**Alunos:**

**Brenda R. Alves**

**Kayque C. D. Silva**

**Vitória N. Pereira**

**PROGRAMAÇÃO EM BACK END**

**MOTOS**

**LIMEIRA – SP  
2025**

Sumário

[1. INTRODUÇÃO 1](#_Toc209192486)

[2. ETAPAS DE DESENVOLVIMENTO DA ATIVIDADE EM GRUPO 1](#_Toc209192487)

[3. ESTRUTURA DO CÓDIGO 2](#_Toc209192488)

[**3.1.** **CLASSE MOTO** 2](#_Toc209192489)

[**3.2.** **ATRIBUTOS PRIVADOS** 3](#_Toc209192490)

[**3.3.** **CONSTRUTOR** 3](#_Toc209192491)

[**3.4.** **Métodos** 3](#_Toc209192492)

[**3.5.** **Get e Set** 5](#_Toc209192493)

[**3.6.** **Objetos** 6](#_Toc209192494)

[**3.7.** **Instanciação dos objetos** 7](#_Toc209192495)

[4. Erros no Códgio 8](#_Toc209192496)

# **INTRODUÇÃO**

Este projeto foi proposto pelo professor Samuel Costa durante a aula de Programação de Back-End (PBE), realizada na sexta-feira, dia 19 de setembro de 2025. A atividade foi estruturada com base na formação de grupos de trabalho, nos quais os próprios integrantes ficaram responsáveis pela definição dos temas a serem desenvolvidos. A escolha temática seguiu critérios de afinidade técnica e interesse individual, promovendo uma abordagem colaborativa e alinhada com os conteúdos abordados na disciplina.

# **ETAPAS DE DESENVOLVIMENTO DA ATIVIDADE EM GRUPO**

A atividade em grupo foi estruturada em três fases principais: **escolha do tema, organização das tarefas e execução prática**, seguindo uma abordagem colaborativa e orientada:

1. **Escolha do Tema:** Cada grupo definiu seu próprio tema e utilizando os ensinamentos aprendidos em aula
2. **Organização das Tarefas:** As atividades foram distribuídas conforme as habilidades dos integrantes para garantir eficiência e colaboração.
3. **Execução:** O desenvolvimento seguiu boas práticas de codificação, controle de versão e testes, priorizando a qualidade técnica, funcionalidade da aplicação e trabalho em equipe.\

# **ESTRUTURA DO CÓDIGO**

O código está dividido nas seguintes partes principais:

* 1. Declaração da classe Moto
  2. Atributos (propriedades privadas)
  3. Método construtor \_\_construct
  4. Métodos de comportamento (Mover, Combustivel, exibirInfo)
  5. Métodos get e set (encapsulamento)
  6. Instanciação dos objetos (motos)
  7. Exibição das informações no terminal

## **CLASSE MOTO**



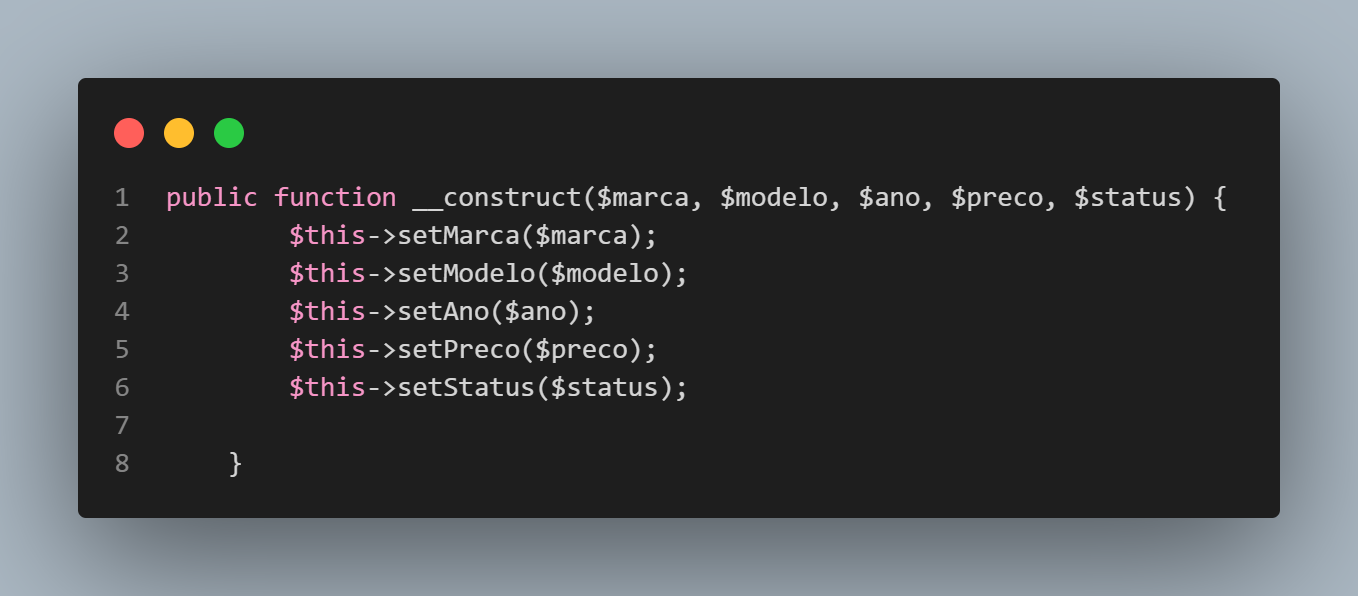
A classe Moto representa o modelo de um objeto com as características básicas de uma motocicleta.

## **ATRIBUTOS PRIVADOS**



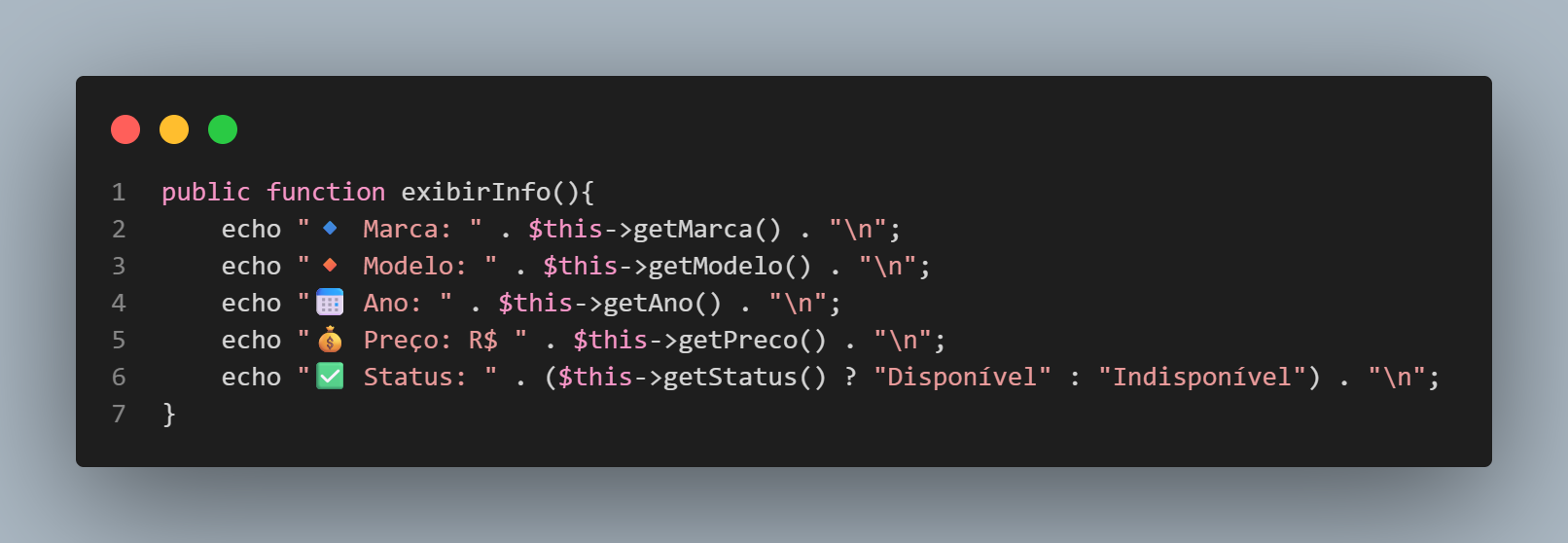
Esses atributos representam os dados que descrevem cada moto. Eles são **privados** (private) para respeitar o **princípio do encapsulamento**, ou seja, só podem ser acessados de forma segura por meio dos métodos get e set.

## **CONSTRUTOR**

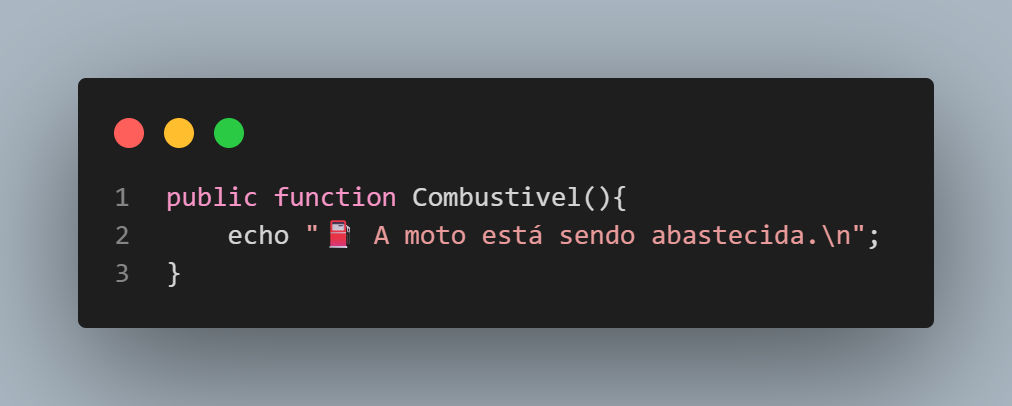
****

O método \_\_construct é chamado automaticamente ao criar um objeto da classe Moto. Ele define os valores iniciais dos atributos utilizando os métodos set.

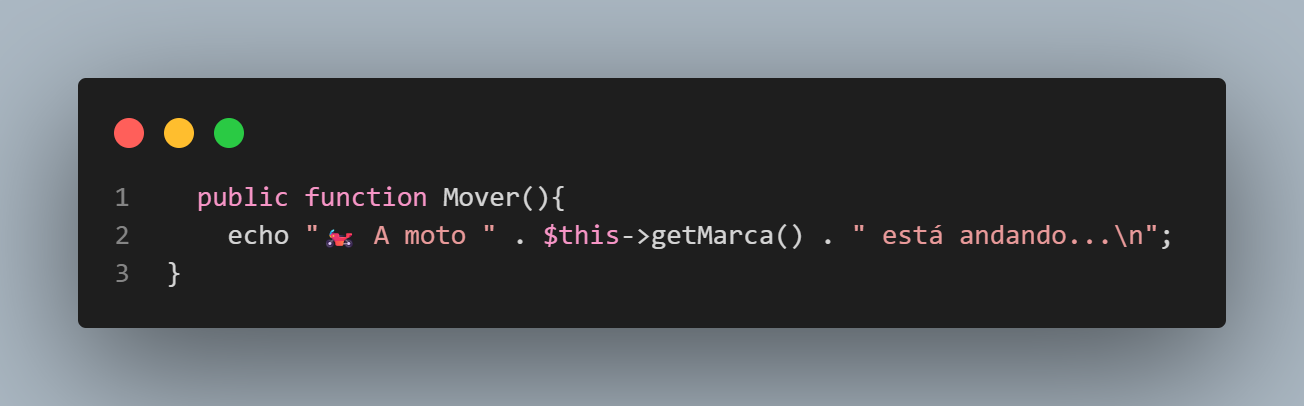
## **Métodos**



Exibe todas as informações da moto formatadas.

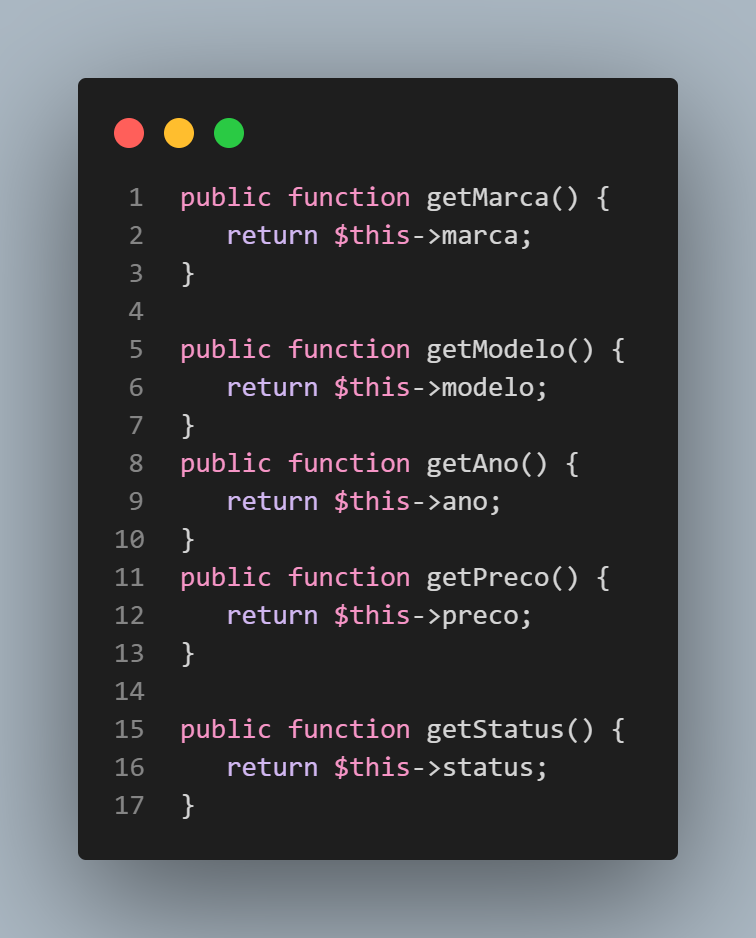


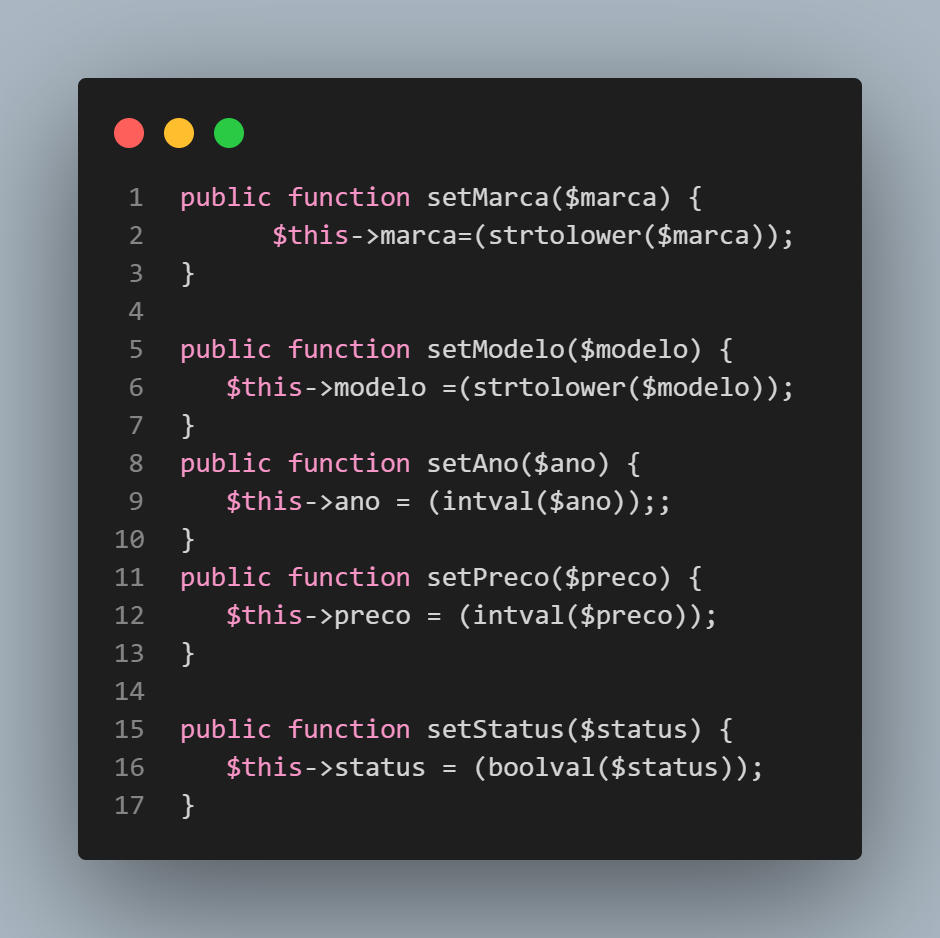
Simula o abastecimento da moto.



Simula o movimento da moto.

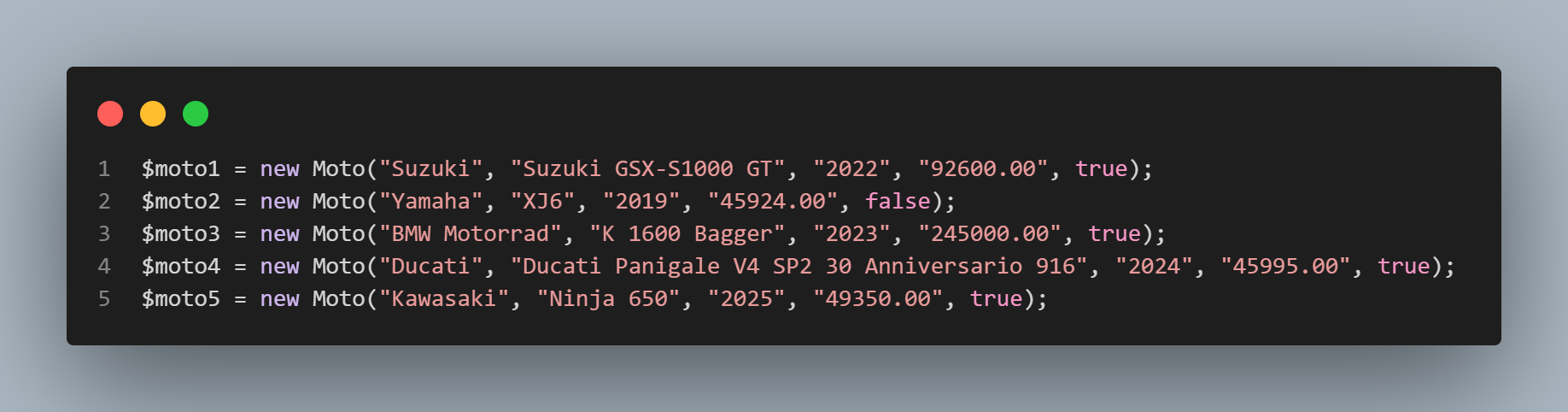
## **Get e Set**





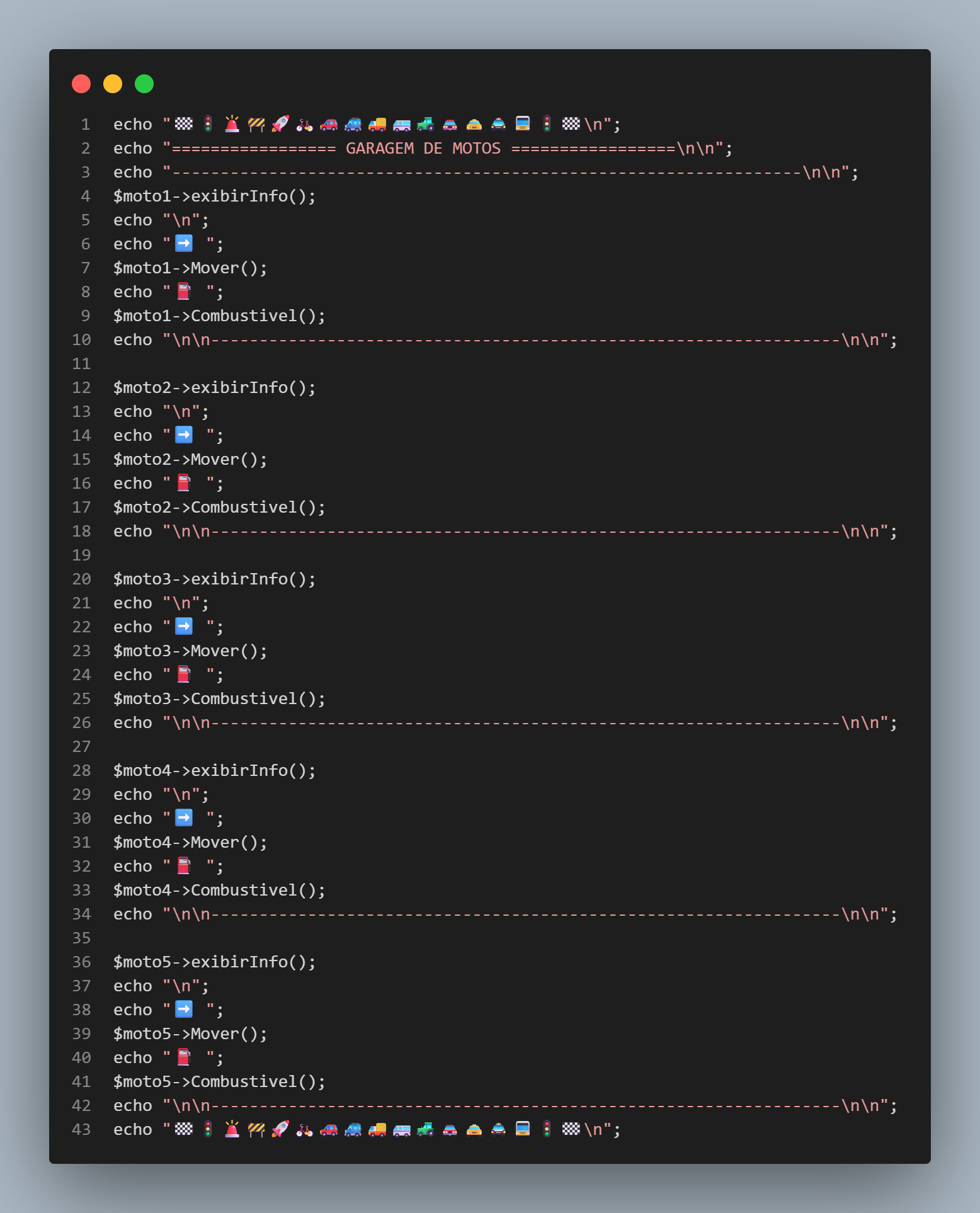
* O método setMarca usa strtolower() para deixar todas as marcas em letras minúsculas.
* O método setAno usa intval() para garantir que o ano seja um número inteiro.
* O método setStatus usa boolval() para garantir que o status seja verdadeiro (true) ou falso (false).

## **Objetos**



Cinco objetos da classe Moto são criados, cada um com dados diferentes, representando diferentes marcas e modelos de motocicletas.

## **Instanciação dos objetos**



A aplicação imprime uma "garagem" com todas as informações das motos e os métodos Mover e Combustivel são chamados para cada uma delas.

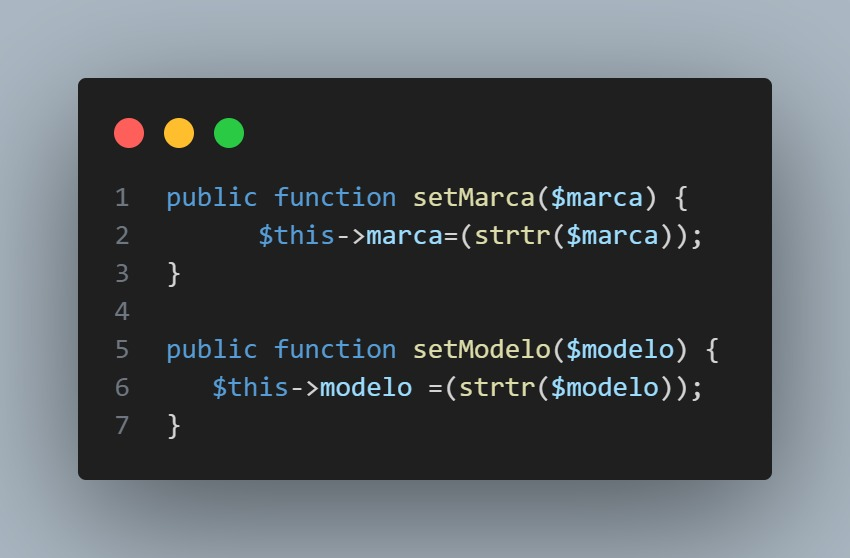
# **Erros no Código**



Funções "Get" (get...) servem para obter ou ler o valor de uma propriedade. Elas não foram feitas para definir ou alterar valores.

Funções "Set" (set...) servem para definir ou atribuir um valor a uma propriedade.

No seu construtor, você está chamando os métodos "get" e passando os novos valores para eles. Isso não tem efeito. Você está basicamente dizendo "pegue o valor da marca e... ignore este novo valor que estou te passando". Como as propriedades ainda não foram inicializadas, elas permanecem com o valor padrão (geralmente null).



O objetivo dentro de um método set (um "setter") é frequentemente validar e limpar os dados antes de armazená-los. A função strtr foi usada de forma errada e provavelmente não era o que o programador queria fazer.

Já a função strtolower é a ferramenta perfeita para padronizar o "case" (maiúsculas/minúsculas) de textos, o que é uma prática excelente para campos como marca e modelo.



Embora o uso da vírgula (,) com o echo seja um pouco mais rápido (uma diferença de microssegundos, imperceptível na prática), a recomendação para programadores, especialmente iniciantes, é:

Use sempre o ponto (.) para juntar strings.