UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Programação Orientada a Objetos

BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

PROF. LUCIO AGOSTINHO ROCHA

AULA 15: POLIMORFISMO

2º.SEMESTRE 2022

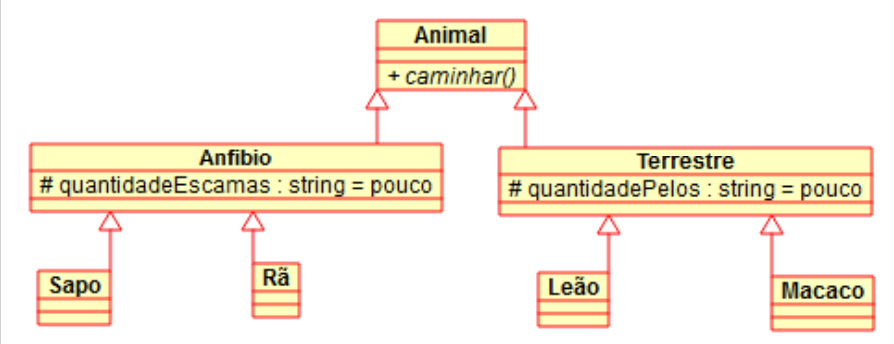
Polimorfismo

Polimorfismo

- Polimorfismo é um recurso de programação onde objetos que compartilham a mesma superclasse são tratados como objetos dessa superclasse.
- Polimorfismo: é um recurso que permite programar “no geral”, ao invés de programar “no específico”.
- Exemplo:
  - Suponha um programa que implemente a ação de caminhar dos seguintes animais: sapo, rã, leão e macaco.
  - Todos os animais caminham.
  - Todos os animais herdam da Classe Animal

Polimorfismo

- Polimorfismo:



- O que estas subclasses têm em comum?

Sapo	Rã	Leão	Macaco

Polimorfismo

- Cada classe derivada herda o método ‘caminhar()’ da superclasse Animal.
- Na Classe Principal é mantida uma lista de objetos das subclasses.
- O mesmo método “genérico” é enviado para cada objeto da classe derivada.
- Esse método é sobrecarregado por cada objeto, que o define da sua própria maneira.

Polimorfismo

- Polimorfismo:
  - Auxilia a construir programas extensíveis.
  - Programas processam genericamente objetos como superclasses
    - Novas classes podem ser facilmente inseridas no sistema
      - Novas classes devem fazer parte da hierarquia.
  - Exemplo:
    - Classe Coelho como subclasse da Classe Terrestre.
    - Classe Coelho herda o método genérico caminhar() e o implementa da sua própria maneira.

Polimorfismo

- Sem Polimorfismo:

```
public class Principal {
...
public static void main(String [ ] args){
   Sapo sapo = new Sapo();
   Ra ra = new Ra();
   Leao leao = new Leao();
   Macaco macaco = new Macaco();

   sapo.caminhar();
   ra.caminhar();
   leao.caminhar();
   macaco.caminhar();
}
```

Polimorfismo

- Com Polimorfismo:

```
public class Principal {
...
public static void main(String [ ] args){
   ArrayList<Animal> lista = new ArrayList<>();
   lista.add( new Sapo() );
   lista.add( new Ra() );
   lista.add( new Leao() );
   lista.add( new Macaco() );

   for( Animal  animal : lista )
       animal.caminhar();
}
```


Polimorfismo

- Polimorfismo:
  - Um objeto da classe base (superclasse) sempre pode receber um objeto da classe derivada (subclasse).
  - Ex.: Sapo é um Animal.

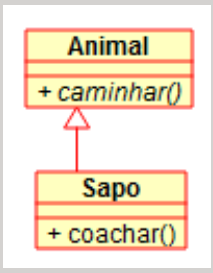
```
Animal  animal = new Sapo();
animal.caminhar();              //Método da classe derivada
```

    - O inverso não é permitido.

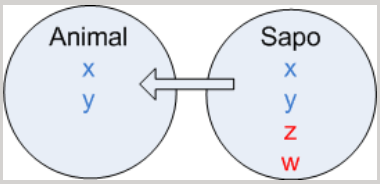
  - O objeto 'animal' continua sendo da Classe Animal, porém, adquiriu mais membros da classe derivada. Logo, caso se queira acessar membros da subclasse específica, é necessário explicitar a subclasse:

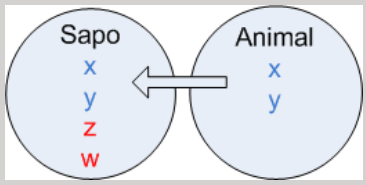
```
((Sapo) animal).coachar();    //Método específico da subclasse
```

Polimorfismo



- Sapo é um Animal





- Faltam campos

Polimorfismo

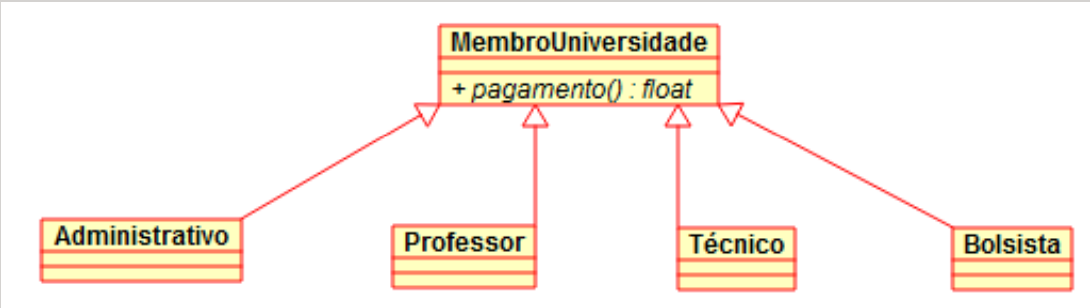

```
public class Principal {

public static void main(String [] args){
   Animal animal = new Sapo();
  animal.caminhar();                    //metodo generico

  animal = new Leao();                  //mudou de forma
  animal.caminhar();                    //metodo generico
}

}
```

Polimorfismo

- Outro exemplo de Polimorfismo:
  - Um programa de planilha de pagamentos dos membros de uma universidade.
  - Os membros da universidade são:
    - Administrativo, Professor, Técnicos, Bolsistas.
  - Todos os membros herdam da Classe MembroUniversidade.
  - Todos os membros recebem um pagamento.

Polimorfismo

- Polimorfismo:



- O que estas subclasses têm em comum?

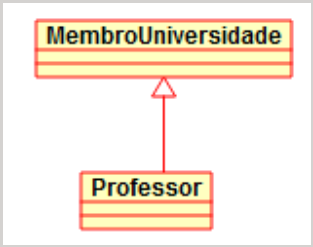
Administrativo	Professor	Técnico	Bolsista

Polimorfismo

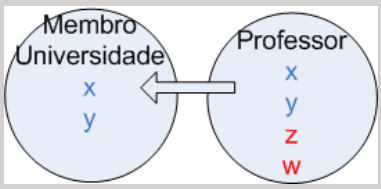
- Com Polimorfismo:

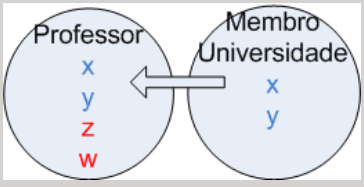
```
public class Principal {
...
public static void main(String [ ] args){
   ArrayList<MembroUniversidade> lista = new ArrayList<>();
   lista.add( new Administrativo() );
   lista.add( new Professor() );
   lista.add( new Tecnico() );
   lista.add( new Bolsista() );

   for( MembroUniversidade  membro : lista )
       membro.pagamento();
}
```


Polimorfismo



- Professor é um MembroUniversidade





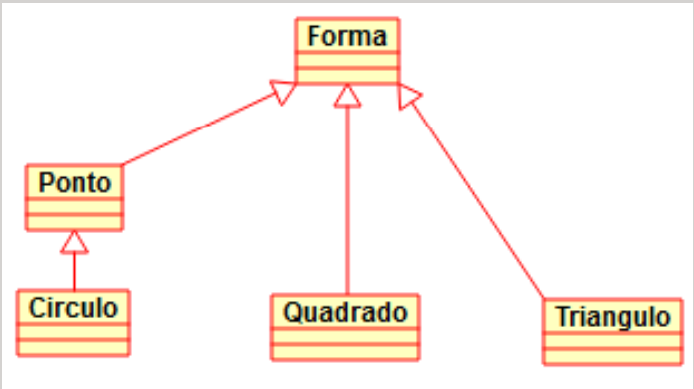
- Faltam campos

Polimorfismo


```
public class Principal {

public static void main(String [] args){
   MembroUniversidade membro = new Professor();
   membro.pagamento();                    //metodo generico

  membro = new Bolsista();               //mudou de forma
  membro.pagamento();                    //metodo generico
}

}
```


Polimorfismo

- Outro exemplo de Polimorfismo:

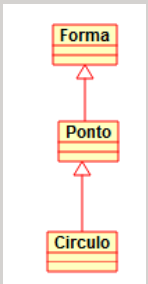


Polimorfismo

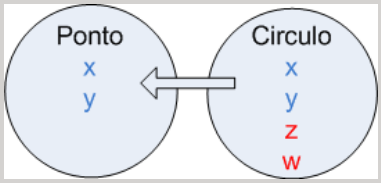
- Com Polimorfismo:

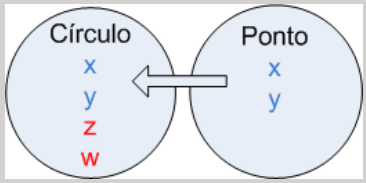
```
public class Principal {
...
public static void main(String [ ] args){
   ArrayList<Forma> lista = new ArrayList<>();
   lista.add( new Circulo() );
   lista.add( new Triangulo() );
   lista.add( new Quadrado() );
   lista.add( new Ponto() );

   for( Forma forma : lista )
       forma.imprimir();
}
```


Polimorfismo



- Círculo é um Ponto.
- Círculo é uma Forma.





- Faltam campos

Polimorfismo


```
public class Principal {

public static void main(String [] args){
   Forma forma = new Circulo();
   forma.imprimir();                    //metodo generico

  forma = new Triangulo();          //mudou de forma
  forma.imprimir();                    //metodo generico
}

}
```


Revisão

Revisão

- Polimorfismo
- Interface
- Classe Abstrata

Revisão

- Ligação dinâmica:
  - Implementa processamento polimórfico de objetos.
  - Utiliza a referência da Superclasse para o objeto da Subclasse.
  - Os métodos são os mesmos, mas cada objeto de uma Subclasse implementa o seu (sobrecarga).

Revisão

- Alternativas ao polimorfismo: switch
  - Tratar cada novo objeto em uma estrutura switch
  - Problemas:
    - Deixar a critério exclusivo do programador a validação dos Membros da Classe
    - Adicionar e remover novas classes.

Exercícios

<Ver conteúdo na plataforma de ensino>

Referências

- Referências bibliográficas da disciplina.