# Programação Orientada a Objetos2

INTERFACE

# CONCEITOS

## INTERFACE

- São padrões definidos através de contratos ou especificações
- Um contrato define um determinado conjunto de métodos que serão implementados nas classes que assinarem esse contrato.
- Todos os métodos da interface são abstratos
- Se possuir atributos, estes devem ser constantes estáticas (static final)

## DECLARAÇÃO DE UMA INTERFACE

```
interface  nome_da_interface  {

    public abstract  tipo  metodo1( ... ) ;

    public abstract  tipo  metodo2( ... );

                ⋮

}
```

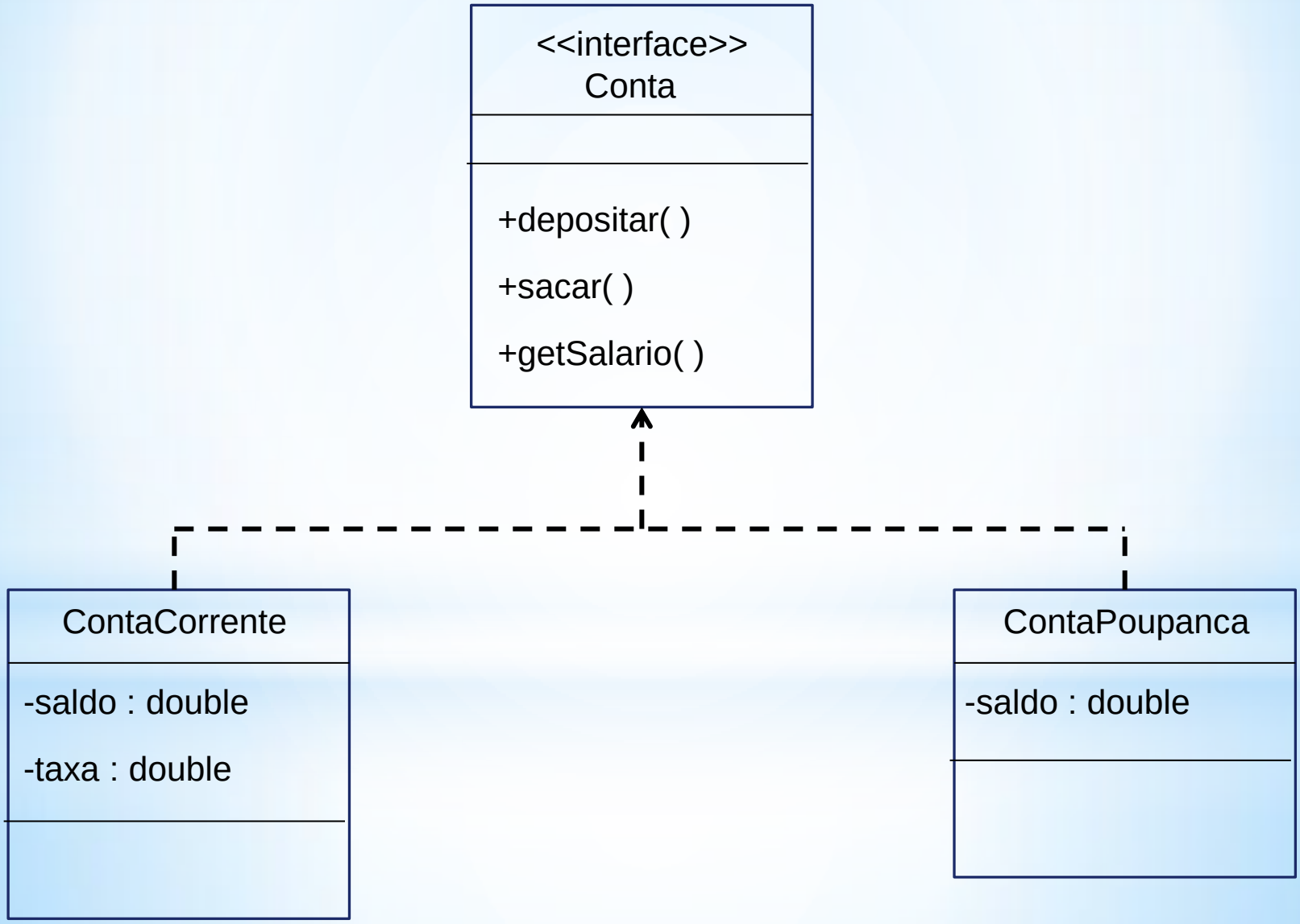
# CONCEITOS

**IMPLEMENTAÇÃO DE UMA INTERFACE**

```
class   nome_classe_implementadora   implements   nome_da_interface  {
        public  tipo  metodo1( ... ) {
              ⋮
        }
        public  tipo  metodo2( ... ) {
              ⋮
         }
}
```

# CONCEITOS

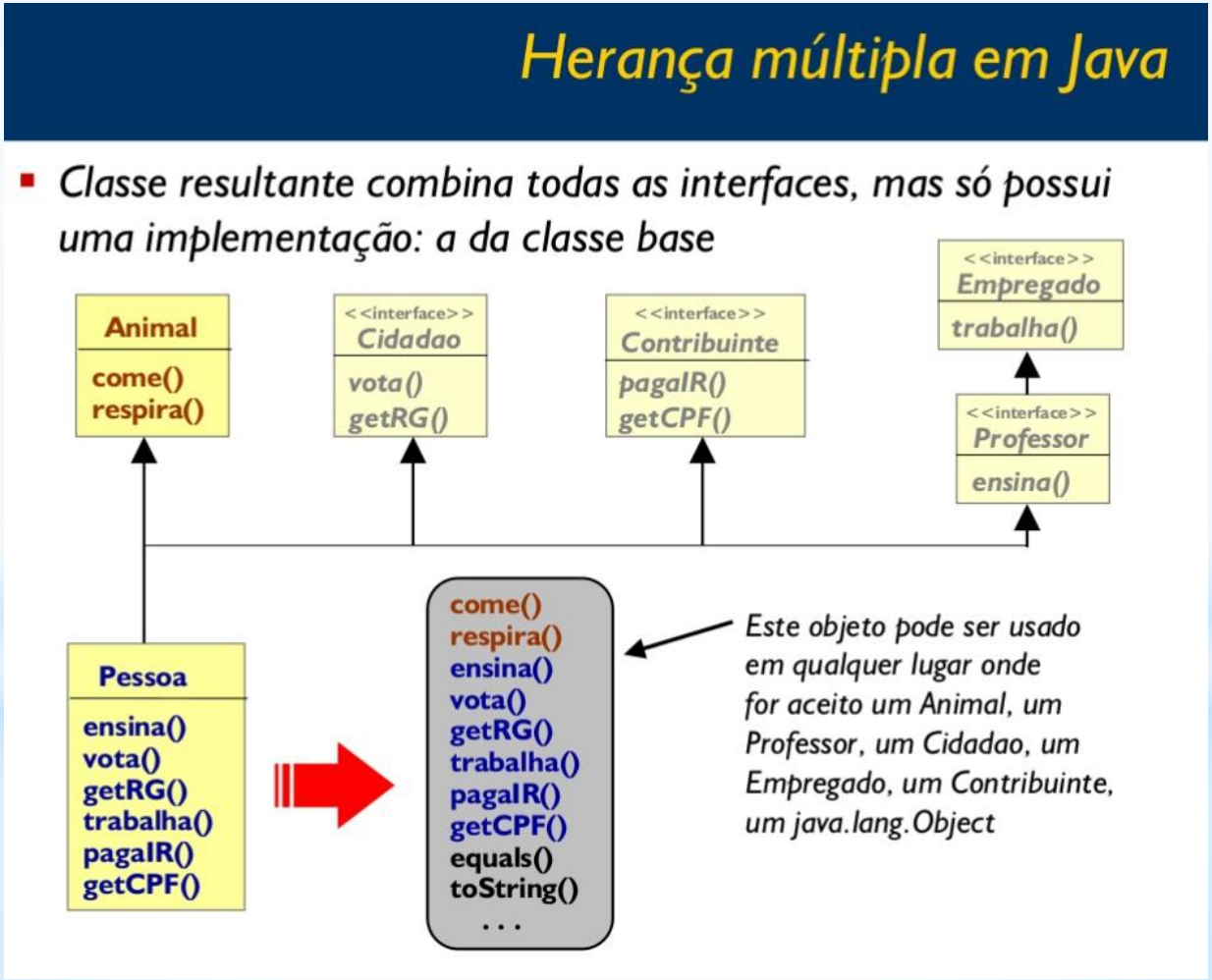
**INTERFACES NA UML**



# CONCEITOS

## IMPLEMENTAÇÃO DE VÁRIAS INTERFACES

- Uma classe pode implementar várias interfaces separando-as por vírgulas



# CONCEITOS

## IMPLEMENTAÇÃO DE UMA INTERFACE

- Uma classe pode implementar várias interfaces separando-as por vírgulas

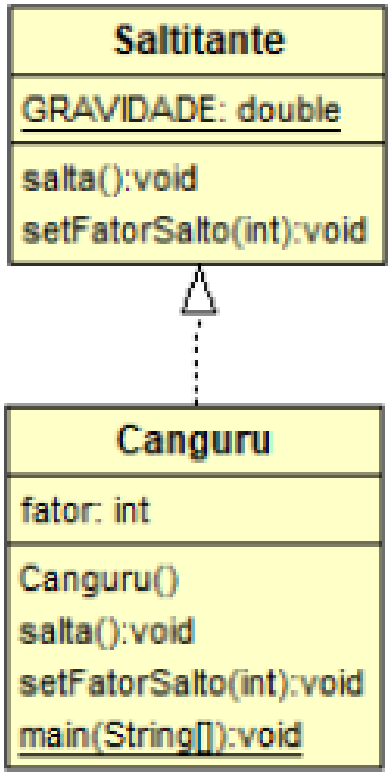
```
class Pessoa extends Animal implements Cidadao, Contribuinte, Professor {

        |
        |
        |
        |

}
```

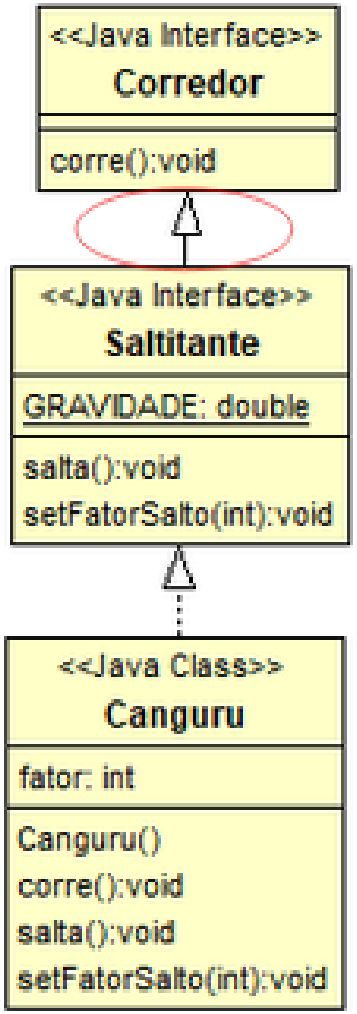
# Interface

```
package heranca;
public interface Saltitante {
    double GRAVIDADE = 9.8;  //é public static e final
    void salta();
    void setFatorSalto(int fator);
}
class Canguru implements Saltitante {
    int fator;
    public void salta() {
        System.out.println("saltando!");
    }
    public void setFatorSalto(int fator) {
        this.fator = fator;
    }
}
```



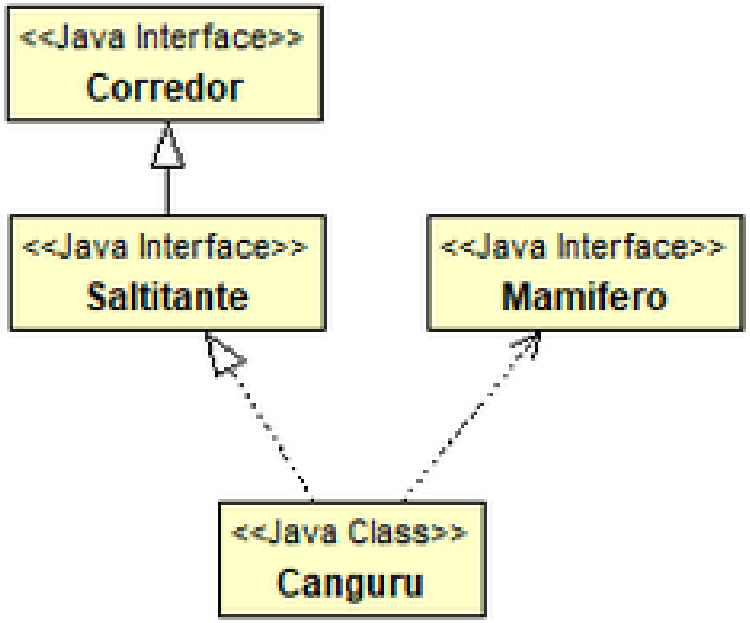
# Interface

```
interface Corredor{void corre();}
interface Saltitante extends Corredor {
    double GRAVIDADE = 9.8;//public static
    final
    void salta();
    void setFatorSalto(int fator);}
class Canguru implements Saltitante {
    int fator;
    public void corre() {
        System.out.println("correndo!");}
    public void salta() {
        System.out.println("saltando!");}
    public void setFatorSalto(int fator) {
        this.fator = fator;}
}
```

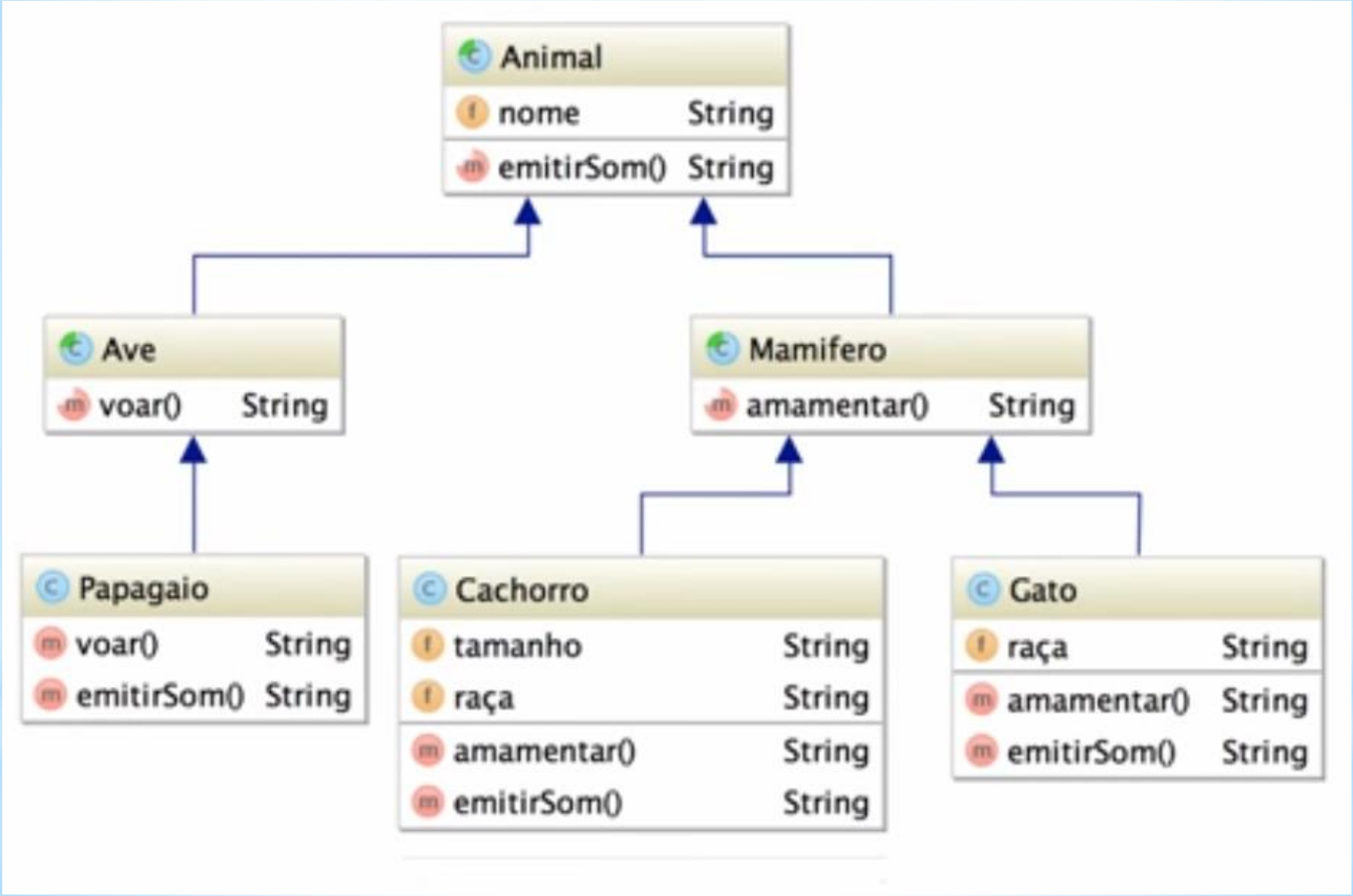


```
interface Saltitante extends Corredor {
     double GRAVIDADE = 9.8;
     void salta();
     void setFatorSalto(int fator);
}
class Canguru implements Saltitante,
     Mamifero {
     int fator;
     public void corre() {
          System.out.println("correndo!"); }
     public void salta() {
          System.out.println("saltando!"); }
     public void mama() {
          System.out.println("mamando!"); }
     public void setFatorSalto(int fator) {
          this.fator = fator;
     }
}
```

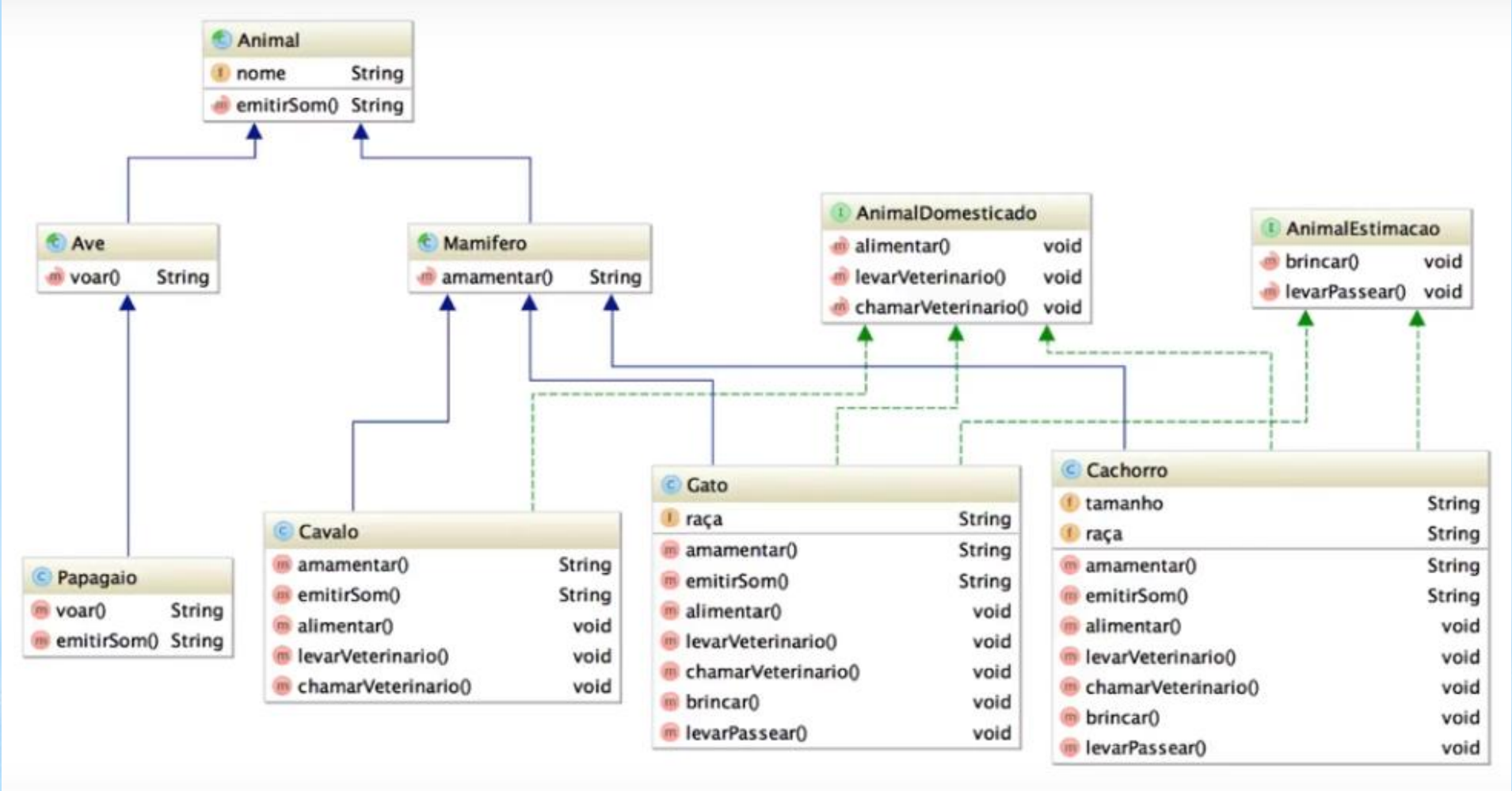
```
interface Corredor {
     void corre();
}
interface Mamifero {
     void mama();
}
```



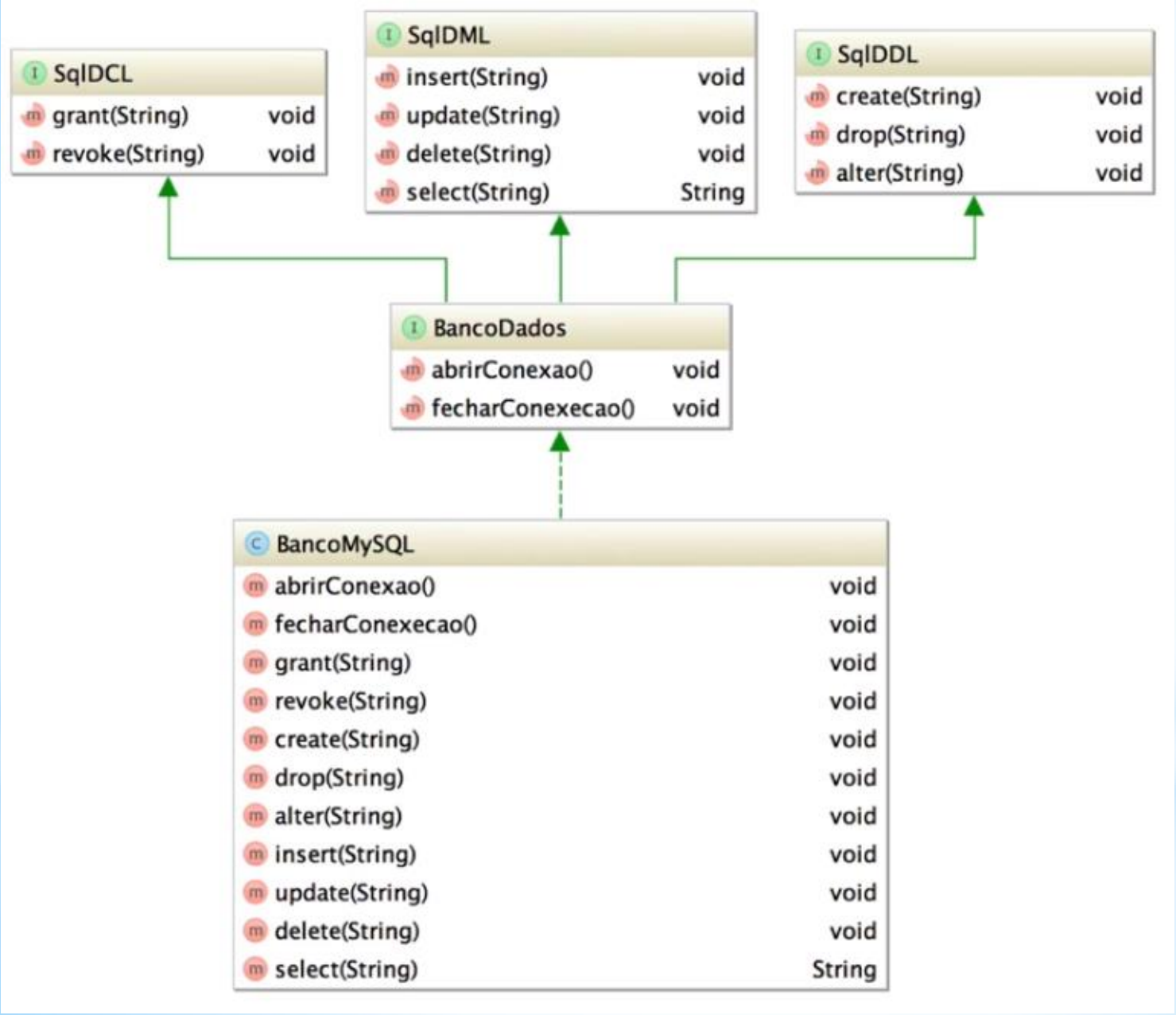
| Interface | Classe Abstrata |
| --- | --- |
| Herança múltipla permitido; uma interface pode estender várias interfaces | Herança múltipla não é possível; uma classe só pode estender uma única classe |
| palavra chave **implements** é utilizada para implementar uma interface | palavra chave **extends** é utilizada para estender uma classe |
| por padrão todos os métodos são públicos e abstratos (public abstract) - não tem necessidade de declarar os mesmos | métodos podem ter modificares public e abstract se necessário, e podem utilizar outros modificares também |
| interfaces não tem implementação | podem ter implementação parcial |
| todos os métodos de uma interface precisam ser sobrescritos | somente métodos abstratos precisam ser sobrescritos (obrigatório) |
| todas as variáveis declaradas numa interface são **public static final** (constantes) | variáveis podem ser declaradas como **public static final** se necessário, mas não é obrigatório |
| interfaces não tem construtor(es) | classes abstratas podem ter construtores |
| métodos não podem ser estáticos (**static**) | métodos não abstratos podem ser estáticos (**static**) |


Animal
nome String
emitirSom() String
Ave
voar() String
Mamifero
amamentar() String
Papagaio
voar() String
emitirSom() String
Cachorro
tamanho String
raça String
amamentar() String
emitirSom() String
Gato
raça String
amamentar() String
emitirSom() String

# ATIVIDADE 1


Animal
nome String
emitirSom() String
Ave
voar() String
Mamifero
amamentar() String
AnimalDomesticado
alimentar() void
levarVeterinario() void
chamarVeterinario() void
AnimalEstimacao
brincar() void
levarPassear() void
Papagaio
voar() String
emitirSom() String
Cavalo
amamentar() String
emitirSom() String
alimentar() void
levarVeterinario() void
chamarVeterinario() void
Gato
raça String
amamentar() String
emitirSom() String
alimentar() void
levarVeterinario() void
chamarVeterinario() void
brincar() void
levarPassear() void
Cachorro
tamanho String
raça String
amamentar() String
emitirSom() String
alimentar() void
levarVeterinario() void
chamarVeterinario() void
brincar() void
levarPassear() void

# ATIVIDADE 2




UMA INTERFACE PODE HERDAR DE VÁRIAS INTERFACES