Polimorfismo

Impresso por: Matheus Ferreira Santos

terça-feira, 15 ago. 2023, 18:19

Data:

Site: <u>HackaTruck MakerSpace</u>

Conceitos e Fundamentos: Algoritmos e Programação

Curso: Orientada a Objetos com Swift

Livro: Polimorfismo

Índice

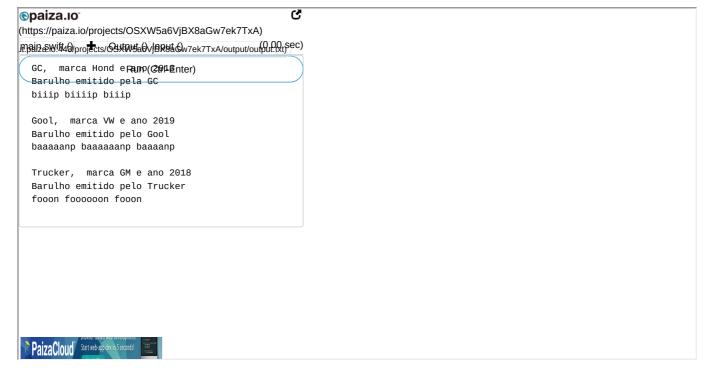
1. Polimorfismo

1. Polimorfismo



A definição de polimorfismo vem da junção de duas palavras e significa "muitas formas" (poli=muitas, morphos=formas). Em orientação a objetos, este conceito permite que classes mais especializadas possam ser tratadas de acordo com sua origem mais genérica, viabilizando um tratamento mais homogêneo. Em outras palavras e exemplificando, conseguimos tratar as classes de acordo com sua natureza, por meio de métodos de mesma assinatura (tipo de retorno, número, ordem e tipo dos parâmetros) e comportamentos diferente, iremos pra um cenário em que temos o método tocarmusica() em um toca disco e o método tocarmusica() no seu novo aparelho de som que tem bluetooth e USB, a forma da mídia é diferente, logo seu processamento também será, mas ambos irão reproduzir os sons. Dizemos que o método "tocar música" é uma forma de **polimorfismo**, pois dois objetos, de duas classes diferentes, têm a mesma assinatura de método que são implementados cada um a sua forma, mas que possuem o mesmo efeito.

Vamos exemplificar:



O polimorfismo permite o envio de uma mesma mensagem para diversos objetos, e que cada um responda da maneira mais apropriada para sua classe. Por exemplo: temos o método buzinar na classe veículo, mas podemos reescrevê-lo nas subclasses, com as suas devidas especificações, a partir do uso da palavra reservada *override*.

Portanto, o polimorfismo permite que classes herdadas tenham um comportamento similar à classe mãe, porém viabilizando certas especializações. Assim, mantém-se uma consistência entre a hierarquia das classes ao mesmo tempo em que se permite graus de especialização. Além disso, o polimorfismo permite maior variabilidade da definição de métodos de uma classe, pois possibilita a definição de múltiplas versões de um mesmo método, desde que cada uma delas receba uma lista de parâmetros distinta.

Detalhando melhor os conceitos, temos:

- Redefinição de métodos: ocorre quando uma classe filha redefine o comportamento de um método da classe mãe. Este tipo de polimorfismo também é chamado de sobrescrita de métodos (ou *overriding*). Na redefinição de métodos, um método é mantido com a mesma assinatura, mas sua implementação é alterada nas classes filhas.
- Sobrecarga de métodos: também conhecido como *overloading*, este conceito trata de múltiplas definições de um método numa mesma classe. A diferenciação se dá por meio de assinaturas distintas para cada "versão" do método, ou seja, apesar dos métodos possuírem o mesmo nome, cada versão deles recebe uma lista de parâmetros distinta. O compilador não identifica conflito, pois ele determina qual método está sendo invocado por meio da lista de parâmetros recebida (assinatura).

Crie as classes Animal, Cao, Gato; faça as devidas associações; crie duas propriedades, sendo elas nome e raça;

Vamos treinar?

e use os conceitos de Overloading e Overriding para criar os metodos emitirSom(), comer(), escalar() e babar ().			
E	Exclusivamente para o método comer(), receba a comida via parâmetro.		
Г			

Resolução:

PaizaCloud Start web-app dev in 5 seconds!



Outras dúvidas ou sugestões entre em contato com contato@hackatruck.com.br.

Faça já os exercícios desde capítulo.

Exercícios – Polimorfismo 🔼