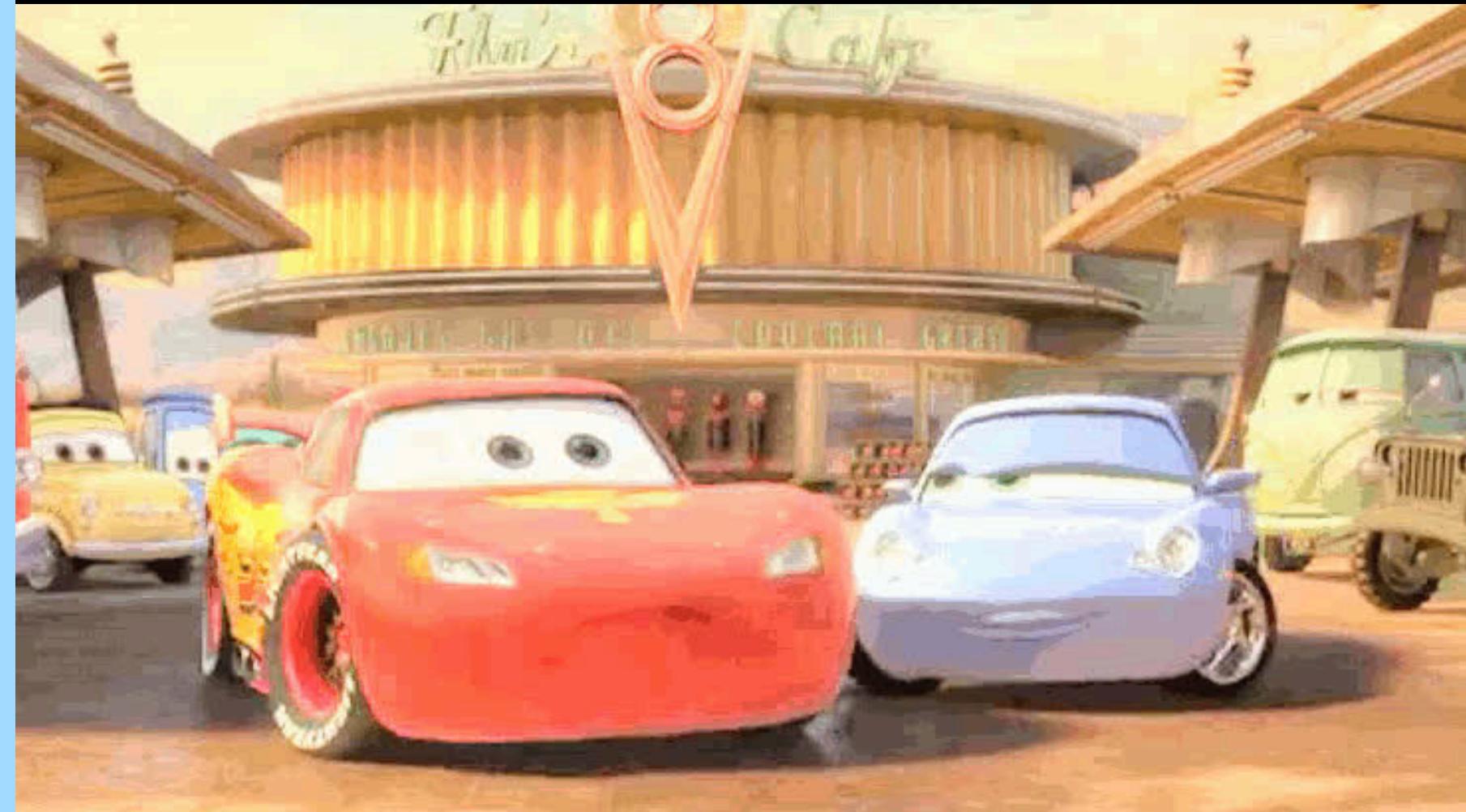
- objeto é a realização física e concreta de uma classe. Se a classe é o esboço, o objeto é a casa construída com base nesse esboço.
- Um mesmo molde (classe) pode ser usado para criar várias instâncias (objetos).



Atributos

- Também conhecidos como propriedades ou campos, são as variáveis definidas dentro de uma classe.
- Representam o estado de um objeto. No exemplo de um Carro, o estado pode incluir coisas como cor, modelo, ano, entre outros.

```
private String cor;  
private String modelo;  
private int ano;
```



Métodos

```
public double calcularIPVA() {  
    return this.valor * 0.04; // 4% do valor do carro  
}
```

- Métodos são essencialmente funções definidas dentro de uma classe e representam as ações que um objeto pode executar.
- Eles operam nos dados de um objeto e podem modificar o estado de um objeto.

Vamos Praticar!



Exercícios

1) Crie uma classe Pessoa com os atributos nome, idade, CPF e data de nascimento. Adicione métodos para exibir os detalhes da pessoa e para alterar cada um dos atributos (incluindo os devidos tratamentos de formatos). Crie um objeto da classe e teste os métodos.

Nome
CPF

Nome
CPF



Exercícios

2) Crie uma classe ContaBancaria com os atributos numeroDaConta, saldo e titular. Adicione métodos para depositar e sacar dinheiro, um método para exibir o saldo e outro para exibir o extrato. Implemente a lógica necessária para evitar saques que deixem o saldo negativo. Crie um objeto e teste os métodos.

