

Universidade Federal da Paraíba

Centro de Informática

Departamento de Informática

Linguagem de Programação I

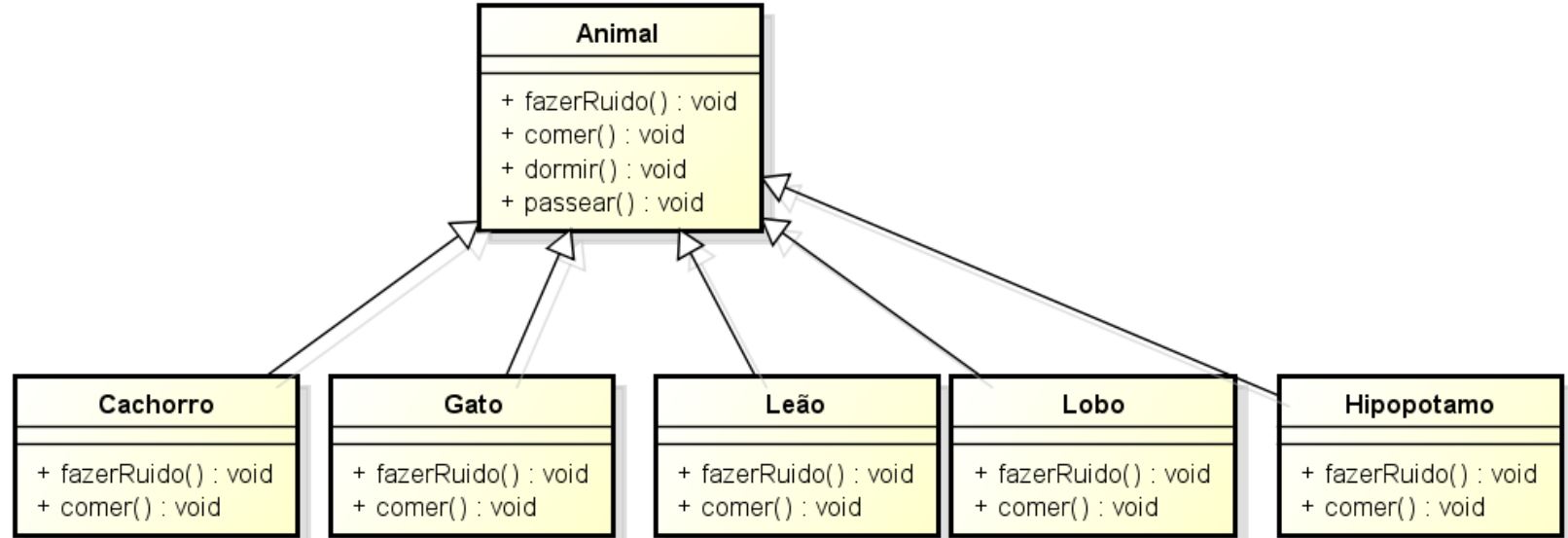
Polimorfismo

- ▶ Tiago Maritan
- ▶ tiago@ci.ufpb.br



Motivação

- ▶ Com a **herança** podemos **evitar códigos duplicados**
 - ▶ Ex: `Conversivel` não replica códigos de `Veiculo`
- ▶ Inserimos um **código comum em uma superclasse** e deixamos as **subclasses herdarem** esse código.



Motivação

- ▶ Ou seja, estamos definindo um **protocolo comum** para um grupo de classes...



- ▶ Todos os subtipos de `Animais` (`Cachorro`, `Gato`, `Lobo`, **etc.**) poderão fazer as mesmas coisas...
 - ▶ `comer()`, `fazerRuido()`, `dormir()`, `passear()`...



Motivação

► Exemplo:

```
Animal animal1 = new Animal();  
animal1.comer();  
animal1.fazerRuido();  
  
Cachorro cachorro1 = new Cachorro();  
cachorro1.comer();  
cachorro1.fazerRuido();  
  
Gato gato1 = new Gato();  
gato1.comer();  
gato1.fazerRuido();  
  
Lobo lobo1 = new Lobo();  
lobo1.comer();  
lobo1.fazerRuido();
```



Motivação

- ▶ Lembrando como funciona a declaração e criação de objetos

I. Declaração da variável

```
Cachorro cachorro1 = new Cachorro();
```

Declara a variável cachorro1



cachorro1



Motivação

- ▶ Lembrando como funciona a declaração e criação de objetos

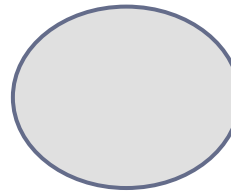
2. Criação do Objeto (Instanciação)

```
Cachorro cachorro1 = new Cachorro();
```

Aloca espaço para um objeto do tipo Cachorro



cachorro1




**Objeto
Cachorro**



Motivação

- ▶ Lembrando como funciona a declaração e criação de objetos
3. Vincula o objeto a variável de referência

```
Cachorro cachorro1  new Cachorro();
```

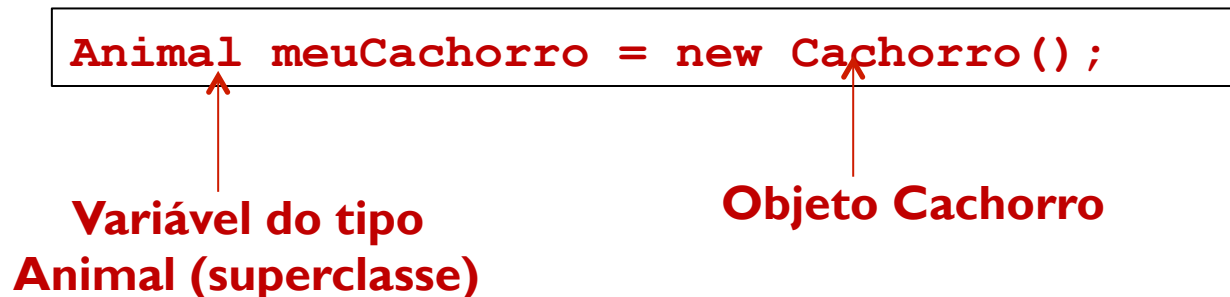
Atribui o novo objeto do tipo Cachorro a cachorro1



Polimorfismo

- ▶ Na POO, o **tipo da variável de referência** também pode ser uma superclasse.

- ▶ Exemplo:



- ▶ Isso define o conceito de **polimorfismo**.



Polimorfismo

► Exemplo: Criação de um array polimórfico

```
Animal []animais = new Animal[5];

animais[0] = new Cachorro();
animais[1] = new Gato();
animais[2] = new Lobo();
animais[3] = new Leao();
animais[4] = new Hipopotamo();

for (int i = 0; i < animais.length; i++){
    animais[i].comer();
    animais[i].dormir();
}
```



Polimorfismo

- Podemos ter também **atributos** ou **tipos de retorno** polimórfico

```
public class Veterinario{  
    public void vacinar(Animal a){  
        a.fazerRuido();  
    }  
}
```

```
public class DonoPetShop{  
    public void start(){  
        Veterinario v = new Veterinario();  
  
        Cachorro c = new Cachorro();  
        Hipopotamo h = new Hipopotamo();  
  
        v.vacinar(c);  
        v.vacinar(h);  
    }  
}
```

Polimorfismo

- Podemos ter também **atributos** ou **tipos de retorno** polimórfico

```
public class Veterinario{  
    public void vacinar(Animal a){  
        a.fazerRuido();  
    }  
}
```

← **Parâmetro pode ser qualquer tipo de Animal**

```
public class DonoPetShop{  
    public void start(){  
        Veterinario v = new Veterinario();  
  
        Cachorro c = new Cachorro();  
        Hipopotamo h = new Hipopotamo();  
  
        v.vacinar(c);  
        v.vacinar(h);  
    }  
}
```

← **Chamará o método fazerRuido() de Cachorro**

Polimorfismo

- Podemos ter também **atributos** ou **tipos de retorno** polimórfico

```
public class Veterinario{  
    public void vacinar(Animal a){  
        a.fazerRuido();  
    }  
}
```

← **Parâmetro pode ser qualquer tipo de Animal**

```
public class DonoPetShop{  
    public void start(){  
        Veterinario v = new Veterinario();  
  
        Cachorro c = new Cachorro();  
        Hipopotamo h = new Hipopotamo();  
  
        v.vacinar(c);  
        v.vacinar(h);  
    }  
}
```

← **Chamará o método fazerRuido() de Hipopotamo**

Polimorfismo

- ▶ Portanto, se usarmos **argumentos polimórficos** (parâmetros com o tipo da superclasse)...
- ▶ podemos passar qualquer objeto subclasse na chamada (em tempo de execução...)
 - ▶ Ex: Argumento de `vacinar()` é do tipo `Animal`, então podemos passar qualquer subclasse (`Cachorro`, `Gato`, etc.) na chamada!



Polimorfismo

- ▶ Além disso, o código de `Veterinário` não precisa ser alterado se um novo tipo de subclasse for introduzido:
- ▶ Criar uma subclasse `Tigre`

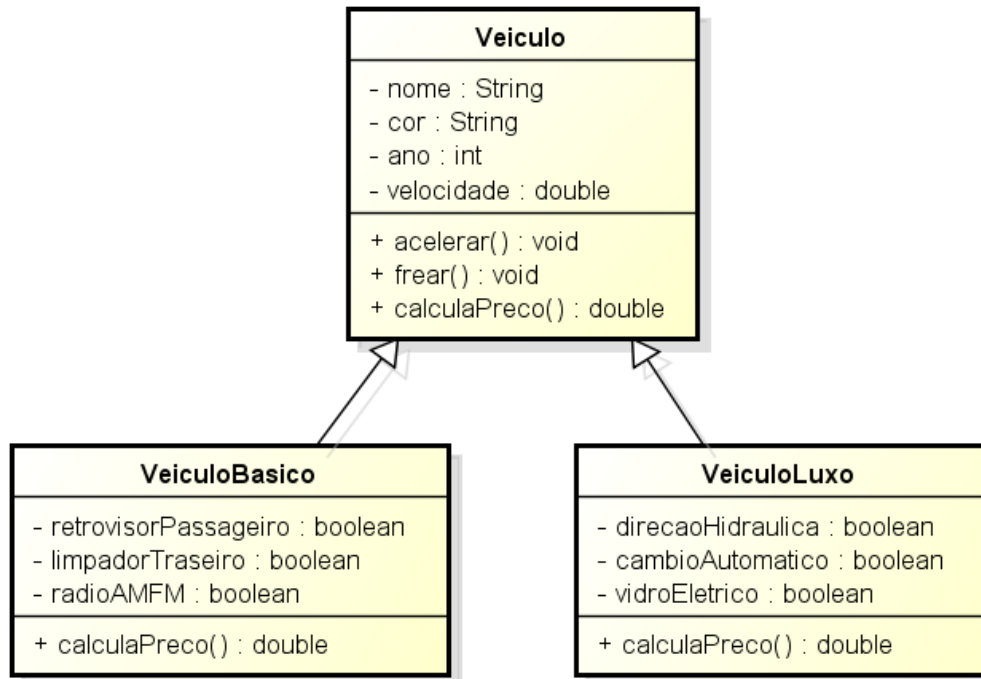
```
public class Veterinario{  
    public void vacinar(Animal a) {  
        a.fazerRuido();  
    }  
}
```

Posso vacinar() um Tigre (nova subclasse)
ou qualquer outro animal novo sem
modificar meu código

```
public class DonoPetShop{  
    public void start(){  
        Veterinario v = new Veterinario();  
  
        Tigre t = new Tigre();  
        v.vacinar(t); ← vacinando o Tigre  
    }  
}
```

Polimorfismo

► Exemplo2:



powered by astah*

Polimorfismo

► Exemplo2:

```
public class Concessionaria {  
    public void imprimePreco(Veiculo v) {  
        System.out.println(v.calculaPreco());  
    }  
}
```

Argumento polimórfico

Chamada polimórfica

```
public class Teste {  
    public void start() {  
        Concessionaria c = new Concessionaria();  
        VeiculoBasico vb1 = new VeiculoBasico();  
        vb1.setRadioAMFM(true);  
  
        VeiculoLuxo vl1 = new VeiculoLuxo();  
        vl1.setDirecaoHidraulica(true);  
  
        c.imprimePreco(vb1);  
        c.imprimePreco(vl1);  
    }  
}
```


Polimorfismo

► Exemplo2:

```
public class Concessionaria {  
    public void imprimePreco(Veiculo v) {  
        System.out.println(v.calculaPreco());  
    }  
}
```

Argumento polimórfico

Chamada polimórfica

```
public class Teste {  
    public void start() {  
        Concessionaria c = new Concessionaria();  
        VeiculoBasico vb1 = new VeiculoBasico();  
        vb1.setRadioAMFM(true);  
  
        VeiculoLuxo vl1 = new VeiculoLuxo();  
        vl1.setDirecaoHidraulica(true);  
  
        c.imprimePreco(vb1);  
        c.imprimePreco(vl1);  
    }  
}
```

Chamada calculaPreco de
VeiculoBasico

Polimorfismo

► Exemplo2:

```
public class Concessionaria {  
    public void imprimePreco(Veiculo v) {  
        System.out.println(v.calculaPreco());  
    }  
}
```

Argumento polimórfico

Chamada polimórfica

```
public class Teste {  
    public void start() {  
        Concessionaria c = new Concessionaria();  
        VeiculoBasico vb1 = new VeiculoBasico();  
        vb1.setRadioAMFM(true);  
  
        VeiculoLuxo vl1 = new VeiculoLuxo();  
        vl1.setDirecaoHidraulica(true);  
  
        c.imprimePreco(vb1);  
        c.imprimePreco(vl1);  
    }  
}
```

Chamada calculaPreco de
VeiculoLuxo

Universidade Federal da Paraíba

Centro de Informática

Departamento de Informática

Linguagem de Programação I

Polimorfismo

- ▶ Tiago Maritan
- ▶ tiago@ci.ufpb.br