Programação Orientada a Objetos - Parte 2

Prof. Wesin Ribeiro

Neste capítulo

- Associação
- Herança
- Polimorfismo
- Classes Abstratas

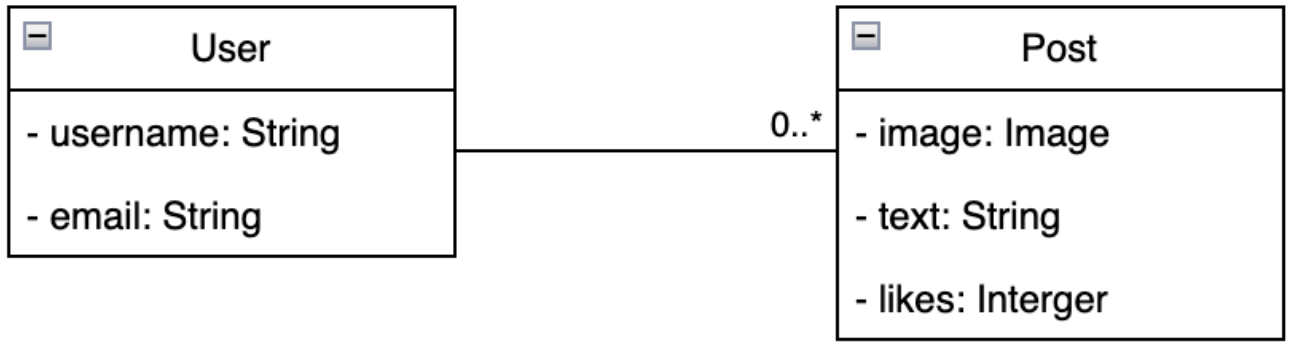
Em um cenário real, sempre irão existir várias entidades que serão encapsuladas em classes e objetos. Inevitavelmente, uma classe poderá estar ligada a outra classe através de um relacionamento.

Associação

Uma associação descer um vínculo que ocorre normalmente entre objetos de uma ou mais classes

Tipos de associação

- Unária
- Binária
- Ternária
- Agregação
  - Representa a ideia de complemento
- Composição
  - Ideia de Todo/Parte



Exemplo de associação binária (a mais comum)

Multiplicidade nos relacionamentos

Multiplicidade	Significado
0..1	No mínimo 0 e no máximo 1. Indica que os objetos das classes associadas não precisam obrigatoriamente estar relacionados, mas se houver relacionamento indica que apenas uma instância da classe se relaciona com as instâncias da outra classe.
1..1	Um e somente um. Indica que apenas um objeto se relaciona com os objetos da outra classe
0..*	No mínimo zero e no máximo muitos. Indica que pode ou não haver instâncias da classe participando do relacionamento
*	Muitos. Indica que muitos objetos da classe estão envolvidos na associação
1..*	No mínimo um e no máximo muitos. Indica que há pelo menos um objeto envolvido no relacionamento, podendo haver muitos objetos envolvidos
3..5	No mínimo 3 e no Máximo 5. Estabelece que existem pelo menos três instâncias envolvidas no relacionamento e que podem ser quatro ou cinco as instâncias envolvidas, mas não mais do que isso.

Herança

O que é herança?

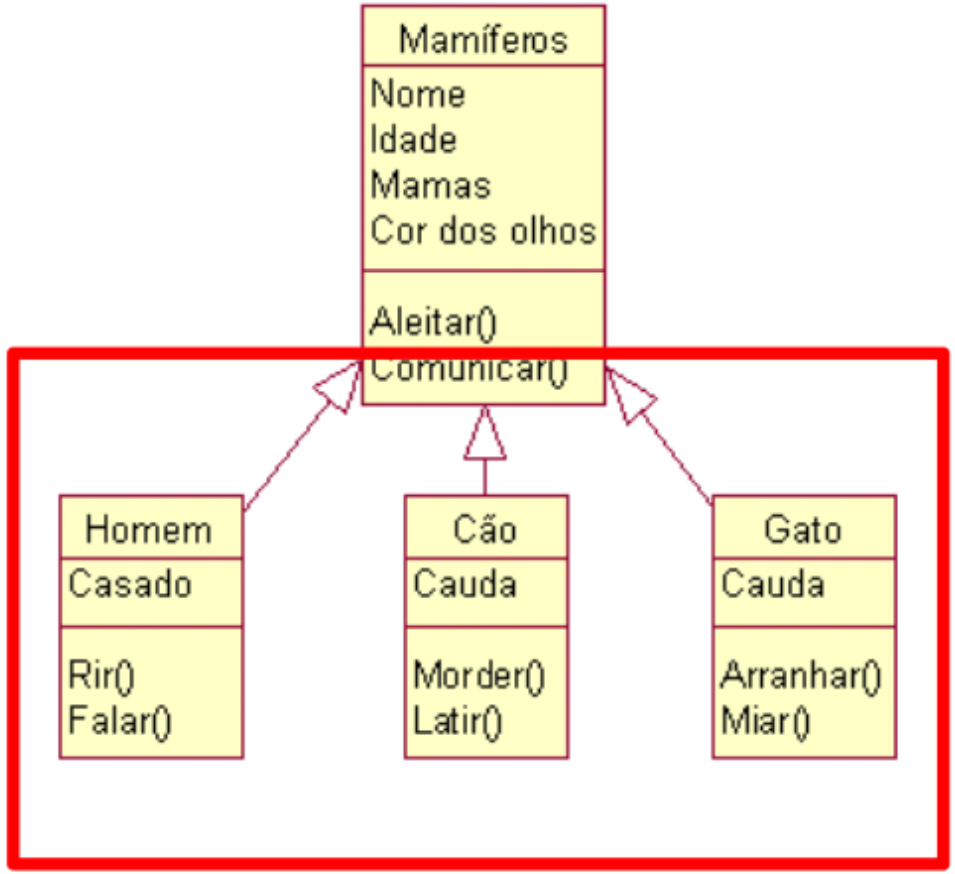
Herança é um mecanismo para reutilização de classes

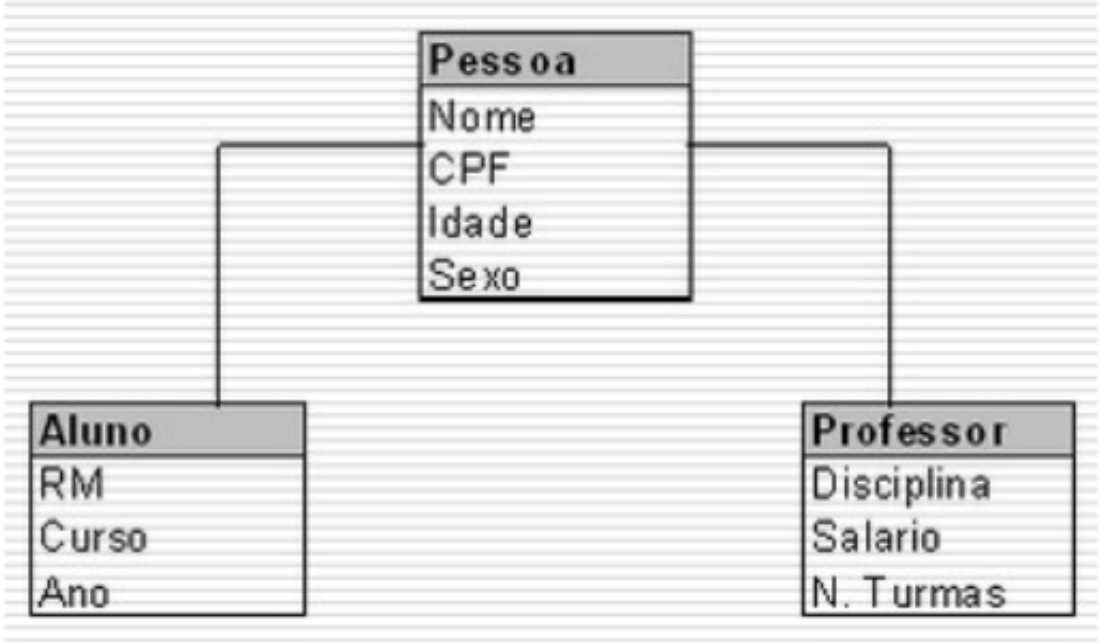
- É comum haver similaridades entre diferentes classes.
- Frequentemente, duas ou mais classes irão compartilhar os mesmos atributos e/ou métodos.
- Como nenhum de nós deseja reescrever várias vezes o mesmo código, seria interessante se algum mecanismo pudesse tirar proveito dessas similaridades.
- Por intermédio da herança, é possível modelar relacionamentos do tipo "é" ou "é semelhante", o que nos permite reutilizar rotinas e dados já existentes.

A classe filha herda os "bens" da classe mãe

- A herança está relacionada às hierarquias e às relações entre os objetos.
- É o mecanismo em que uma classe filha compartilha automaticamente todos os métodos e atributos de sua classe mãe.
- A herança permite implementar classes descendentes implementando os métodos e atributos que se diferenciam da classe mãe.

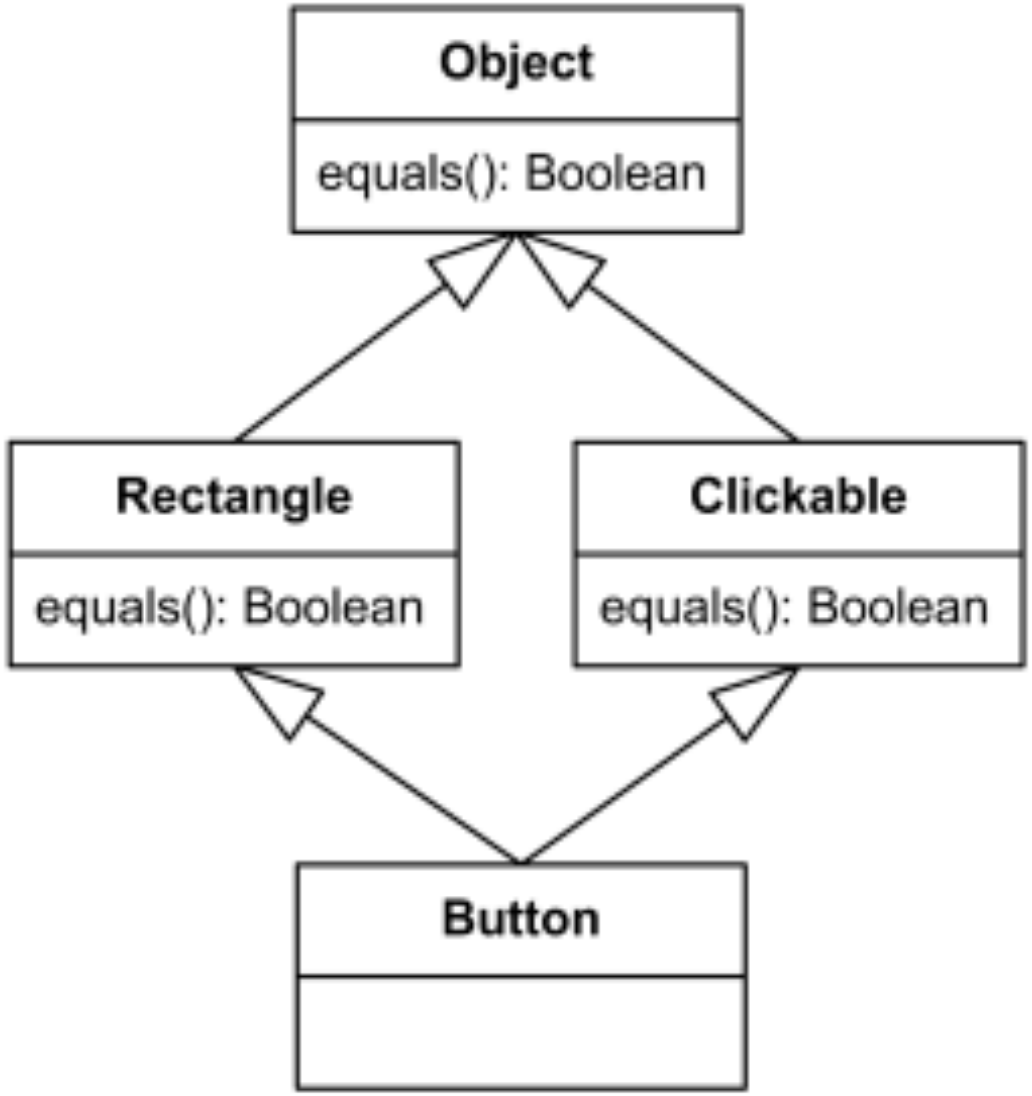
Exemplos de herança





Tipos de Herança

- Simples
  - Quando uma classe herda as propriedades de uma única classe mãe
- Múltipla
  - Ocorre quando uma classe tem mais de uma mãe



Sintaxe do mecanismo de herança

```
class veículo
{
  string cor;
  string combustivel;
  ...
  ...
};

class carro : public veiculo
{
 int nrodas;
 ...
 ...
 int mover( int nkilometros );

};
```

- A classe carro possui a extensão ": public veiculo“
- Referindo-se a uma declaração de parentesco.
- Estamos informando ao compilador que a classe veiculo é mãe da classe carro.
- A classe carro possui toda a estrutura da classe veiculo além de seus próprios membros.

```
#include <iostream>
using namespace std;
class veiculo_rodoviario // Define uma classe base veículos.
   {
        int rodas;
        int passageiros;
   public:
        void set_rodas(int num) { rodas = num; }
        int get_rodas() { return rodas; }
        void set_pass(int num) { passageiros = num; }
        int get_pass() { return passageiros; }
   };
class caminhao : public veiculo_rodoviario // Define um caminhao.
   {
          int carga;
   public:
           void set_carga(int size) { carga = size; }
           int get_carga() { return carga; }
           void mostrar();
   };
enum tipo {car, van, vagao};
class automovel : public veiculo_rodoviario // Define um automovel.
   {
        enum tipo car_tipo;
   public:
        void set_tipo(tipo t) { car_tipo = t; }
        enum tipo get_tipo() { return car_tipo; }
        void mostrar();
   };
void caminhao::mostrar()
```

- Temos a classe mãe veiculo_rodoviario e duas classes filhas “:” caminhao e automovel.
- Características comuns estão na classe mãe.
- Características exclusivas estão nas classes filhas.
- Eventuais modificações na classe mãe são estendidas a todos os objetos criados.
- Repare que as duas classes filhas possuem o método mostrar(), mas uma não interfere na outra, caracterizando o polimorfismo

Sintaxe do mecanismo de herança

Com herança, os atributos e métodos comuns a todas as classes na hierarquia são declarados na classe mãe. Quando alterações são necessárias para as características comuns, você só precisa modificar a classe mãe. As classes irão herdar as mudanças.

Ao instanciar uma classe filha, o construtor da classe mãe é executado primeiro, depois os membros da classe filha são inicializadas, e por fim o construtor da classe filha é executado.

Quando uma classe filha é destruída, seu método destrutor é chamado, depois invoca o destrutor da próxima classe mãe na hierarquia. O processo se repete até encontrar a classe mãe no topo da hierarquia.

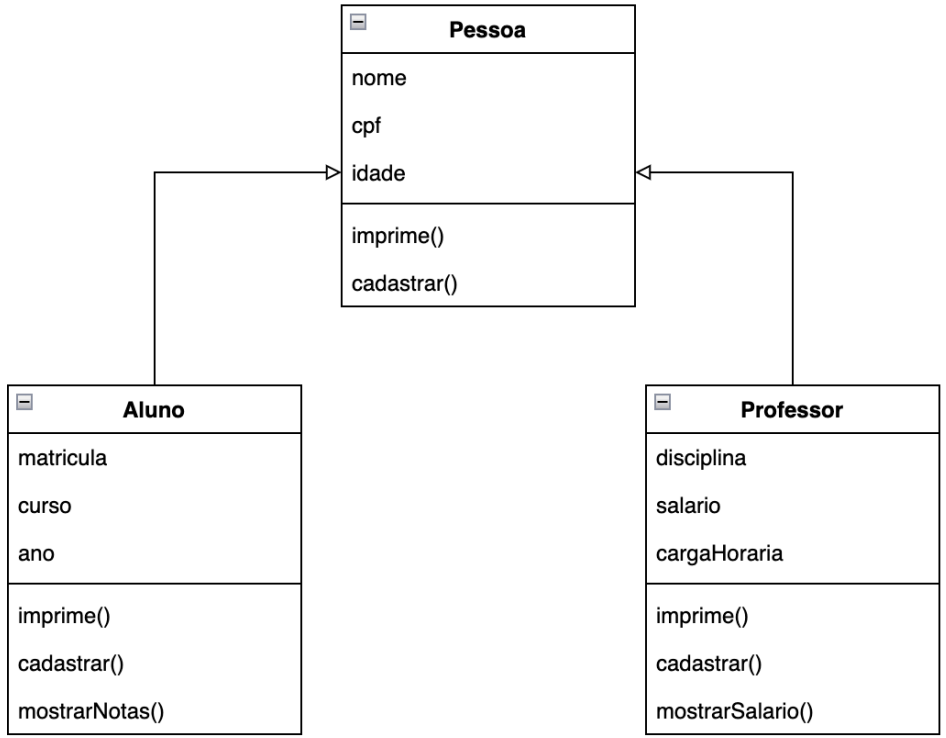
Polimorfismo

O que é polimorfismo?

Polimorfismo é o mecanismo que permite invocar métodos das classes filhas através da classe mãe.

Polimorfismo

- Da mesma forma que podemos reaproveitar classes (em Herança), podemos reaproveitar métodos (através de Polimorfismo);
- Ex: As classes aluno e professor, ambas possuem os métodos imprime() e cadastra(), cada um com seu objetivo, mas por estarem em classes diferentes, uma não interfere na outra. Mas o objetivo dos métodos é praticamente o mesmo.



O polimorfismo permite lidar com generalidades e deixar que o ambiente de tempo de execução se preocupe com os detalhes. Você pode direcionar uma variedade de objetos para se comportarem de maneira apropriada a esses objetos, mesmo sem conhecer seus tipos.

Classes abstratas

- Função virtual ocorre quando a classe mãe delega a implementação de um método para a classe filha.
- Um classe é abstrata quando possui pelo menos uma função virtual pura.
- Classes abstratas não podem ser instanciadas
- Se a função virtual não for implementada, então a classe filha também será abstrata.
- A classe abstrata pode ter método construtor.