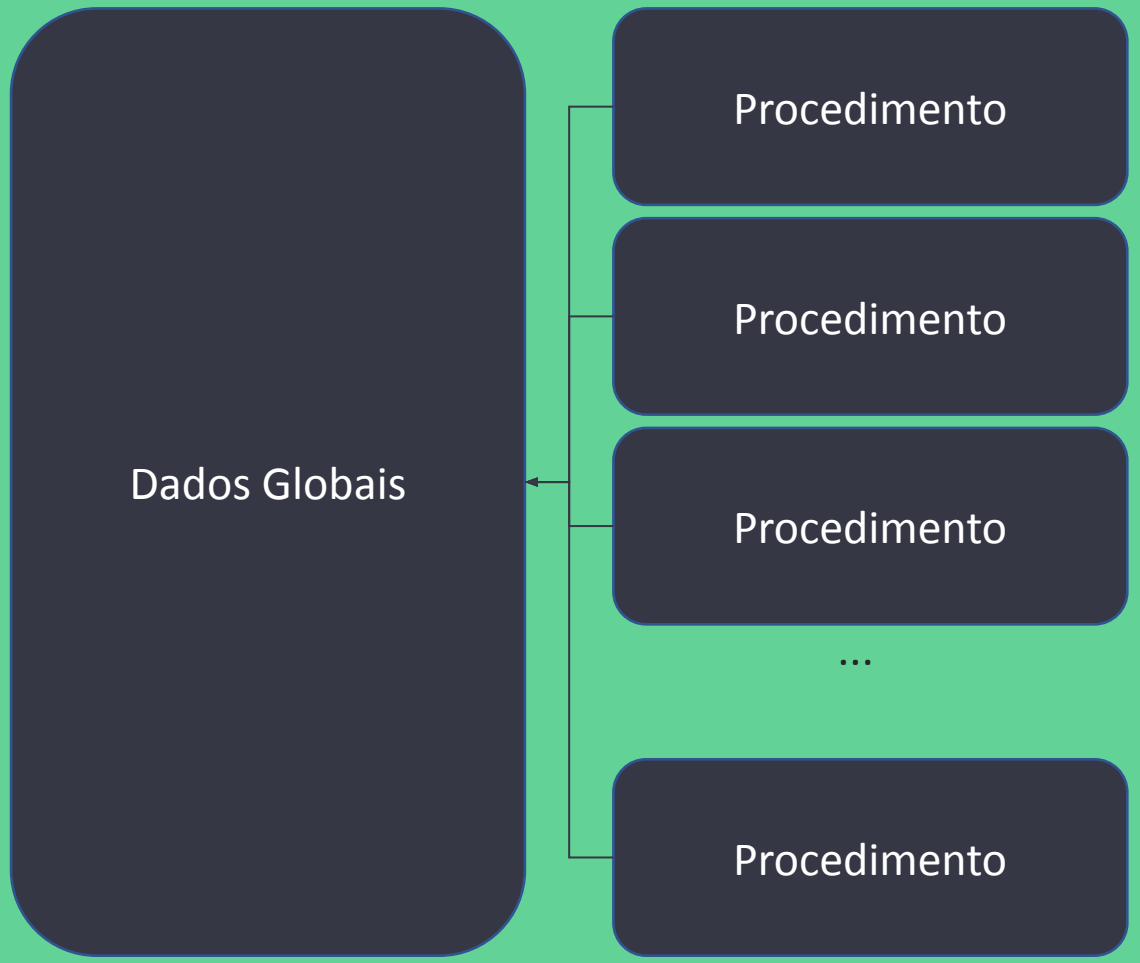
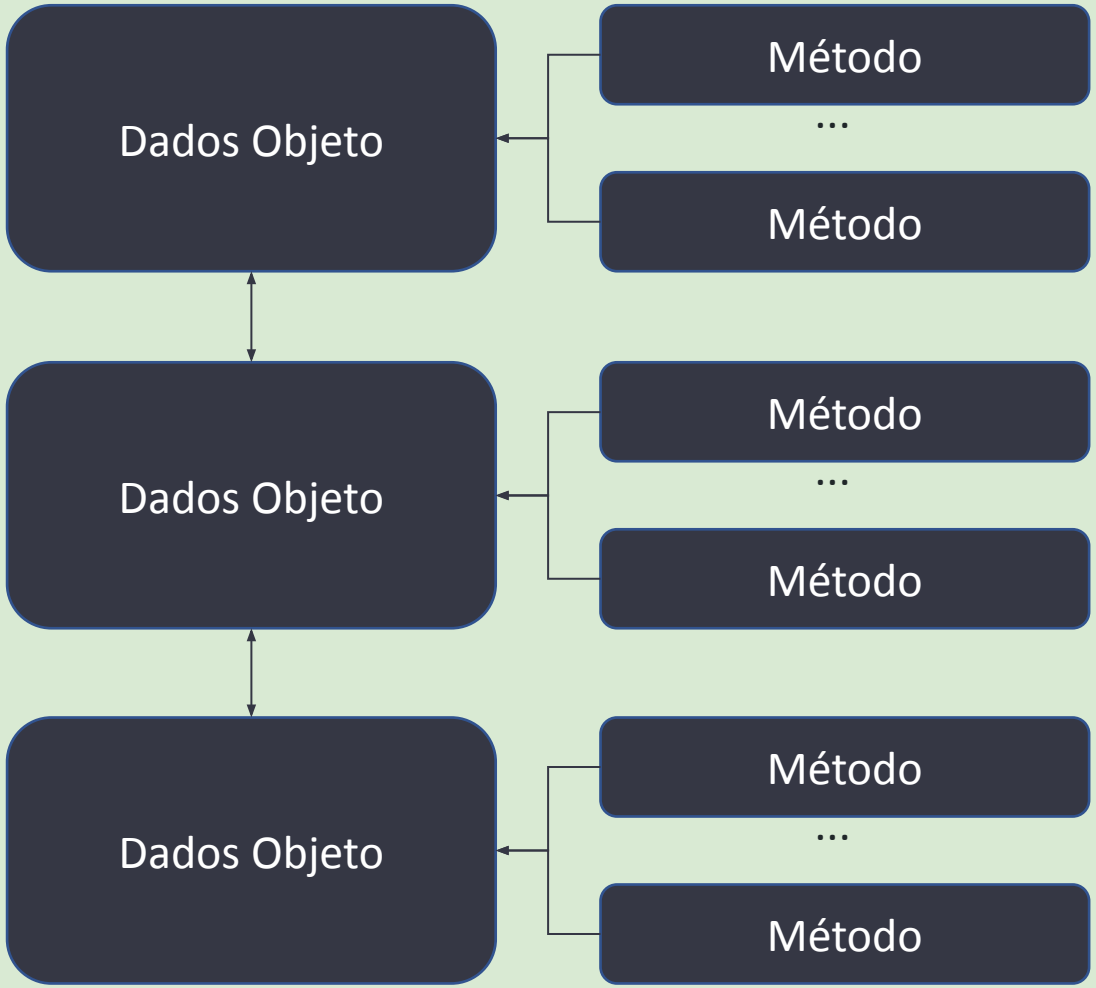
Programação **O**rientada a **O**bjetos

Carlito Alves
Fernando Simões
Gustavo Spindola
João Vitor
Pedro Rabis
Victor Ferrari

Thunder Hawks

Programação Estruturada





Programação Orientada a Objetos

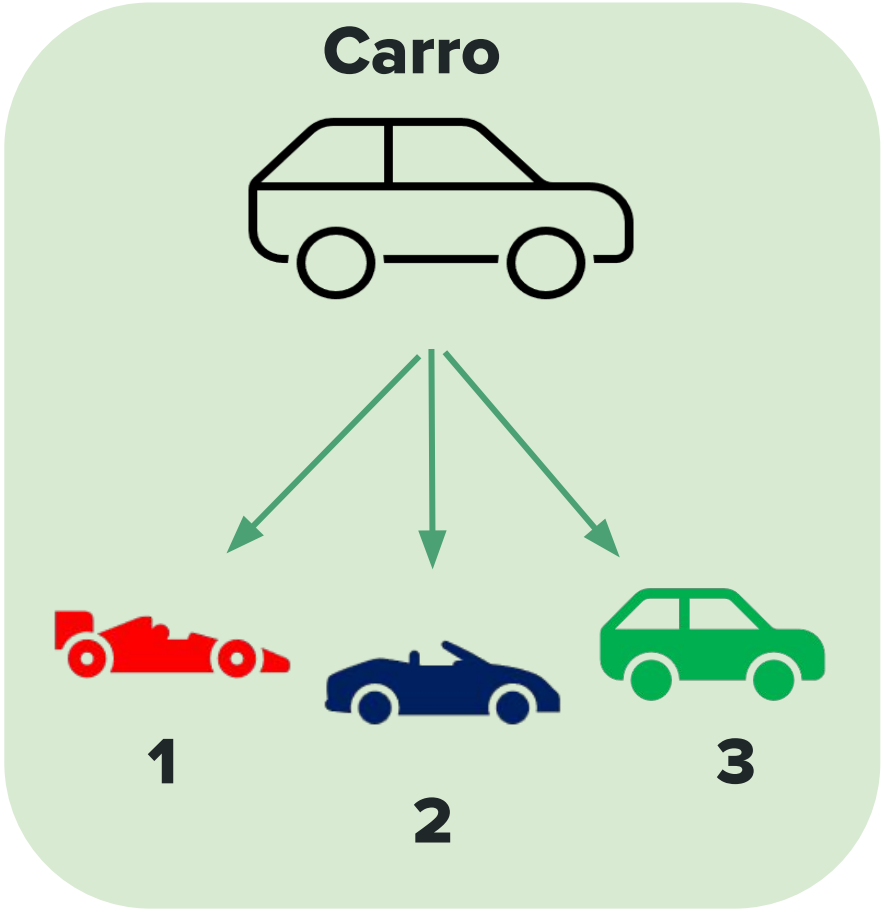
Classe e Objetos

CLASSE:

- Representa ideia ou conceito e classifica objetos que tenham propriedades similares.
- Possuem responsabilidades bem definidas.
- Tipo personalizado de dados.
- "Molde" para a criação de objetos.

OBJETO:

- Ocorrência específica de uma classe (instância de classe).
- Representam entidades do mundo real.
- Tem características próprias (atributos) e executa ações (métodos) provenientes da classe que originou o objeto.



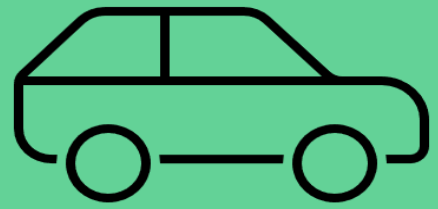
Atributos e Métodos

Atributos:

- Características que descrevem um objeto.
- “Campos” que armazenam diferentes valores que o objeto pode conter

Métodos:

- Lógica contida em uma CLASSE para atribuir comportamentos (sequência de comandos), identificada por um nome.
- Similar a funções e procedimentos.
- Ato de invocar (chamar) um método é a passagem de mensagens para o Objeto



Carro

```
Marca
Modelo
Cor
numeroPassageiros
capacidadeCombustivel
consumoCombustivel


Acelerar()
Frear()
Ir para frente()
Ré()
Ligar()
Desligar()
```

P A

POLIMORFISM

- Uma operação comportamento dependendo lvendo respostas distintas.
- Ocorre quando nte para a mesma ação.

HERANÇ

- Relaciona e métodos de outra
- Assim, po go.

```
    }

    public void acelerar() {
        System.out.printf("Acelerando...\n");
    }

    public void frear() {
        System.out.printf("Freando...\n");
    }

    public void buzinar() {
        System.out.printf("BI BI!!!\n");
    }
```

```
public class CarroConversivel extends Carro {

    @Override
    public void buzinar(){
        System.out.printf("PAN PAN NA NAM PAN PAN!!!\n");
    }
}
```

ENCAPSULAMENTO

ENCAPSULAMENT

- Combinação de atr xando visível apenas o necessário para a co talhes da implementação ou a lógica de um me
- Permite ocultar a cor
- Não é necessário e se para poder utilizar os seus métodos.

```
}

public String getCor() {
    return cor;
}

public void setCor(String cor) {
    this.cor = cor;
}

public String getMarca() {
    return marca;
}

public void setMarca(String marca) {
    this.marca = marca;
}

public String getModelo() {
    return modelo;
}
```

INTERFACE

- Quando duas (ou mais) classes possuem comportamentos comuns que podem ser separados em uma outra classe, dizemos que a "classe comum" é uma interface, que pode ser "herdada" pelas outras classes.
- Uma interface não é exatamente um classe, mas sim um conjunto de métodos que todas as classes que herdam dela devem possuir
- No entanto, uma interface pode herdar de outra interface, criando uma hierarquia de interfaces.

```
interface VeiculoTerrestre {
    void acelerar();
    void buzinar();
    void frear();
}
```

CLASSES ABSTRATAS

- Uma classe abstrata é uma classe que serve de modelo para outras classes. Ela sempre será uma superclasse genérica, e suas subclasses serão mais específicas. Além disso, ela não pode ser instanciada e pode conter ou não métodos abstratos, podendo ser implementados nas classes descendentes.

- Por exemplo ao ouvir “meio de transporte” podemos imaginar um carro, um barco, etc. Ou seja, uma palavra genérica pode nos levar a imaginar coisas mais específicas.