Programação Orientada a Objeto

Contexto Histórico

A *crise* do desenvolvimento do *software* (década 60/início 70):

- Demanda muito superior à capacidade de desenvolvimento;
- Qualidade insuficiente dos produtos;
- Estimativas de custo e tempo raramente cumpridas nos projetos.

Visando

- Melhorar a qualidade dos produtos de software produzidos e
- Aumentar a produtividade no processo de desenvolvimento
  - Surge a *Engenharia de Software*;

Engenharia de Software

- Engenharia de software é o mesmo que Processo de desenvolvimento de software

Busca organizar esforços no

desenvolvimento de ferramentas,

metodologias e ambientes de suporte ao

desenvolvimento de software

Paradigma de Desenvolvimento de Software

Para aplicar o processo de desenvolvimento é necessário ter:

- Ferramentas, metodologias e ambientes;
  - Este elementos/fatores dependem do Paradigma de Desenvolvimento utilizado.

Mas o que é um paradigma?

- Paradigma é a “filosofia” adotada na construção do software, ou seja,
  - É o estilo ou padrão ou modelo de programação:
  - É um padrão de pensamento para a solução de problemas;
- Há vários paradigmas de programação, aplicados dependendo do domínio do problema que se quer resolver:
  - Os principais paradigmas existente são:
    - Lógico, estruturado ou imperativo, orientado a objetos, funcional, etc

Paradigma de Desenvolvimento de Software

Os dois paradigmas bastante utilizados para implementação de software são:

- Estruturado ou imperativo:
  - Representado por linguagens como C e Pascal.
- Orientado a objetos:
  - Representado por linguagens como Java, C++, C#.
    - Vamos analisá-los.....

O Paradigma Imperativo ou Estruturado

O Paradigma Imperativo ou Estruturado

O paradigma estruturado leva em consideração que o mais importante no programa é o algoritmo, ou seja o conjunto de ações.

- Programas centrados no conceito de um estado (modelado por variáveis) e ações (comandos) que manipulam o estado

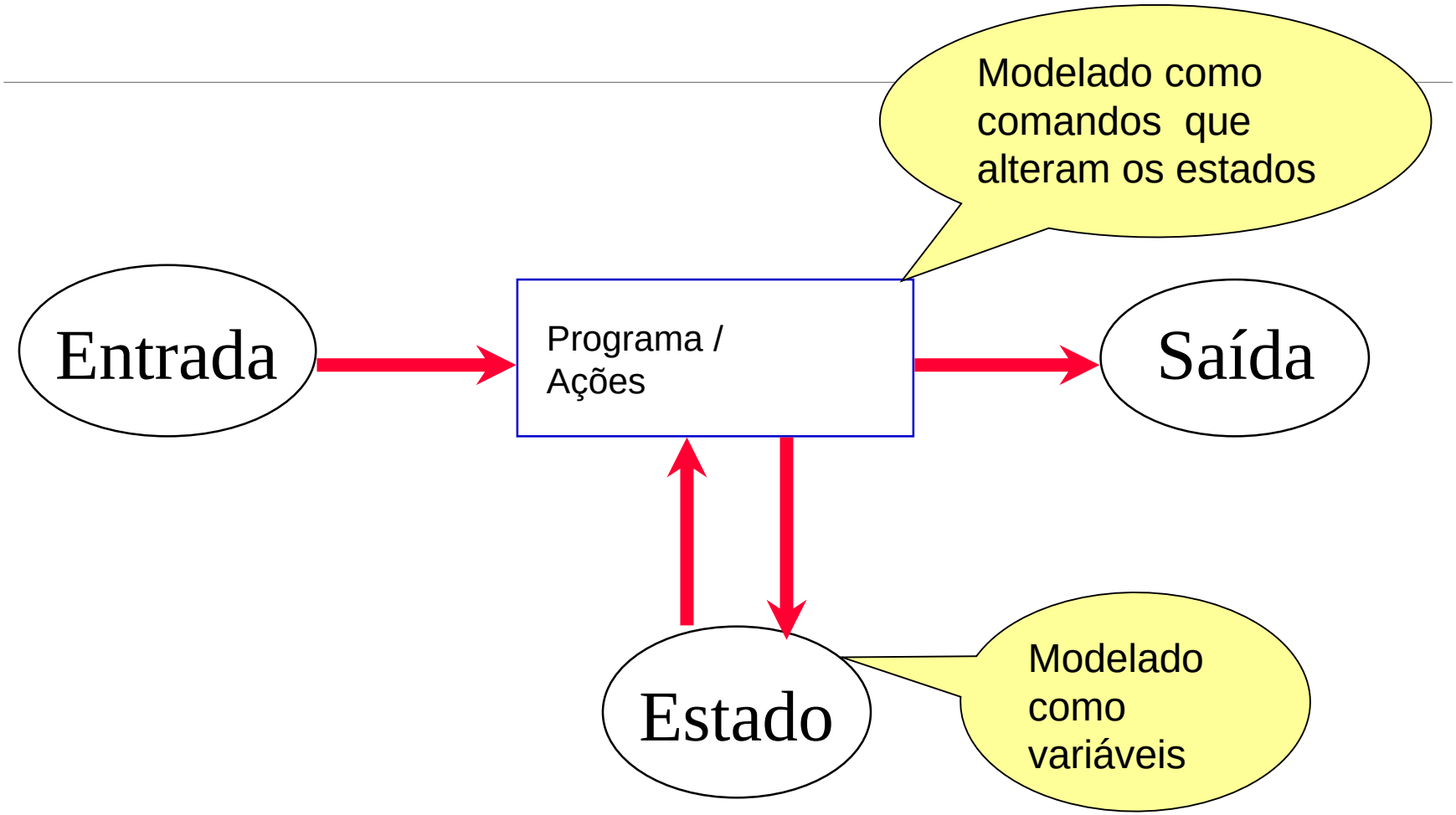
Um algoritmo é visto como uma seqüência de procedimentos que manipulam dados.

- Este paradigma também é denominado de procedural, por incluir subrotinas ou procedimentos como mecanismo de estruturação

Está preocupado nas ações ou funções que fazem parte do problema.

- O QUÊ e COMO o sistema deve fazer

O Paradigma Imperativo ou Estruturado



• Linguagens Imperativas: FORTRAN, COBOL, ALGOL 60, APL, BASIC, PL/I, SIMULA 67, ALGOL 68, PASCAL, C, MODULA 2, ADA

Modelagem no Paradigma Imperativo ou estruturado

Considere o seguinte problema:

- Implementar um sistema de matrícula de alunos em uma universidade

Poderíamos modelar um sistema de matrícula nas seguintes tarefas:

- Cadastrar aluno
- Cadastrar disciplina
- Cadastrar nota do bimestre
- Gerar histórico de aluno
- Matricular aluno
  - Matricular aluno em disciplina
  - Verificar validade da matricula
- Cadastrar professor em disciplina
- Cadastrar horário do bimestre
- Modificar horário do bimestre

O Paradigma Imperativo ou Estruturado

Vamos agora ver alguns conceitos do paradigma estruturado que servem de base para a criação da OO

- Aqui vamos ver vários conceitos aplicados no paradigma estruturado que *são naturalmente utilizados e incorporados* na POO

Tudo começa com a necessidade de abstração da codificação...

- E no começo haviam os Bits...
  - Inicialmente os programadores manipulavam bits individuais
    - 00100110011100010101010
  - Então foi inventado a linguagem *Assembly*
    - Os programadores passaram a escrever instruções equivalentes a alguns *bytes*
- Início da abstração dos dados, facilitando o desenvolvimento
  - Queremos mais.... Vamos ver mais formas de abstrações ☺

O Paradigma Imperativo ou Estruturado

Procedimentos e Funções

- Procedimentos e funções foram as primeiras facilidades reais de abstração oferecidas aos programadores
  - O Assembly ainda deixava a desejar

```
inteiro maior(inteiro a, inteiro b) {
        se (a > b)
                retorne a;
        senão
                retorne b;
}
```

- Um procedimento agrupa, sob um nome, uma sequência de comandos que realizam uma dada função
  - É como se fosse uma extensão da linguagem

- Na hora da criação do executável,
  - O procedimento é substituído pelas linhas de código aonde ele foi chamado.

O Paradigma Imperativo ou Estruturado

Os módulos

- A possibilidade de particionar um programa em *módulos* combináveis através de um *ligador* permitiu
  - Não só agrupar num módulo funções afins
  - Como também a *reutilização* do mesmo

```
Módulo estatístico

 media()
 mediana()
 desvio_padrão()
 variancia()
 correlacao()
```

Esse conceito permitiu a criação de bibliotecas de funções orientadas a aplicações específicas

O Paradigma Imperativo ou Estruturado

Composições de tipos

- Recursos para se definir dados através da composição de tipos existentes
- Esse é um conceito importante presente em todas as LP
  - Permite a composição de elementos a partir de elementos nativos da linguagens.

```
...
int vetor[100];
float matriz[10,10];
...
struct  funcionario {
    char [20] nome;
    char [20] sobrenome;
    char [30] endereco;
    char sexo;
    int salario;
}
```

- A gente vai ver *quais são e como* os relacionamentos são tratados na POO. Como a Composição é tratada em OO....

O Paradigma Imperativo ou Estruturado

Procedimento Formal

- Procedimento formal é um conceito que permite a chamada indireta a um procedimento
  - Essa chamada é feita como parâmetro ou como variável do tipo procedimento
    - Em Java é chamado de reflexão (*Reflection*)

```
function Integral( function f(real):real
                     x1, x2:real;
                   ): real;
                  "calcula a integral de f(x) no
                   intervalo (x1-x2) "
...
w:=Integral(sin,0,Pi/2);
z:=Integral(sqrt,10.0,20.0);
...
```

O Paradigma Imperativo ou Estruturado

Tipos Abstratos de Dados (ADT)

- Tipos abstratos de dados é um conceito que “agrupa”
  - Tipos definidos pelo usuário
  - E as operações sobre os mesmos
- Normalmente o “agrupamento” é feito num módulo
  - Conceito de agrupamento é bastante importante para a OO.
    - O conceito de ADT são diretamente aplicados em OO

```
type funcionario = record ... end;
                     ...
procedure NovoFuncionario(...); ...
procedure CadastraFuncionario(...); ...
procedure AlteraCadastro(...); ...
....
....
```

Tipos abstratos de dados permitem tratar de forma unificada *dados* e *código*.

O Paradigma Imperativo ou Estruturado

- O problema das pilhas

Imagine se tivéssemos que modelar uma pilha sem os procedimentos. Cada vez que tivéssemos que empilhar ou desempilhar um elemento teríamos que escrever o mesmo código

```
int pilhaDeDados[100];
int TopoDosDados = 0;

void init()      // inicialize uma pilha
{ TopoDosDados = 0; }

void push(int val)    // push a value on to the stack
{ if (TopoDosDados < 100)
              pilhaDeDados [TopoDosDados++] = val; }

int top()    // get the top of the stack
{ if (TopoDosDados > 0)
              return pilhaDeDados [TopoDosDados - 1];
       return 0; }

int pop()    // pop element from the stack
{ if (TopoDosDados > 0)
              return pilhaDeDados [--TopoDosDados];
       return 0; }
```

O Paradigma Imperativo ou Estruturado

Procedimentos Genéricos

- Procedimentos formais permitem, ainda que de forma limitada, a construção de procedimentos genéricos
  - Ideia base para o desenvolvimento de *Templates* (C++)

```
void ordena( void (* entrada) (void *),
             void (* saida)   (void *),
             int  (* compara) (void *, void *)
           )
{ /* ordena um conjunto de dados */ }
```

- Com procedimento genéricos é possível definir uma pilha genérica
  - A partir daí, pode-se usar o comportamento natural de uma pilha para trabalhar com qualquer elemento ou tipos diferentes
    - Mensagens, carros, clientes, int, string, float, etc

O Paradigma Imperativo ou Estruturado

IMPORTANTE: Abstração em Programação

- Por trás dos conceitos que acabamos de ver, está a preocupação com:
  - Aproximação a situações do “mundo real”
  - Reutilização de código
  - Aplicação de uma mesma solução a diversas situações diferentes
- Tais necessidades são naturalmente supridas na POO
  - Vamos começar a entender esse paradigma...

Paradigma Orientado a Objetos

Paradigma Orientado a Objetos

Mudanças nos paradigmas de programação

- Contexto do surgimento
  - Com o desenvolvimento e popularização da Engenharia de Software
  - E o aumento da complexidade dos softwares,

- Surgiu a necessidade de criação de um novo paradigma de programação que permitisse:
  - Modelar o mundo real
  - Implementar abstrações com maior consistência
  - Reutilizar software
  - Produzir software mais robusto
  - Oferecer mais facilidade de uso para as características avançadas do paradigma estruturado

- O paradigma tem as suas raízes nos conceitos do Simula 67
  - Smalltalk - puramente orientada

Paradigma Orientado a Objetos

Surge o paradigma orientado a objetos oferecendo soluções com:

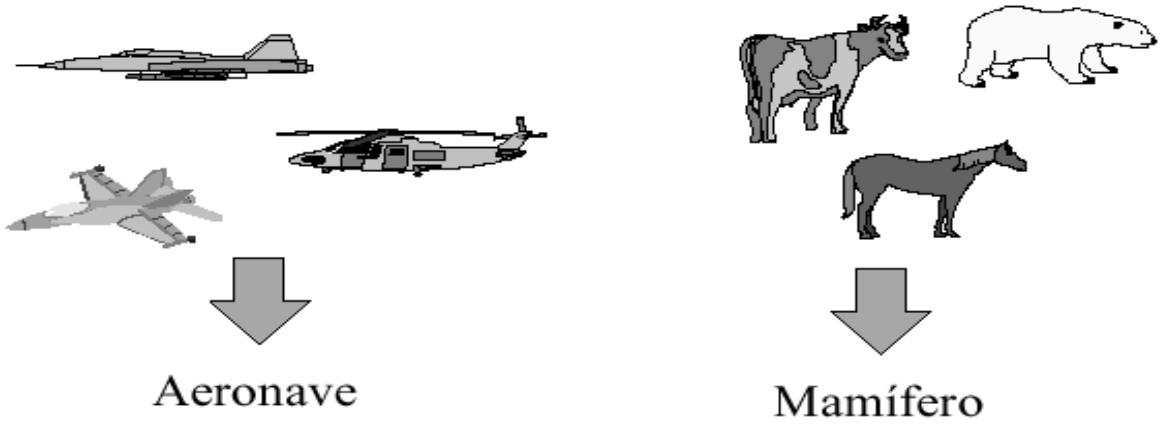
- Abstração e Encapsulamento
- Modularidade e Hierarquia
- Reutilização
- Polimorfismo
- Vinculação Dinâmica

  - Possibilidade de determinar os métodos de acordo com o objeto que a chama.
  - Útil numa hierarquia de classes para saber qual o método (dentro da hierarquia) que está sendo chamado.
    - Vamos entender esses conceitos importantes...

Paradigma Orientado a Objetos

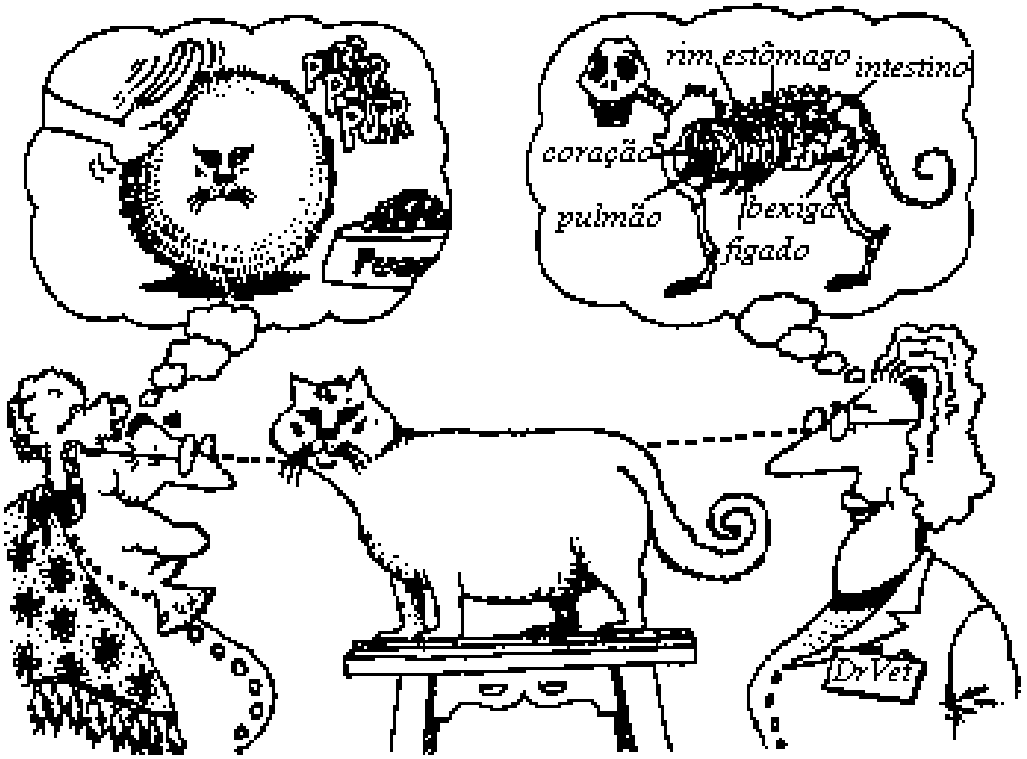
Conceito de Abstração

- Abstração ressalta:
  - As características essenciais de um objeto que o distinguem de outros tipos de objetos,



  - Provê limites conceituais entre objetos, que dependem da perspectiva do observador
    - Idéia principal: Focalizar o essencial, ignorar propriedades acidentais
      - A abstração é a representação de uma entidade que inclui somente os atributos de importância em um contexto particular
      - Simplifica o processo de programação

Paradigma Orientado a Objetos



A abstração enfoca as características essenciais de um objeto [Booch94].

Paradigma Orientado a Objetos

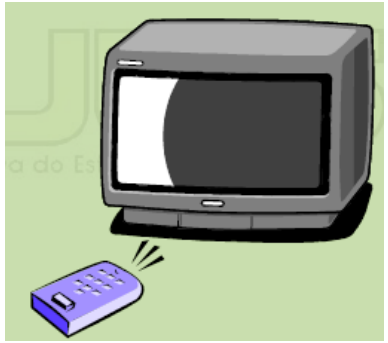
Conceito de Encapsulamento

- Processo de “agrupamento” dos elementos de uma abstração:
  - Que constituem sua estrutura e comportamento;
- Encapsulamento serve para separar:
  - A interface contratual de uma abstração de sua implementação
- Tem o poder de esconder a informação:
  - Reduzindo desse modo a quantidade de detalhes que necessitam ser lembrados,
    - ou seja, trocados entre programadores.

Paradigma Orientado a Objetos

Conceito de Encapsulamento

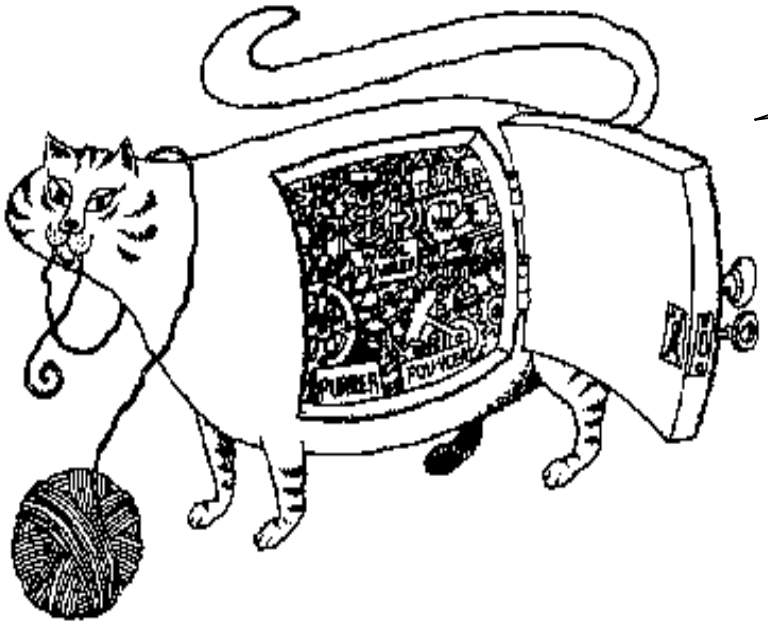
- Não estou preocupando em “como tal coisa está implementada”, mas em “como posso usá-la”.
  - A TV encapsula:
    - Mudar de canais
      - Não preciso me preocupar com o eletróns dentro do tubo de raios catódicos (implementação), basta apenas saber que preciso apertar o botão (interface)
    - Aumentar/Diminuir o volume



Paradigma Orientado a Objetos

Resumo - Conceito de Encapsulamento

- Usamos objetos sem saber seu funcionamento interno;
- Assim também deve ser em nossos sistemas OO:
  - Maior manutenibilidade;
  - Maior reusabilidade.



O encapsulamento oculta os detalhes de implementação de um objeto [Booch94].

- Abstração e encapsulamento são conceitos complementares:
  - enquanto a abstração enfoca o comportamento observável de um objeto,
  - o encapsulamento enfoca a implementação que origina esse comportamento.

Paradigma Orientado a Objetos

Conceito de Modularidade

- Propriedade de um sistema que foi decomposto em um conjunto de módulos coesos e acoplados
  - Separação por partes do problema

Conceito de Hierarquia

- É a organização/ordenação de abstrações.
  - Generalização / Especilização

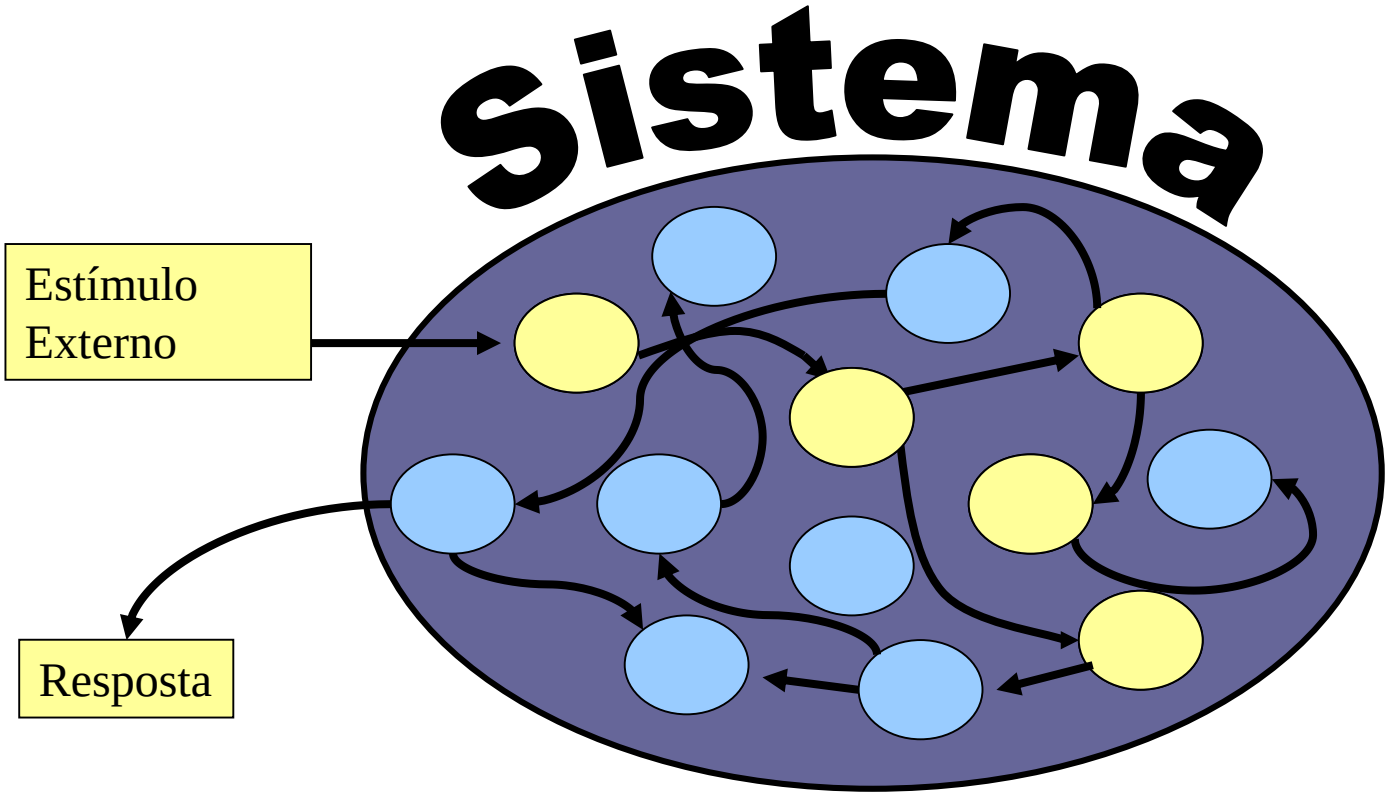
Conceito de Reutilização

- Necessidade adicional, quando estamos falando de software.
  - Todos os elementos anteriores são compostos de maneira a prover reutilização
    - Ex: construir minha própria janela, meus próprios botões, minhas entradas e modificar apenas parte do comportamento

Paradigma Orientado a Objetos

Dinâmica de um Sistema Orientado a Objetos

- Um sistemas OO é uma:
  - Coleção de objetos se comunicando, cooperando para atingir um resultado



Modelo Computacional do Paradigma Orientado a Objetos

- Dinâmica de um Sistema Orientado a Objetos
  - Uma visão mais detalhada...
    - As ações (funções/método) necessários e os estados possíveis (variáveis) é o digrama 1.
    - Na modelagem OO é observado os elementos que participam e se interagem das ações. (diagrama 2)
      - Forma mais natural de ver o mundo.

Diagrama 1

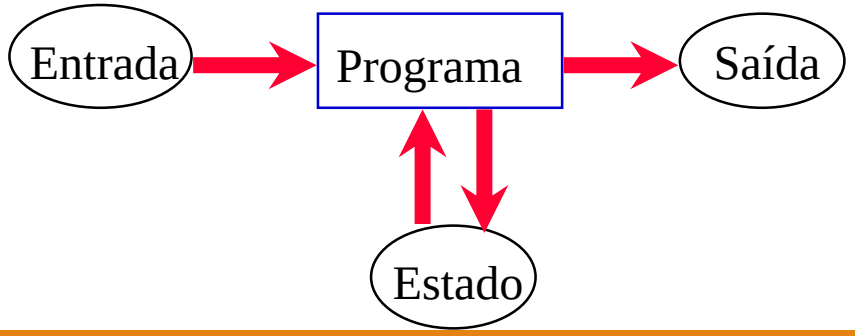
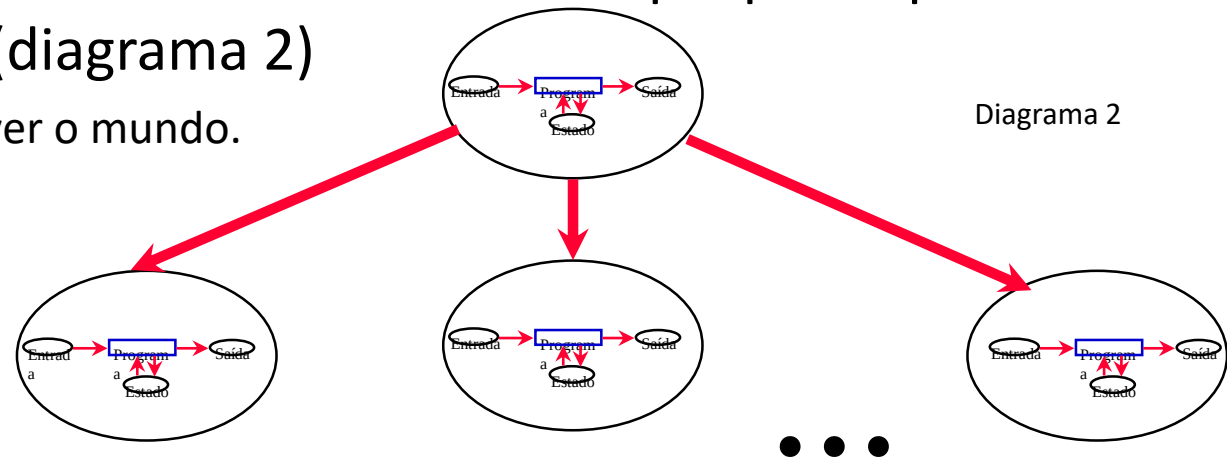


Diagrama 2



As ações e estados encapsulados pelos objetos que se comunicam

Modelo Computacional do Paradigma Orientado a Objetos

Poderíamos modelar o sistemas de matrícula em uma linguagem OO da seguinte forma:

- Aluno
- Professor
- Disciplina
- Curso
- Histórico
- ResultadoDisciplina
- Matrícula

Após isso, nós iríamos nos preocupar no relacionamento entre eles (ações e mensagens trocadas)

Cada uma das entidades anteriores potencialmente pode virar um tipo para a nossa solução, o que implica em:

- Identificar e implementar o conjunto de operações que permitam manipular aquele tipo em questão. (como se fosse um tipo abstrato de dados)

Paradigma OO

Resumindo 1:

O paradigma OO utiliza uma perspectiva mais

humana de observação da realidade,

incluindo objetos, classificação e

compreensão hierárquica

Paradigma OO

Resumo 2:

- Conceitos OO fundamentais
  - Abstração:
    - Desprezar conceitos irrelevantes.
  - Encapsulamento:
    - Separar a interface da implementação.
  - Modularidade:
    - Agrupar objetos em módulos coesos independentes.
  - Hierarquia:
    - Organizar abstrações em hierarquias quando necessário.

Conceitos Básicos

Objetos

A perspectiva mais humano desse paradigma é que a solução....

- É composta por objetos que “que faz sentido no domínio da aplicação”

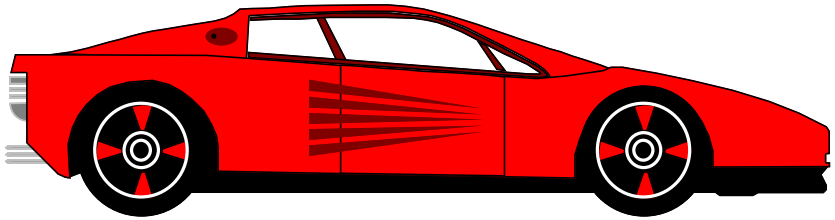
Mas esses objetos possuem que características???

- Estado,
- Comportamento e
- Identidade

Estado do Objeto

O Estado do Objeto é conjunto de suas propriedades associadas a seus valores correntes

- Propriedades geralmente referenciadas como *Atributos*
  - Ex: A cor do carro



Propriedades:

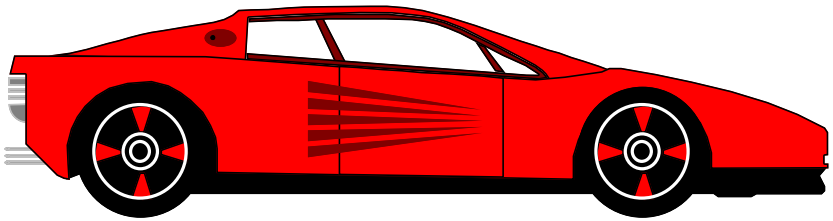
- Cor = Vermelha
- Ano = 2001
- Velocidade = 0 Km/h
- Combustível = Gasolina

- Em outras palavras.. É a característica do objeto...

Comportamento do Objeto

O Comportamento do Objeto é conjunto de serviços ou operações que outros objetos podem requisitar

- Operações geralmente referenciadas como *Métodos*
- Representa como o objeto reage às *Mensagens* a ele enviadas
  - Ex: Ações de um carro



Operações

- Acelerar ( )
- Frear ( )
- Acender Faróis ( )
- Virar a Direita ( )

- Em outras palavras.. É o que o objeto faz.... Quais são suas ações...

Comportamento do Objeto

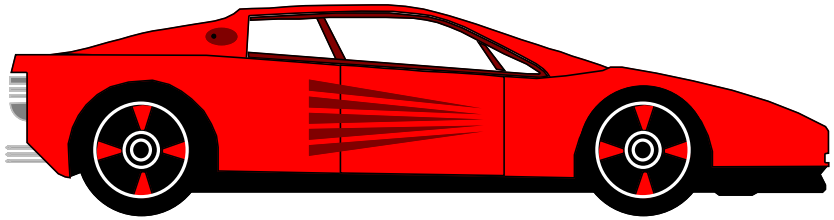
Há duas operações / métodos especiais em um objeto:

- Construtores
  - Indicam como as propriedades devem ser inicializadas quando um objeto é criado.
- Destruidores
  - Indicam as ações que devem ser realizadas quando um objeto for destruído.
- Este métodos representam o ciclo de vida que existe para um objeto
  - Basta imaginar a vida de vocês...

Identidade Única

A Identidade Única dos Objetos indicam que eles têm existência própria

- São distintos mesmo se seus estados e comportamento forem iguais
- Identificador que permite referência inequívoca
  - Ex: O carro é do Vinicius e não do João.



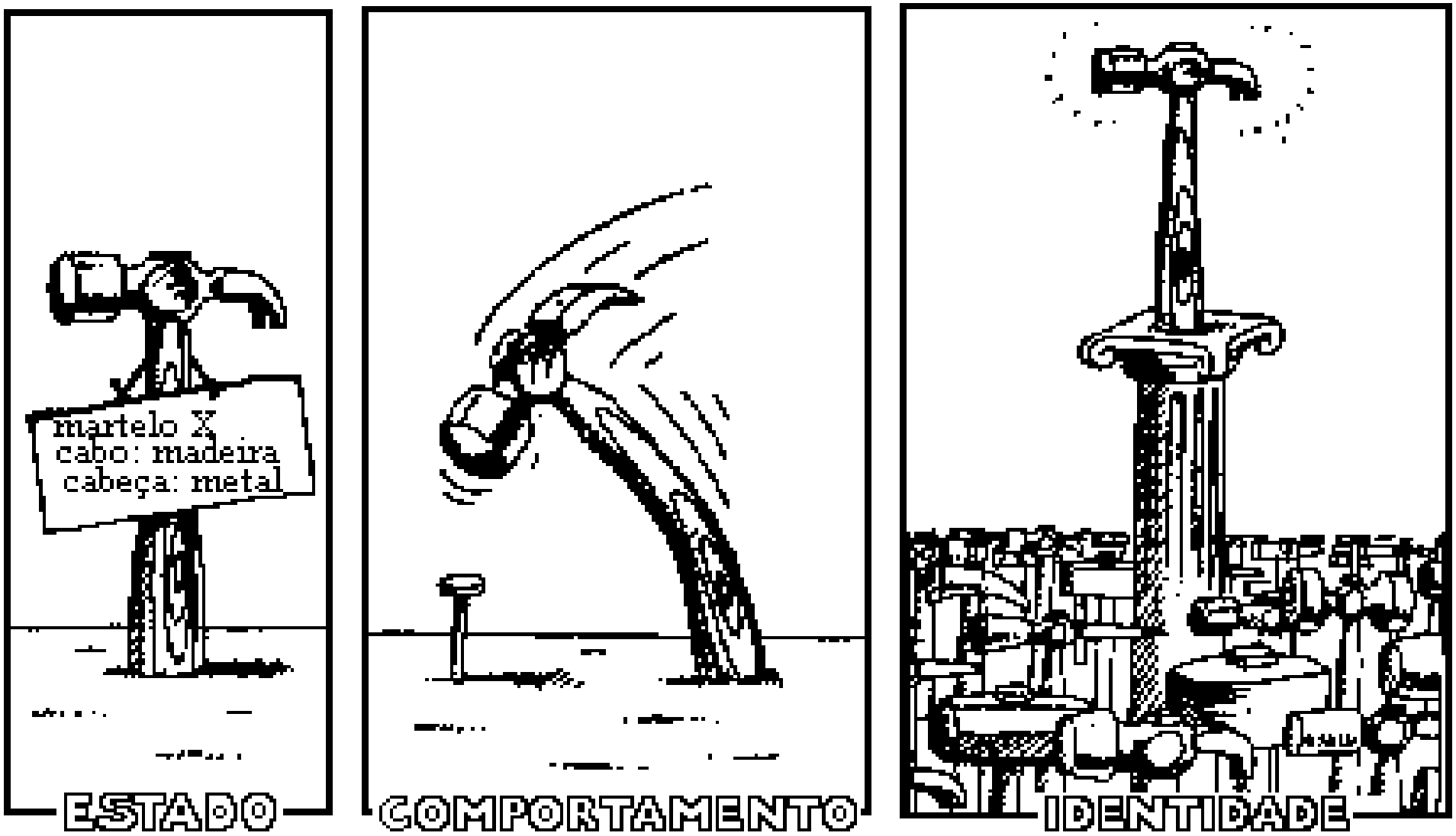
Carro (Ferrari) do Vinicius

É meu e ninguém “tasca”... :-)

- Esta característica representa a individualidade de cada elemento
  - Basta olhar para o lado e observar seus colegas....

Objetos

Resumo



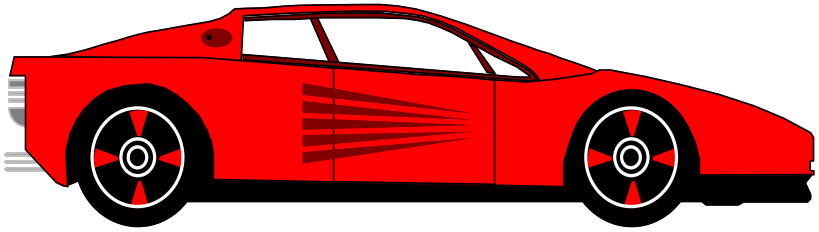
Descrição de um Objeto

Logo o objeto carro...

- ... pode ser descrito por um conjunto de atributos e comportamentos.....

Atributos

Motor
Cor
Potência
Rodas



Carro do Vinicius

Comportamentos

Avançar
Retroceder
Parar
Abastecer

substantivos

verbos

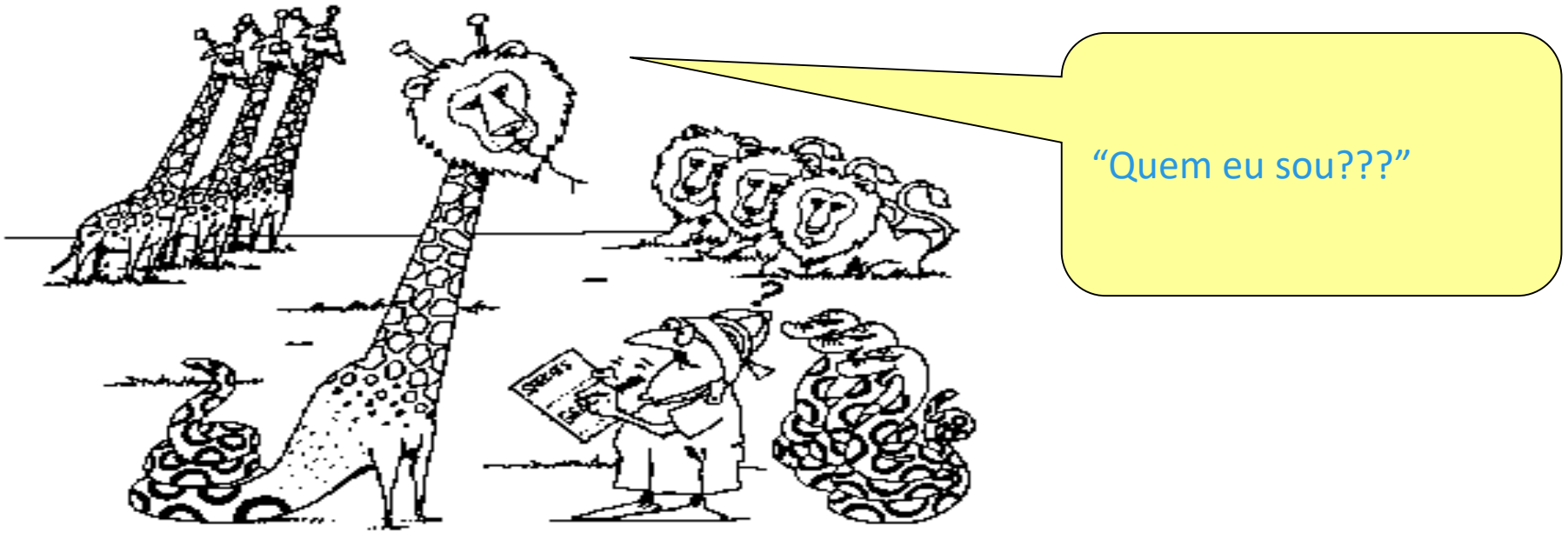
Classes

- Agora vamos falar sobre...
  - ....As classes em orientação a objetos..
    - Mas que raios é isso....

Classes

Conceito Principal

- A estrutura e o comportamento comuns dos objetos podem ser agrupados em classes.
  - "A classificação é o meio pelo qual ordenamos o conhecimento" (Glady Booch)



Classe de Objetos

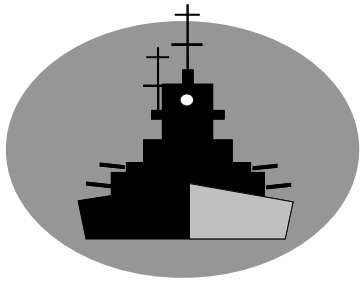
- Uma definição de classe então seria:

"Classe de Objetos é um grupo de objetos com os mesmos atributos

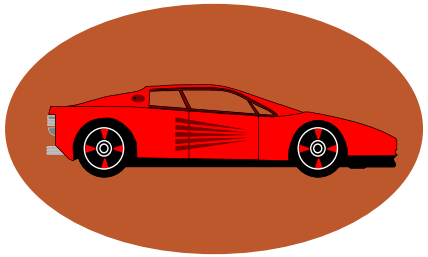
e os mesmos comportamentos

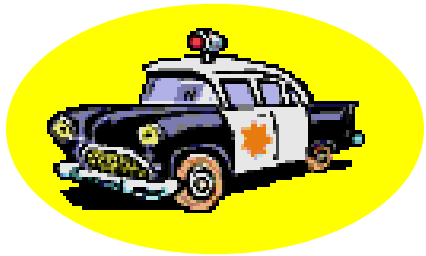
pertencem a mesma classe."

Transporte Marítimo



Transporte Terrestre





Classes

Em outras palavras: Uma classe armazena:

- As propriedades (estado) e
- As operações (comportamento) que são comuns a um conjunto de objetos.

Classe = “Molde de Objetos”

- Define Estrutura e Comportamento (notação UML)

Nome da Classe
Atributo1:tipo=valor Atributo2:tipo=valor
operação1(argumentos): tipo-retorno operação2(argumentos): tipo-retorno

Carro
Chassi: String Cor: Integer Motor: Motor
Ligar(): void TemGasolina(): boolean Virar(direção): void

Classes

Blz..

- Entendi o que é uma classe e o que é um objeto..
  - Mas qual a relação entre eles???....

Classes

Uma classe define um tipo, a partir do qual todos os objetos são criados.

- Definir uma classe não cria um objeto,
  - Assim como um tipo de variável NÃO é uma variável.

Um objeto é uma instância ou representação de uma classe.

- Em outras palavras, um objeto é um “indivíduo” de uma classe
  - É semelhante à alocação de uma variável em memória.

Com isso podemos verificar que o estado de um objeto é o conjunto de valores dos seus atributos.

- Um estado de um objeto só pode (ou deve!) ser modificado por meio de suas operações.

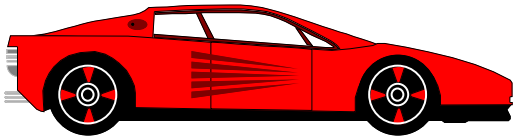
Instanciação de um Objeto

Tempo de Compilação

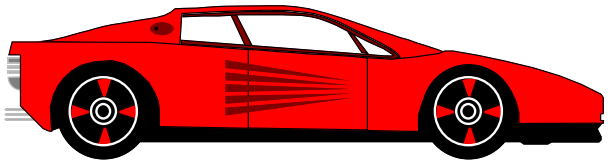
Tempo de Execução

Classe Carro
Chassi: String Cor: Integer Motor: Motor
Ligar(): void TemGasolina(): boolean Virar(direção): void

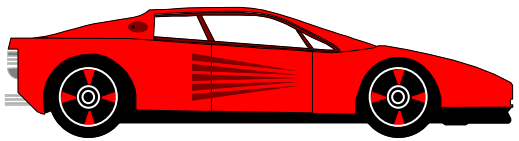
Instanciando Objetos



Objeto Carro:
Instância #1



Objeto Carro:
Instância #2



Objeto Carro:
Instância #3

Mensagens e Métodos

Blz....

- Mas e agora???...
  - Como é possível que um objeto converse com outro objeto nos sistemas OO???...

Mensagens e Métodos

Um objeto cliente só pode comunicar-se com outro através da emissão de mensagens

Essa emissão ocorre quando utilizamos algum método,

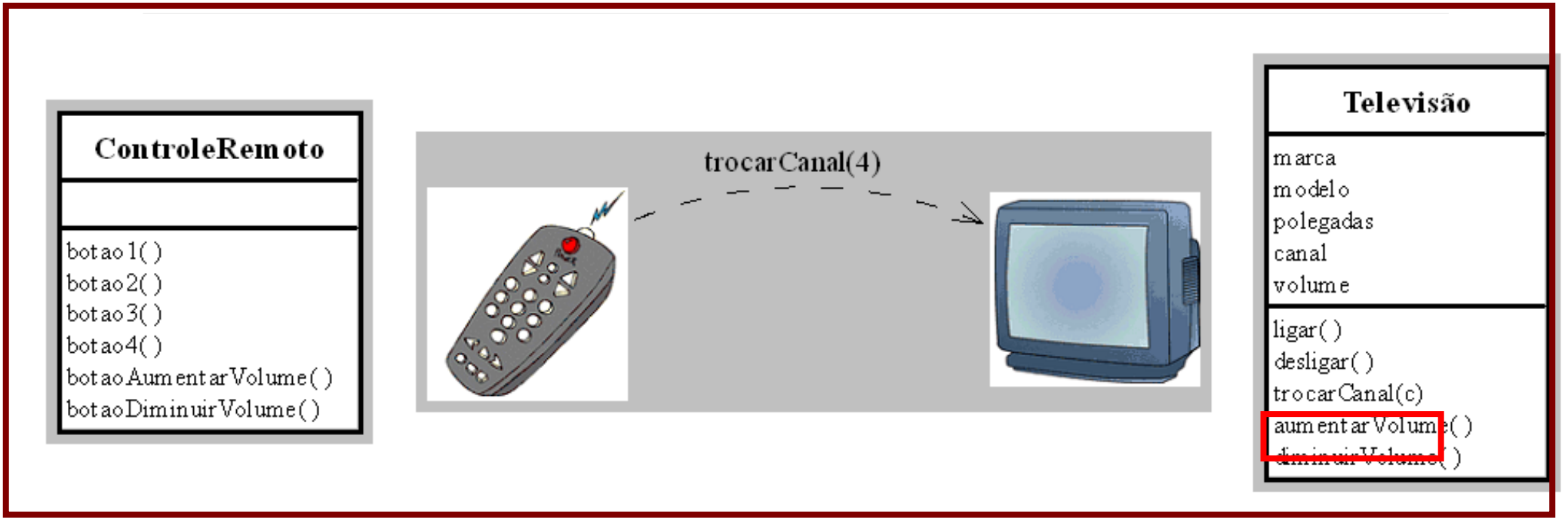
- Dizemos que ele enviou uma mensagem ao outro objeto

Além disso, estas mensagens podem conter alterações nas propriedades

- Que definem o estado de um objeto

Mensagens e Métodos

Classes e Objetos – Revisão:



- Ao ser pressionado o botão “4” do controle remoto,
  - Este envia uma mensagem para o objeto televisão,
  - Chamando a operação trocarCanal(4).
- Ou seja, o controle remoto invoca uma operação do objeto televisão.