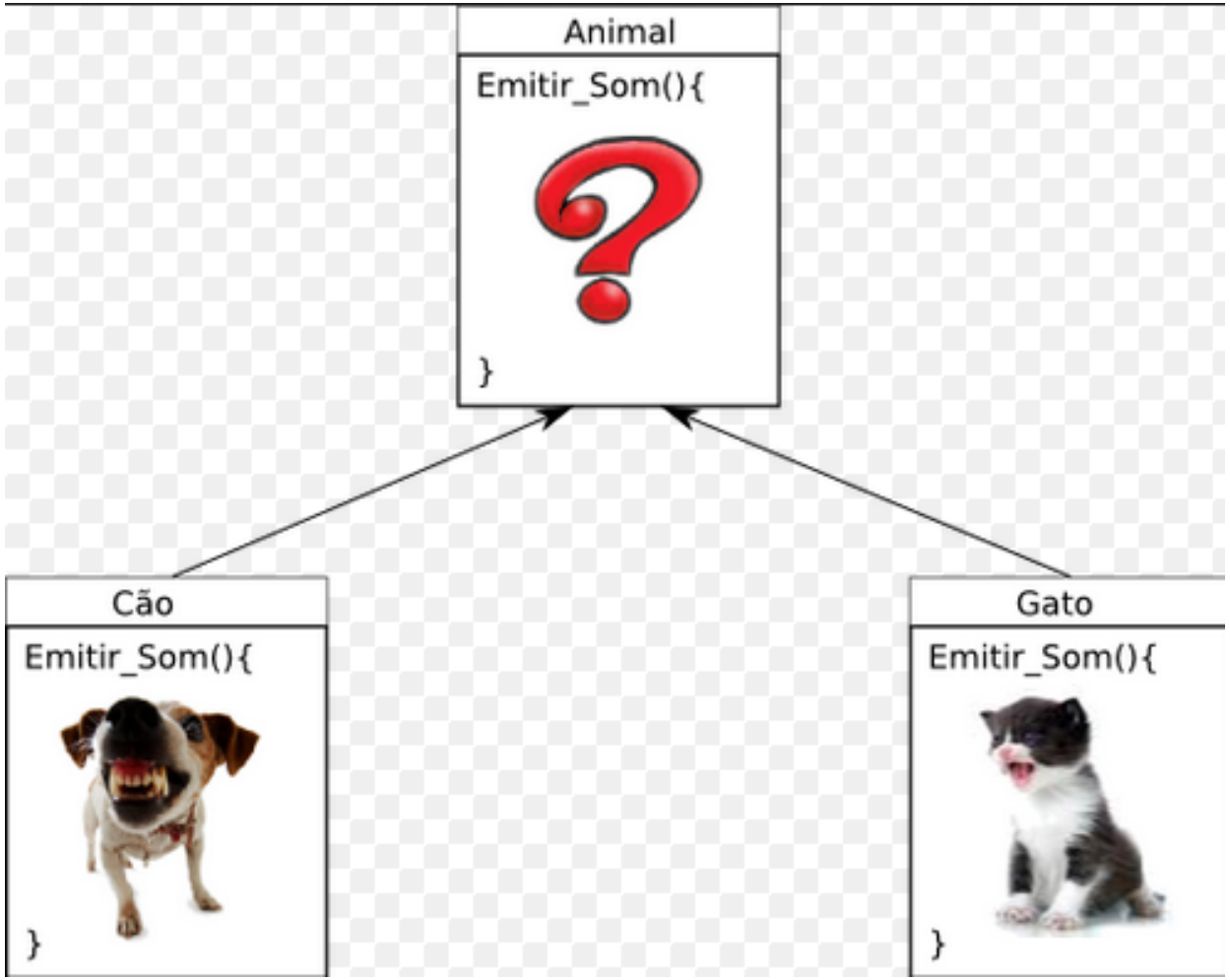
Polimorfismo

- A palavra polimorfismo vem do grego poli morfos e significa muitas formas. Na orientação a objetos, isso representa uma característica que permite que classes diferentes sejam tratadas de uma mesma forma.
- O polimorfismo permite escrever programas que processam objetos que compartilham a mesma superclasse (direta ou indiretamente) como se todos fossem objetos da superclasse; isso pode simplificar a programação (DEITEL; DEITEL, 2010, p. 305).

- Polimorfismo é o princípio pelo qual duas ou mais classes derivadas de uma mesma superclasse podem invocar métodos que têm a mesma identificação (assinatura) mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse. A decisão sobre qual o método que deve ser selecionado, de acordo com o tipo da classe derivada, é tomada em tempo de execução, através do mecanismo de ligação tardia.

- No caso de polimorfismo, é necessário que os métodos tenham exatamente a mesma identificação, sendo utilizado o mecanismo de redefinição de métodos. Esse mecanismo de redefinição não deve ser confundido com o mecanismo de sobrecarga de métodos.
- É importante observar que, quando polimorfismo está sendo utilizado, o comportamento que será adotado por um método só será definido durante a execução. Embora em geral esse seja um mecanismo que facilite o desenvolvimento e a compreensão do código orientado a objetos, há algumas situações onde o resultado da execução pode ser não-intuitivo.


Animal
Emitir_Som(){
}
Cão
Emitir_Som(){
}
Gato
Emitir_Som(){
}

- **public class Animal** {
  public void comer() {
    System.out.println("Animal Comendo...");
  }
}
- **public class Cao extends Animal** {
  public void comer() {
    System.out.println("Cão Comendo...");
  }
}
- **public class Tigre extends Animal** {
  public void comer() {
    System.out.println("Tigre Comendo...");
  }
}

- public static **class Test** {

```
public void fazerAnimalComer( Animal animal ) {
        animal.comer();
}
 public static void main( String[] args ) {
        Test t = new Test();
        t.fazerAnimalComer( new Animal() );
        t.fazerAnimalComer( new Cao() );
        t.fazerAnimalComer( new Tigre() ); } }
```