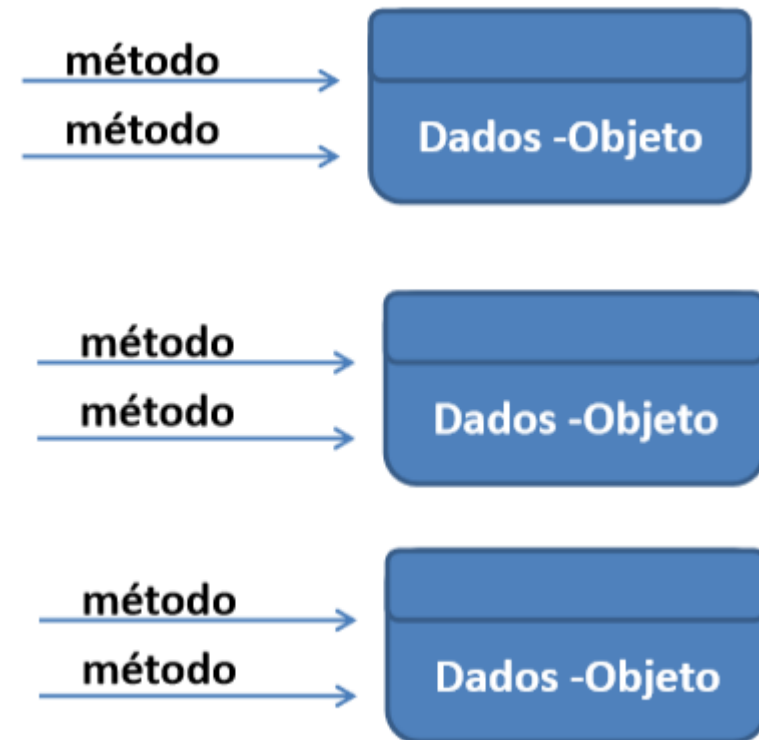
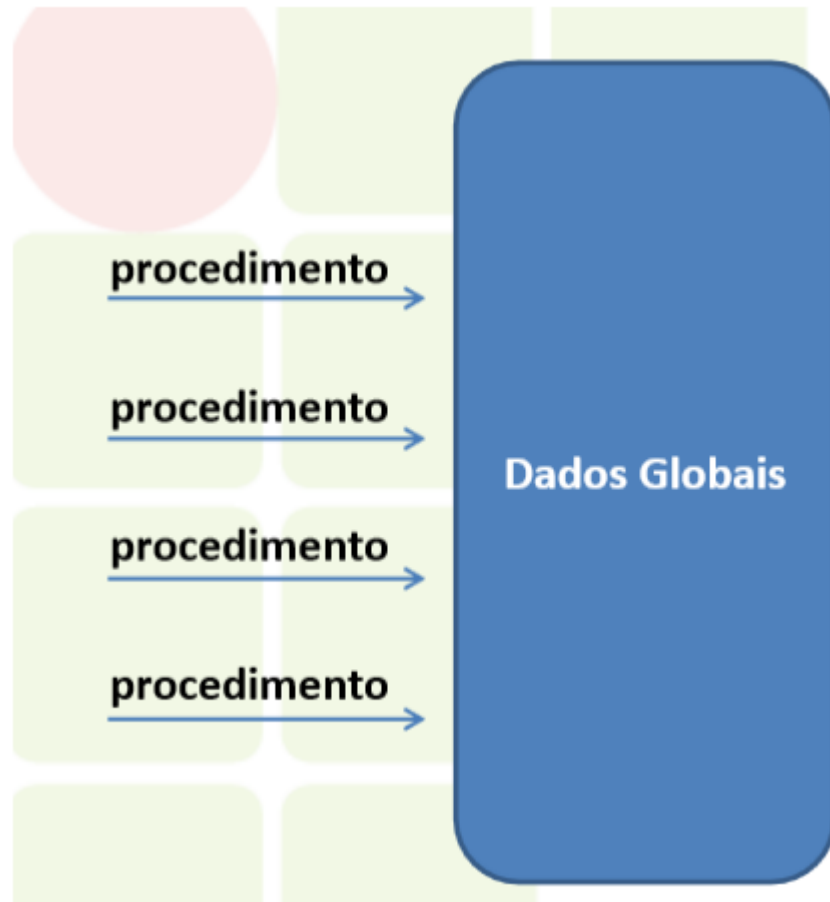


Orientação a objetos

Orientação a objetos

- Comparação Programação Estrutura e POO
- Definição de Objetos
 - Conceito, propriedades
- Definição de Classe
- Elementos Principais de Programação Orientada a Objeto
 - Abstração, Encapsulamento, Modularidade, Herança,
- Polimorfismo
- Visibilidade
- Classe em C#

Estrutural x OO



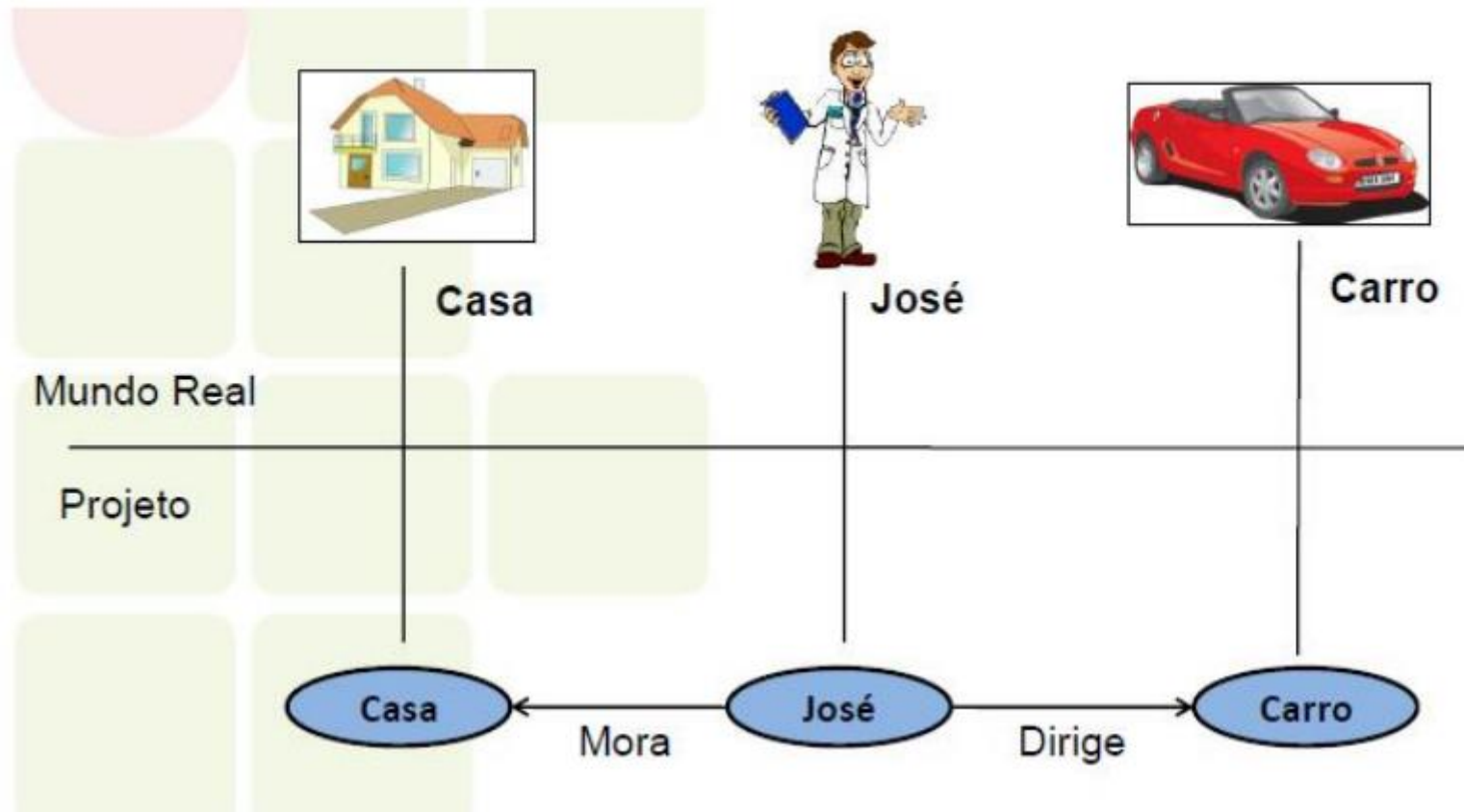
Estrutural x OO

- Consiste no mapeamento do problema no mundo real a ser resolvido num modelo computacional.
- Programação Estruturada
 - Consiste na criação de um conjunto de procedimentos (algoritmos) para resolver o problema
 - Encontrar modos apropriados de armazenar os dados
- Programação Orientada a Objetos
 - Consistem em identificar os objetos e as operações relevantes no mundo real
 - O mapeamento desses em representações abstratas no espaço de soluções

Orientação a objetos

- Paradigma de Programação
- Dominante nos dias atuais
- Substituiu as técnicas de programação procedimental (estruturada)
- “Fornece um mapeamento direto entre o mundo real e as unidades de organização utilizadas no projeto”
- Diversas unidades de software, chamadas de objetos, que interagem entre si
- Separa claramente a noção de o que é feito de como é feito

Orientação a objetos



Definição de objetos

- Um objeto é algo do mundo real:
 - Concreto ou Abstrato
- A percepção dos seres humanos é dada através dos objetos
- Um objeto é uma entidade que exhibe algum comportamento bem definido.



Definição de objetos

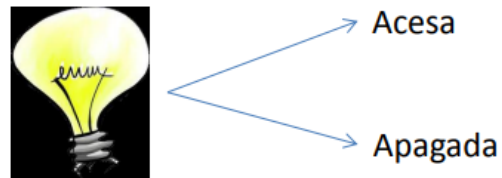
- Características
 - Dados representam características
 - São chamados atributos
 - São as variáveis do objeto
- Comportamento
 - Operações definem comportamento
 - São os métodos de um objeto
 - São as funções que são executadas por um objeto

Objetos - Propriedades

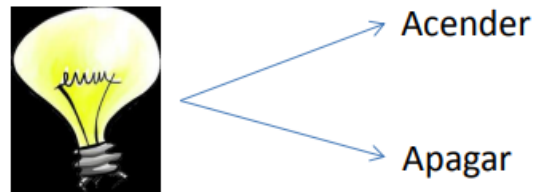
- Estado
 - Representado pelos valores dos atributos de um objeto
- Comportamento
 - Definido pelo conjunto de métodos do objeto
 - Estado representa o resultado cumulativo de seu comportamento
- Identidade
 - Um objeto é único, mesmo que o seu estado seja idêntico ao de outro;
 - Seu valor de referência
- Os valores dos DADOS são modificados a partir das OPERAÇÕES sobre estes dados

Objetos - Propriedades

- Estado



- Comportamento



- Identidade



L7

L8



Classes

- São especificações para objetos;
 - Representam um conjunto de objetos que compartilham características e comportamentos comuns.
 - Todo carro tem em comum:
 - Característica
 - Cor
 - Pneu
 - Direção
 - Comportamento
 - Dirigir
 - Frear

Classes

- Classe Lampada
- Atributos
 - potencia, ligada
- Métodos
 - ligar, desligar, EstaLigada

Lampada
- ligada : boolean - potencia : double
+ ligar() : void + desligar() : void + estaLigada() : boolean

Classes

- Nome da classe

► Atributos

► Métodos

Lampada
- ligada : boolean - potencia : double
+ ligar() : void + desligar() : void + estaLigada() : boolean

+	Pública – O atributo ou método pode ser usado por qualquer objeto.
#	Protegida – O atributo ou método pode ser usado por qualquer objeto da classe e também por suas subclasses.
~	Pacote – O atributo ou método é visível por qualquer objeto dentro do pacote.
-	Privada – O atributo ou método é visível somente pela classe que o define.

Classes em C#

- Declaração de uma classe em C#
 - A palavra-chave `class` é utilizada para definir uma nova classe
 - A classe `Lâmpada`, por exemplo, representa uma entidade lâmpada
 - Cada lâmpada armazena o seu estado (ligado ou desligado) e realiza as operações `ligar` e `desligar`

Classes em C#

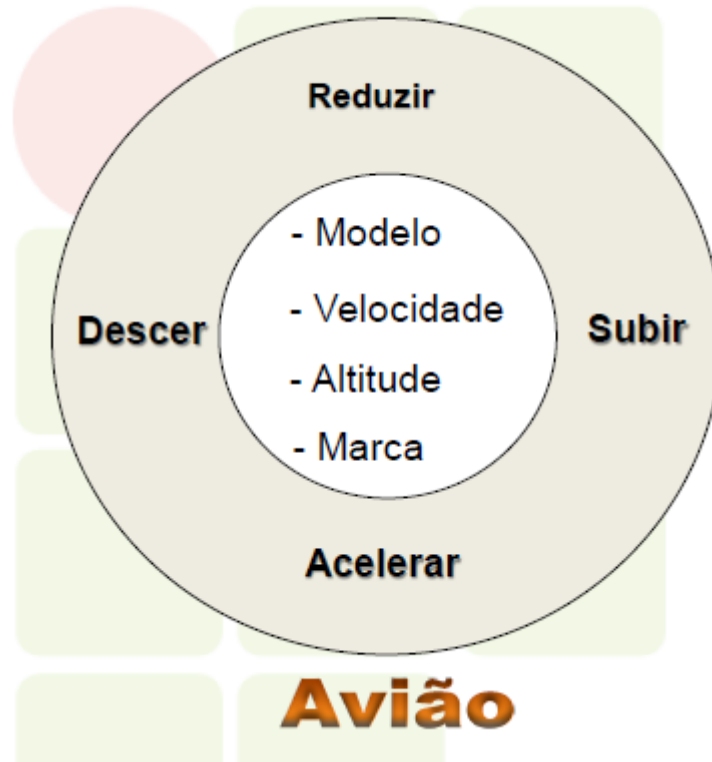
- Declaração de uma classe em C#

```
1 public class Lampada
2 {
3     private bool ligado;
4     public void ligar() { ligado= true; }
5     public void desligar() { ligado=false; }
6     public bool estaLigada(){return ligado;}
7 }
```

Resumo

- Objeto
 - Qualquer entidade que possui características e comportamento
- Classe
 - Descreve um tipo de objeto
 - Define atributos e métodos
- Atributo
 - Define características do objeto
- Método
 - Operações que o objeto pode realizar

Exercício 1



Exercício 2

- Imagine uma lâmpada que possa ter três estados: apagada, acesa e meia-luz.
- Usando a classe `Lampada`, vista nas transparências, escreva uma classe em C# para essa nova lâmpada.

Construtor

- Um método construtor pode ser utilizado para inicializar um objeto de uma classe.
- *Por padrão, o compilador fornece um construtor-padrão sem parâmetros em qualquer classe que não inclua explicitamente um construtor.*

Construtor

- A palavra-chave `new` chama o construtor da classe para realizar a inicialização
- Ou seja, quando uma classe é instanciada, o primeiro método a ser executado é o construtor daquela classe..

Construtor

- A chamada de um construtor é indicada pelo nome da classe seguido por parênteses.
- O nome do construtor deve ser igual ao nome da classe

Exercício 3

- Crie uma classe Livro que represente os dados básicos de um livro.

Exercício 4

- Escreva uma classe Pessoa contendo todos os atributos de uma pessoa. Faça métodos para apresentar os dados.

Exercício 5

- Crie uma classe chamada Disciplina que contenha os atributos nome da disciplina e carga horária. Tanto o nome da disciplina quanto a carga horária são definidos pelo construtor da própria classe. Além do construtor, crie um método para exibir esses dados após a sua atribuição. O método para exibir os dados deverá ser chamado em uma outra classe.

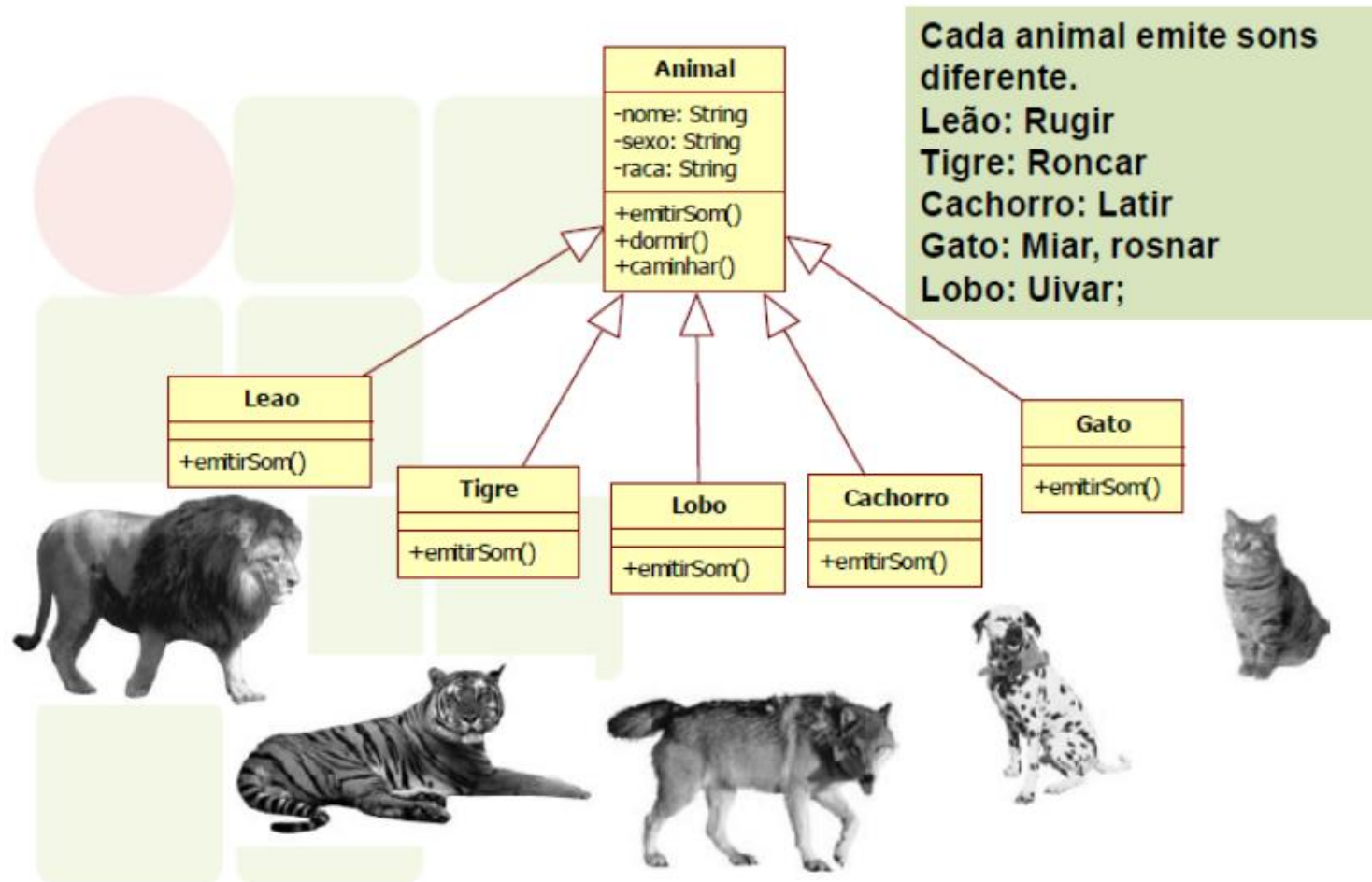
Exercício 6

- Crie uma classe chamada Motor que possua três atributos visíveis apenas na própria classe, são eles: nomeFabricante, potência e tipo.
- Também deverão ser criados dois métodos visíveis a qualquer classe, sendo que um desses métodos deverá associar valores aos atributos e outro método deverá retornar o conteúdo desses atributos. Em uma outra classe deverá ser instanciada a classe Motor e apresentado na tela os valores dos atributos retornados pelo método correspondente da classe Motor.

Exercício 7

- Criar uma classe Carro e declare os seguintes atributos na classe: Modelo, Cor, Ano Marca, Chassi, Proprietário, Velocidade máxima, Velocidade atual, Nr de portas, tem teto solar? Nr Marchas, tem cambio automático? Volume de combustível
- Faça o encapsulamento da classe Carro e seus atributos
- Implemente o método acelera que aumenta a velocidade de 1 em 1 km/h e não ultrapassando o limite máximo.
- Implemente o método freia que para o carro – Velocidade = 0 km/h
- Implemente o método troca marcha
- Implemente o método reduz a marcha;

Exercício 8

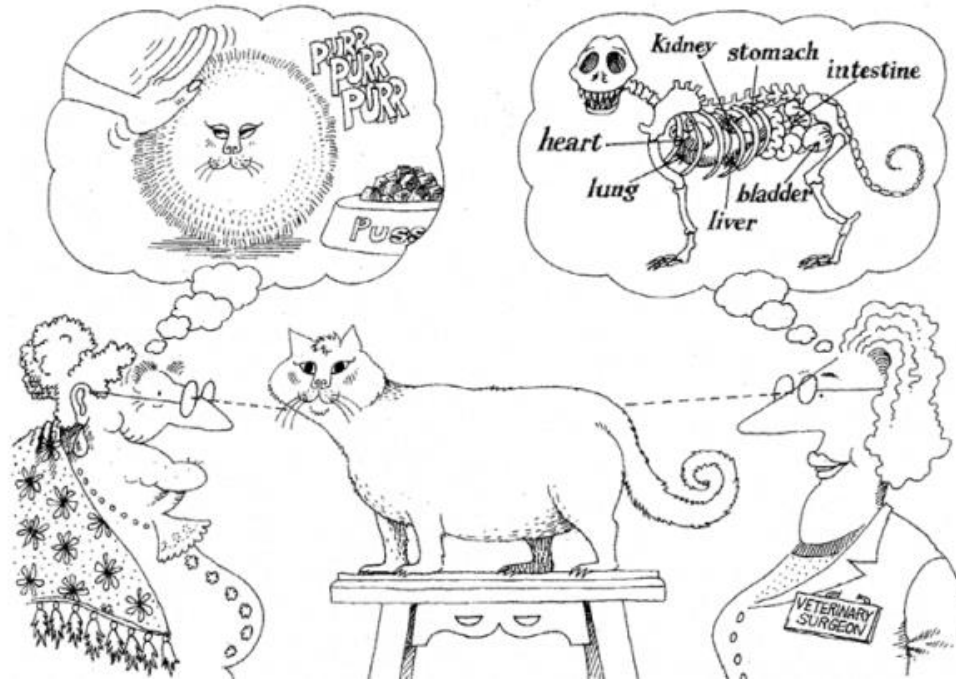


Abstração

- Abstração é uma das formas fundamentais que nós lidamos com a complexidade;
- Quando queremos diminuir a complexidade de alguma coisa, ignoramos detalhes sobre as partes para concentrar a atenção no nível mais alto de um problema;
- Não se analisa o “todo”, em POO é importante analisar as partes para entender o todo.

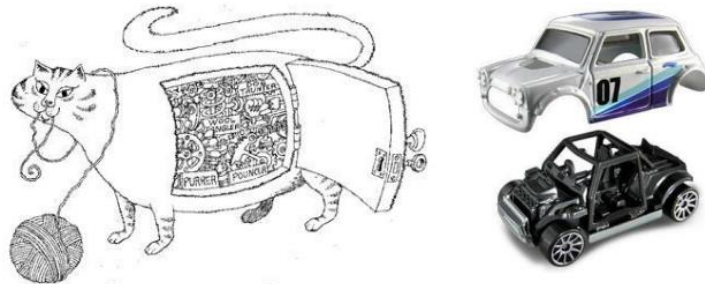
Abstração

- Foca a característica essencial de alguns objetos relativo a perspectiva do visualizador



Encapsulamento

- Encapsulamento é o processo de esconder todos os detalhes de um objeto que não contribuem para as suas características essenciais;
- O encapsulamento é o modo de dar ao objeto seu comportamento “caixa-preta”, que é o segredo da reutilização e confiabilidade.
- Se o estado de um objeto foi modificado sem uma chamada de método desse objeto, então o encapsulamento foi quebrado.



Encapsulamento e Abstração

- São conceitos complementares
 - Abstração foca sobre o comportamento observável de um objeto, enquanto encapsulamento se concentra na execução que dá origem a esse comportamento

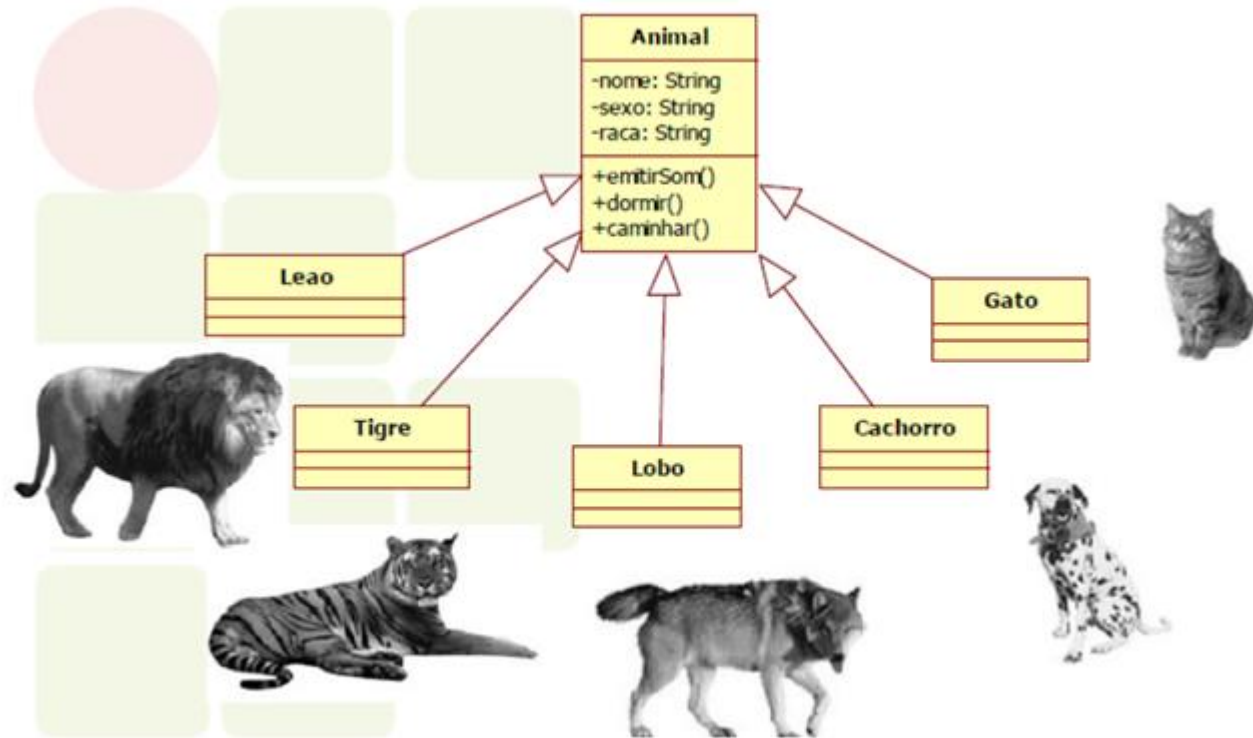
Herança

- A abstração ajuda a diminuir a complexidade.
- Encapsulamento ajuda a gerenciar essa complexidade, ocultando a visão dentro de nossa abstrações.
- A modularidade também ajuda, dando-nos uma maneira de agrupar logicamente abstrações relacionadas.
- Um conjunto de abstrações, muitas vezes forma uma hierarquia, e identificando essas hierarquias no nosso projeto, simplifica grandemente o nossa compreensão do problema.

Herança

- Herança é o mecanismo para expressar a similaridade entre Classes, simplificando a definição de classes iguais que já foram definidas.
- O que um leão, um tigre, um gato, um lobo e um dálmatas têm em comum?
- Como eles são relacionados?

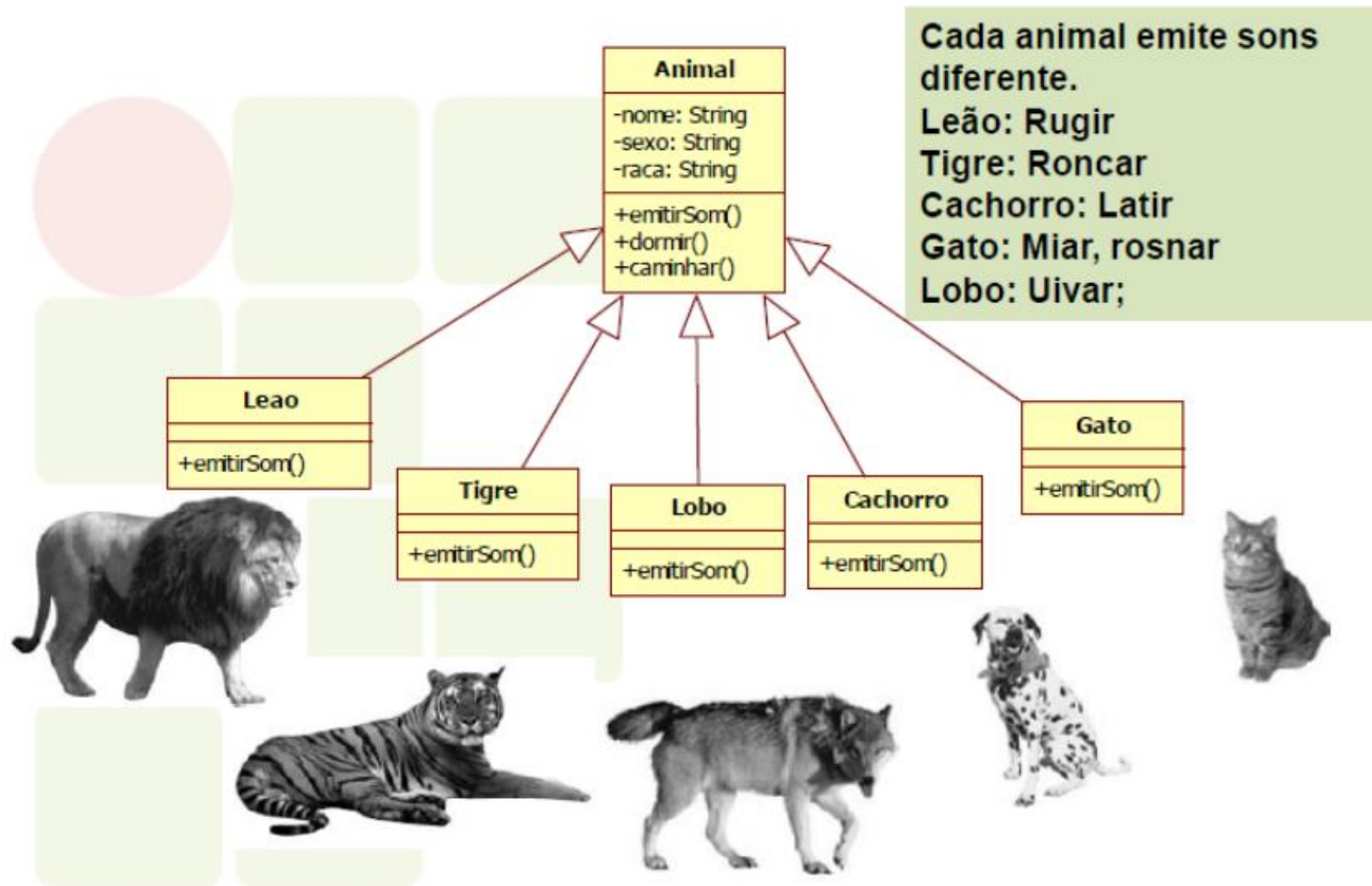
Herança



Polimorfismo

- Polimorfismos
 - Poli -> varias; Morfos -> formas;
- Significa que um objeto pode assumir diferentes formas;
- O conceito de polimorfismo está associado a
- Herança;
- É caracterizado como o fato de uma operação poder ser implementada de diferentes maneiras pelas classes na hierarquia.

Polimorfismo



Visibilidade

- Private
 - O nível de acesso se restringe apenas a classe;
 - Não é passado por herança;
- Public
 - O nível de acesso é irrestrito;
 - Por padrão, é a visibilidade definida para métodos e atributos em uma classe
- Protected
 - É visível em toda a classe;
 - É passado por herança (mesmo em pacotes diferentes);
- Internal
 - Com este modificador, o acesso é limitado apenas ao assembly atual.
- Protected Internal
 - Com este modificador, o acesso é limitado ao assembly atual e aos tipos derivados da classe que contém o modificador.

Modificadores de acesso (visibilidade)

Carro
+modelo +ano de fabricação -cor #estado
-Ligar() -Desligar()

- _ **Proteção dos dados**
 - _ Público (+)
 - _ Privado (-)
 - _ Protegido (#)

Métodos e classes abstratas

- Classe que contém um método abstract deve ser declarada abstract
- Abstract não pode ser utilizado em:
 - Construtor
 - Métodos estáticos
- Classes abstratas não podem ser instanciadas
- Classe que estende uma classe abstract:
 - Deve implementar todos os métodos abstract
 - Caso algum método não seja implementado, a classe deve ser declarada abstract

Interface

- Uma interface é uma coleção de operações utilizadas para especificar um serviço de uma classe ou de um componente.
- As interfaces são empregadas para visualizar, especificar, construir e documentar a coesão interna do sistema.
- Uma interface bem-estruturada fornece uma clara separação entre a visão externa e a visão interna de uma abstração, tornando possível que se compreenda e se tenha acesso a uma abstração sem ser preciso se aprofundar nos detalhes de sua implementação