Faculdade de Computação - FACOM

Bacharelado em Sistemas de Informação

FACOM32305 - Programação Orientada a Objetos

Prof. Thiago Pirola Ribeiro

Sumário

Objeto I

Objeto

Um elemento ou entidade do mundo real de um determinado domínio

Exemplos:

- *Objetos físicos*: livro, mesa, pessoa, mercadoria, etc.
- *Ocupações de pessoas*: cliente, vendedor, etc.
- *Eventos*: compra, telefonema, etc.
- *Lugares*: loja, cidade, etc.
- *Interações entre objetos*: item (de uma nota fiscal), parágrafo (em um sistema editor de texto), etc.

Objeto II

Objetos são instâncias de classes:

- As classes é que determinam quais informações o objeto contém, e como manipulá-las.

Objetos podem reter um estado (informação) e possuem operações para examinar ou alterar seu estado.

É através dos objetos que praticamente todo o processamento ocorre em sistemas OO

Objeto III

Exemplo: objeto cachorro

Características próprias, como:

- Um nome;
- Uma idade;
- Um comprimento de pêlos;
- Uma cor dos pêlos;
- Uma cor dos olhos;
- Um peso;
- ...

As características que descrevem um objeto são denominadas atributos.

Objeto IV

Exemplo: objeto cachorro

Pode executar ações, como:

- Latir;
- Correr;
- Sentar;
- Pegar uma bola;
- Comer;
- Dormir;
- ...

As ações que um objeto pode executar são denominadas métodos.

Objeto V

A única forma de interação com os objetos é através de seus métodos:

- Para alimentar o cachorro Lulu, usamos o método Comer
- Para brincar com ele, usamos o método Pegar uma Bola
- etc.

O conjunto de métodos disponíveis em um objeto é chamado interface.

Atributos

Como visto, objetos do mundo real têm *propriedades*:

- Essas propriedades recebem o nome de atributos. São como *variáveis* ou *campos* que armazenam os valores das características dos objetos

Exemplo (Cachorro):

- *Nome*: Lulu
- *Idade*: 2 anos
- *Comprimento de pêlos*: curto
- *Cor dos pêlos*: marrom
- *Cor dos olhos*: marrom
- *Peso*: 4 kg

Estado I

Estado de um objeto: conjunto de valores de seus atributos em um determinado instante

- Para haver mudança de valores, são necessários estímulos internos ou externos

Estado (caso anterior):

- *Nome*: Lulu
- *Idade*: 2 anos
- *Comprimento de pêlos*: curto
- *Cor dos pêlos*: marrom
- *Cor dos olhos*: marrom
- *Peso*: 4 kg

Estado II

Dois cachorros diferentes:

Estado

- *Nome*: Lulu
- *Idade*: 2 anos
- *Comprimento de pêlos*: curto
- *Cor dos pêlos*: marrom
- *Cor dos olhos*: marrom
- *Peso*: 4 kg

Estado

- *Nome*: Rex
- *Idade*: 4 anos
- *Comprimento de pêlos*: longo
- *Cor dos pêlos*: branco
- *Cor dos olhos*: preto
- *Peso*: 10 kg

Métodos

Métodos são os procedimentos ou funções que realizam as ações do objeto, ou seja, implementam as ações que o objeto pode realizar.

É por seus métodos que um objeto se manifesta e através deles que o objeto interage com outros objetos.

Comportamento

Comportamento de um objeto: como ele age e reage em termos de mudanças de estado e trocas de mensagens com outros objetos

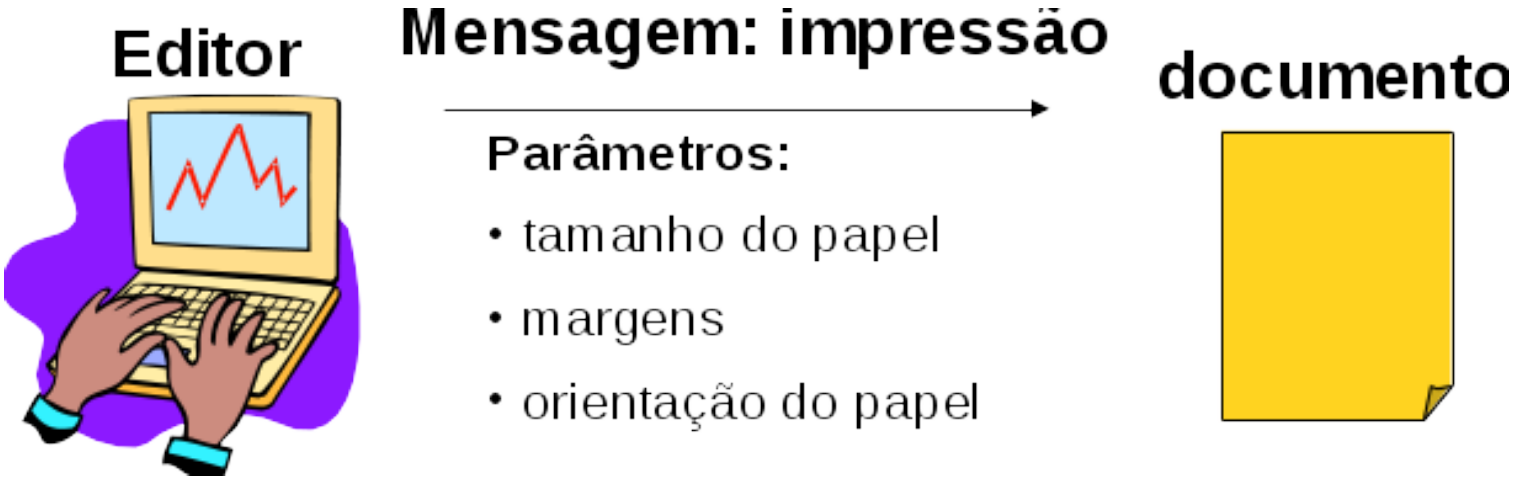
- Execução de uma operação (um método)
- Métodos são responsáveis pelas ações, que podem envolver acessar ou alterar os valores dos atributos

Se a requisição é feita pelo outro objeto, ela é enviada por uma mensagem.

- Mensagem é solicitação para que objeto execute um método.

Mensagem

Quem envia a mensagem não necessita saber como o receptor irá tratá-la: Deve apenas *conhecer o resultado final* do tratamento e *o que é necessário para obtê-lo*.



Objeto

Um objeto não é muito diferente de uma variável normal:

- Ex. quando se define uma variável do tipo `int` em Java, essa variável tem:
  - Um espaço em memória para registrar o seu estado (valor);
  - Um conjunto de operações que podem ser aplicadas a ela (operadores que podem ser aplicados a valores inteiros).
- Quando se cria um objeto, ele adquire:
  - Um espaço em memória para armazenar seu estado (valores de seus atributos);
  - Um conjunto de operações que podem ser aplicadas ao objeto (seus métodos).

Exemplo I

Identificar atributos e métodos em:

1. Uma tela de computador

Exemplo II

Identificar atributos e métodos em:

1. Uma tela de computador
   - Atributos:
     - `Modo de operação (texto, gráfico);`
     - `Tamanho horizontal;`
     - `Tamanho vertical;`
     - `Paleta de cores.`
   - Métodos:
     - `modo texto ( );`
     - `modo gráfico ( );`
     - `muda cor ( );`
     - `escreve caractere ( );`
     - `muda dimensões (x,y).`

Classes I

Uma classe representa um *conjunto de objetos* que possuem características e comportamentos comuns.

- Um objeto é uma *instância* de uma classe;
- Ou seja, criam-se objetos baseados no que é definido nas classes.

Ênfase na realidade deve ser nas classes.

Classes II

Exemplo: Cachorros podem ser descritos pelos mesmos atributos e comportamentos, pois são da mesma classe:

Cachorro_1

- *Nome*: Lulu
- *Idade*: 2 anos
- *Comprimento de pêlos*: curto
- *Cor dos pêlos*: marrom
- *Cor dos olhos*: marrom
- *Peso*: 4 kg

Cachorro_2

- *Nome*: Rex
- *Idade*: 4 anos
- *Comprimento de pêlos*: longo
- *Cor dos pêlos*: branco
- *Cor dos olhos*: preto
- *Peso*: 10 kg

Classes III

Classe Cachorro

Objetos da mesma classe possuem a mesma definição para métodos e atributos (embora valores sejam, em geral, diferentes).

Classes IV

Exemplo 2: Classe gato, formada por objetos "gato"

Algumas características:

- Nome
- Idade
- Comprimento dos pêlos
- Cor dos pêlos
- Peso

Algumas ações:

- Miar
- Comer
- Dormir
- Subir na árvore

Há atributos e métodos comuns entre cães e gatos. O que fazer? Criar a *super-classe* `Mamíferos`. Veremos mais a respeito quando estudarmos o conceito de herança.

Sumário

Palavras reservadas do Java:

`abstract`	`boolean`	`break`	`byte`
`case`	`catch`	`char`	`class`
`continue`	`default`	`do`	`double`
`else`	`extends`	`false`	`final`
`finally`	`float`	`for`	`if`
`implements`	`import`	`instanceof`	`int`
`interface`	`long`	`native`	`new`
`null`	`package`	`private`	`protected`
`public`	`return`	`short`	`static`
`super`	`switch`	`synchronized`	`this`
`throw`	`throws`	`transient`	`true`
`try`	`void`	`volatile`	`while`

Palavras reservadas, mas não usadas pelo Java: `const`, `goto`

Declarando classes I

Uma classe é declarada com a palavra reservada `class`, seguida do nome da classe e de seu corpo *entre chaves*.

```
1 <modificador de acesso> class NomeDaClasse
2 {
3     // declaracao dos atributos
4     // declaracao dos métodos
5 }
```

Declarando classes II

O nome da classe é também chamado de identificador.

Regras para nome de uma classe:

- Não pode ser uma palavra reservada do Java
- Deve ser iniciado por uma letra, ou _ ou $
- Não pode conter espaços

```
public class Pessoa
{
    //declaracao dos atributos
    //declaracao dos métodos
}
```

Declarando classes III

Convenção de estilo para nomes de classes (boa prática de programação):

- A palavra que forma o nome inicia com letra maiúscula, assim como palavras subsequentes:
  - Exemplos: `Lampada`, `ContaCorrente`, `RegistroAcademico`, `NotaFiscalDeSupermercado`, `Figura`, ...
- Devem preferencialmente ser substantivos
- Sugere-se que cada classe em Java seja gravada em um arquivo separado, cujo nome é o nome da classe seguido da extensão .java

Declarando classes IV

Ex.: classe `Pessoa` no arquivo Pessoa.java

```
1 public class Pessoa {
2     //declaracao dos atributos
3     //declaracao dos métodos
4 }
```

Exercício I

A partir dos conceitos de classe, objeto, atributos e métodos, crie em Java uma classe Aula.

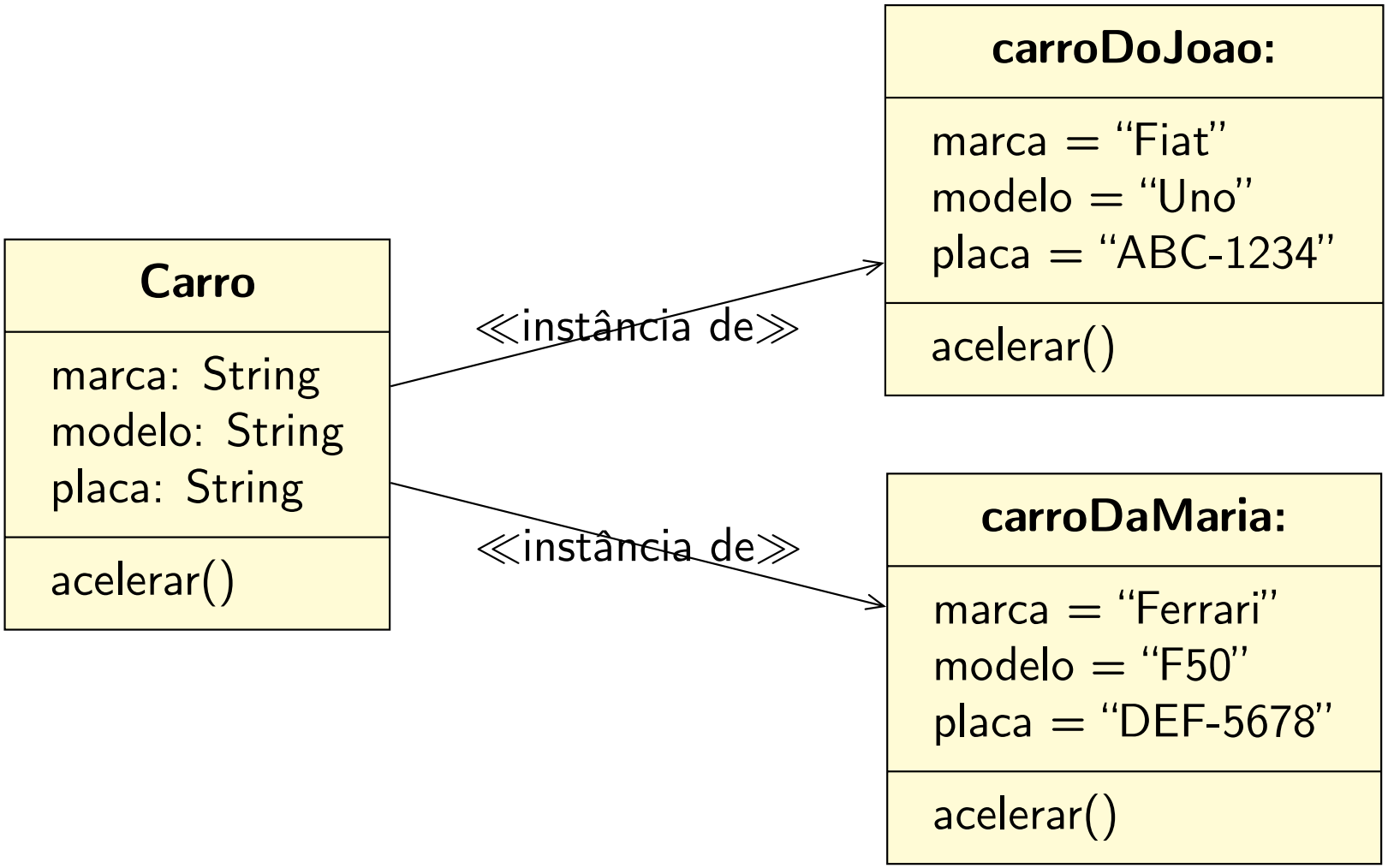
Atributos / variáveis de instância I

- Atributos / variáveis de instância: espaço de memória reservado para armazenar dados, tendo um nome para referenciar seu conteúdo;
- Um atributo ou variável de instância (ou ainda campo) é uma variável cujo valor é específico a cada objeto;
- Ou seja, cada objeto possui uma cópia particular de atributos/variáveis de instância com seus próprios valores.
- Estilo de nome (Java): "Camel Case" com primeira letra da primeira palavra minúscula, primeira das demais maiúscula. Ex.: `nome`, `dataDeNascimento`, `caixaPostal`.

Atributos / variáveis de instância II

Exemplo: classe `Carro`

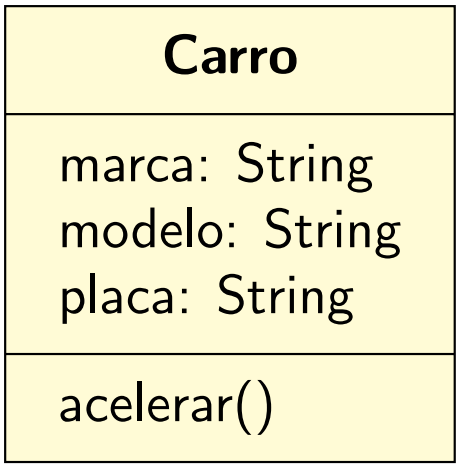
Atributos / variáveis de instância III



Atributos / variáveis de instância IV

```
public class
    NomeDaClasse
{
    //variáveis de instâ
    ncia
    //métodos
}
```

- Variáveis de instância são definidas dentro da classe, fora dos métodos;
- Inicializadas automaticamente.
  - Ou por meio de construtores dos objetos da classe.




```
public class Carro
{
    String marca;
    String modelo, placa;
    // métodos ...
}
```

Exercício I

A partir dos conceitos de classe, objeto, atributos e métodos, crie em Java uma classe Aula e implemente:

- atributos:
  - numAlunos (Inteiro),
  - intervalo (Booleano)

Métodos em Java I

- Os métodos implementam o *comportamento* dos objetos;
- Um método agrupa, em um bloco de execução, uma sequência de comandos que realizam uma determinada função;
- Os métodos em Java são declarados dentro das classes que lhes dizem respeito;
- Eles são capazes de retornar informações quando completam suas tarefas.
- Estilo de nome (Java): "Camel Case" igual à forma usada para atributos. Ex.: `salvar`, `salvarComo`, `abrirArquivo`.

Métodos em Java II

Sintaxe da declaração de um método (semelhante a C):

```
1  public class NomeDaClasse
2  {
3      // declaração dos atributos
4      // ...
5
6      <mod. acesso> <tipo de retorno> nomeDoMétodo (
       argumento[s])
7      {
8          // corpo do método
9      }
10 }
```

Métodos em Java III

Exemplo:

```
1  public class Carro
2  {
3      // declaração dos atributos
4      // ...
5
6      public void acelerar()
7      {
8          // corpo do método
9      }
10 }
```

Métodos em Java IV

- Em geral, um método recebe argumentos / parâmetros, efetua um conjunto de operações e retorna algum resultado.
- A definição de método tem cinco partes básicas:
  - modificador de acesso (`public`, `private`, ...);
  - nome do método;
  - tipo do dado retornado;
  - lista de parâmetros (argumentos);
  - corpo do método.

Métodos em Java V

- Os métodos aceitam a passagem de um número determinado de parâmetros, que são colocados nos parênteses seguindo ao nome;
- Os argumentos são variáveis locais do método – assim como em C.

Exemplo:

```
float hipotenusa (float catetoA, float catetoB)
{
    // corpo do método
    // ...
}
```

Métodos em Java VI

Se o método não utiliza nenhum argumento, parênteses vazios devem ser incluídos na declaração.

Exemplo:

```
public class Carro
{
    String marca;
    String modelo;
    String placa;
    public void acelerar()
    { // corpo do método
    }
}
```

Métodos em Java VII

Um método pode devolver um valor de um determinado tipo de dado.

- Nesse caso, existirá no código do método uma linha com uma instrução `return`, seguida do valor ou variável a devolver.

```
1 class Administrativo {
2     float salario;
3     public float retorneSalario( )
4     {
5         //calcular salario
6         return salario;
7     }
8 }
```

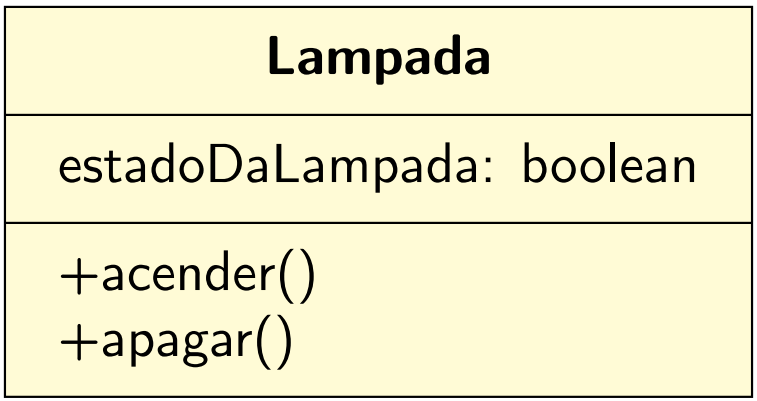
Métodos em Java VIII

Se o método não retorna nenhum valor, isto deve ser declarado usando-se a palavra-chave `void` – como em C.
Exemplo:

```
public class Carro {
    String marca;
    String modelo;
    String placa;
    public void acelerar() // void indica que nao ha
    retorno de valor
    {
    }
}
```

Exemplo – métodos I

Criar um modelo em Java para uma lâmpada. As operações que podemos efetuar nesta lâmpada também são simples: podemos ligá-la ou desligá-la. Quando uma operação for executada, imprima essa ação.

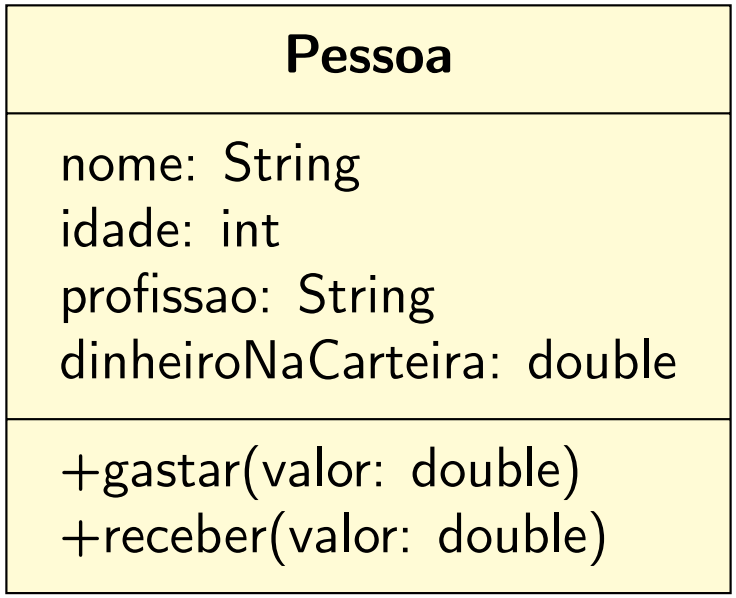


Exemplo – métodos II

```
1  public class Lampada {
2      boolean estadoDaLampada;
3
4      public void acender( )
5      {
6          estadoDaLampada = true;
7          System.out.println("A acao acender foi executada");
8      }
9      public void apagar( )
10     {
11         estadoDaLampada = false;
12         System.out.println("A acao apagar foi executada");
13     }
14 }
```

Exemplo – métodos III

Criar um modelo em Java para uma pessoa. Considere os atributos nome, idade e profissão. A pessoa pode gastar ou receber uma certa quantia de dinheiro (assuma que o dinheiro se encontra na carteira).



Exemplo – métodos IV

```
1  public class Pessoa
2  {
3      String nome, profissao;
4      int idade;
5      double dinheiroNaCarteira;
6      public void gastar( double valor )
7      {
8          dinheiroNaCarteira = dinheiroNaCarteira - valor;
9      }
10     public void receber( double valor )
11     {
12         dinheiroNaCarteira = dinheiroNaCarteira + valor;
13     }
14 }
```

Exercício I

A partir dos conceitos de classe, objeto, atributos e métodos, crie em Java uma classe Aula e implemente:

- atributos:
  - numAlunos (Inteiro), que indica quantos alunos têm na aula.
  - intervalo (Booleano), que indica se a aula está em intervalo ou não.
- métodos:
  - status (Que mostrará a quantidade de alunos e se estão em intervalo ou não, mostrar em mensagem)
  - exercício (Se possuir mais que um aluno e se não estiverem em intervalo, mostrar a mensagem: Começar a aplicar exercícios na sala...)
  - fimIntervalo (Se não possuir nenhum aluno e estiver em intervalo, mostrar a mensagem: Chamar os alunos de volta à sala)

Exercício II

- começo (Se tiver mais de um aluno, mostrar a mensagem: Começo a dar aula para meus alunos; Se tiver um aluno, mostrar a mensagem: Parece que somente um aluno compareceu, começar a aula...; Se não tiver nenhum aluno presente, mostrar a mensagem: É necessário chamar alunos do intervalo...)
- fim (Se tiver pelo menos um aluno, mostrar a mensagem: Fim da aula! Senão mostrar a mensagem: Erro, não há alunos!)

Instanciação de objetos I

- Declaração: a seguinte instrução declara que a variável `nomeDoObjeto` refere-se a um objeto / instância da classe `NomeDaClasse`:

```
NomeDaClasse nomeDoObjeto;
```

- Criação: a seguinte instrução cria (em memória) um novo objeto / instância da classe `NomeDaClasse`, que será referenciado pela variável `nomeDoObjeto` previamente declarada:

```
nomeDoObjeto = new NomeDaClasse();
```

As duas instruções acima podem ser combinadas em uma só:

```
NomeDaClasse nomeDoObjeto = new NomeDaClasse();
```

Instanciação de objetos II

```
public class Carro {
    String marca;
    String modelo;
    String placa;
    public void acelerar()
    {
    }
}

...
Carro carro1 = new Carro();
...
```

Instanciação de objetos III

O comando `new` cria uma instância de uma classe:

- Aloca espaço de memória para armazenar os atributos;
- Chama o construtor da classe;
- O construtor inicia os atributos da classe, criando novos objetos, iniciando variáveis primitivas, etc;
- Retorna uma *referência* (ponteiro) para o objeto criado.

Comando ponto I

Para acessar um atributo de um objeto, usa-se a notação ponto:

<nome do objeto>.<nome da variavel>

```
public class Carro {
    String marca;
    String modelo;
    String placa;
    public void acelerar()
    {
    }
}
```

```
...
Carro car1 = new Carro();
Carro car2 = new Carro();
//inicializando car1
car1.marca = "Fiat";
car1.modelo = "2000";
car1.placa = "FRE-6454";
//inicializando car2
car2.marca = "Ford";
car2.modelo = "1995";
car2.placa = "RTY-5675";
...
```

Comando ponto II

Para acessar um método de um objeto, usa-se a notação ponto:

<nome do objeto>.<nome do metodo>

```
public class Carro {
    String marca;
    String modelo;
    String placa;
    public void acelerar()
    { // corpo do metodo
    }
    public void frear()
    { // corpo do metodo
    }
}
```

```
...
Carro car1 = new Carro();
//inicializando car1
car1.marca="Fiat";
car1.modelo="2000";
car1.placa="FRE-6454";
//usando os métodos
car1.acelerar();
car1.frear();
...
```

Passagem de mensagens I

- Para *mandar mensagens* aos objetos utilizamos o operador ponto, seguido do método que desejamos utilizar;
- Uma mensagem em um objeto é a ação de efetuar uma chamada a um método.

```
Pessoa p1;
p1 = new Pessoa();
p1.nome = "Vitor Josue Pereira";
p1.nascimento = "10/07/1966";
p1.gastar( 3200.00 ); // Mensagem sendo passada ao
    objeto p1
```

Passagem de mensagens II

Um programa orientado a objetos nada mais é do que vários objetos dizendo uns aos outros o que fazer.

- Quando você quer que um objeto faça alguma coisa, você envia a ele uma "mensagem" informando o que quer fazer, e o objeto faz;
- Se o método for público, o objeto terá que executá-lo;
- Se ele precisar de outro objeto para o auxiliar a realizar o "trabalho", ele mesmo vai cuidar de enviar mensagem para esse outro objeto.

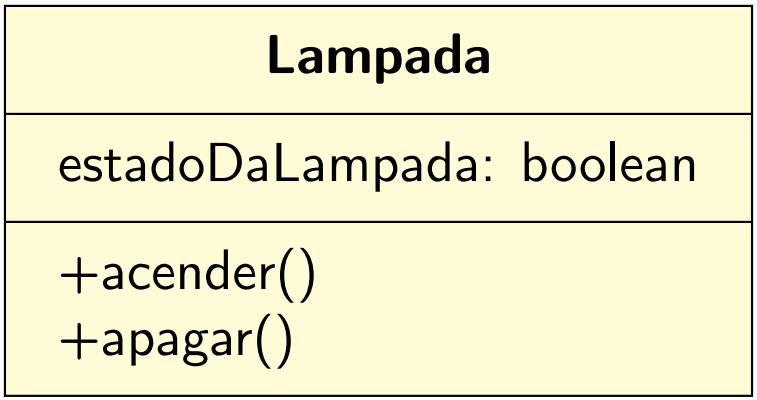
O método `main()`

- O método main() é o ponto de partida para todo aplicativo em Java.
- É nele que são instanciados os primeiros objetos que iniciarão o aplicativo.
- A forma mais usual de se declarar o método `main()` é mostrada abaixo:

```
public class ClassePrincipal
{
    public static void main (String args [])
    {
        //corpo do método
    }
}
```

Exemplo I

Criar um modelo em Java para uma lâmpada. Implemente o modelo, criando dois objetos `Lamp1` e `Lamp2`. Simule a operação `acender` para `Lamp1` e `apagar` para `Lamp2`.



Exemplo II

```
1  public class Lampada {
2      boolean estadoDaLampada;
3
4      public void acender( )
5      {
6          estadoDaLampada = true;
7          System.out.println("A acao acender foi executada");
8      }
9      public void apagar( )
10     {
11         estadoDaLampada = false;
12         System.out.println("A acao apagar foi executada");
13     }
14 }
```

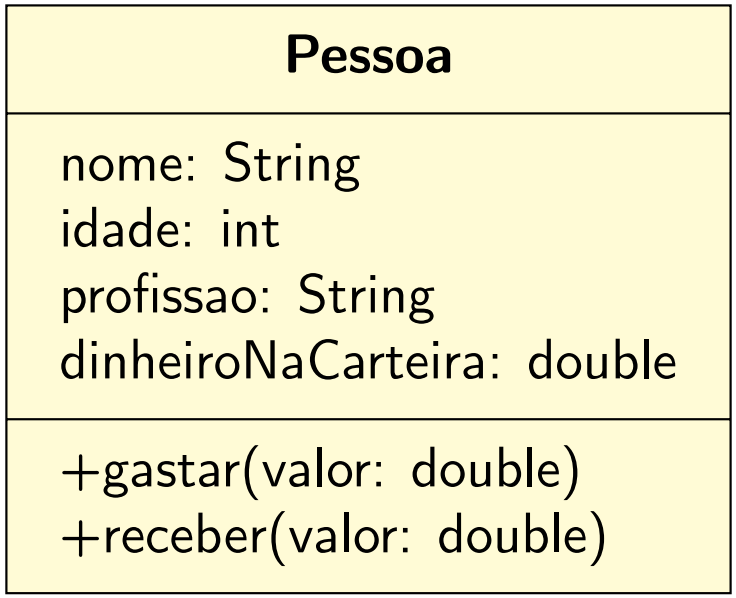
Exemplo III

```
1  public class GerenciadorDeLampadas {
2      public static void main(String args[])  {
3          // Declara um objeto Lamp1 da classe Lampada
4          Lampada Lamp1;
5          // Cria um objeto da classe Lampada
6          Lamp1 = new Lampada();
7          //Simulando operação sobre objeto Lamp1
8          Lamp1.acender();
9
10         // Declara um objeto Lamp2 da classe Lampada
11         Lampada Lamp2;
12         // Cria um objeto da classe Lampada
13         Lamp2 = new Lampada();
14         //Simulando operação sobre objeto Lamp2
15         Lamp2.apagar();
16     }
17 }
```

Exemplo IV

Criar um modelo em Java para uma pessoa. Considere os atributos `nome`, `idade` e `profissão`. A pessoa pode gastar ou receber uma certa quantia de dinheiro (assuma que o dinheiro se encontra na carteira). Implemente o modelo, criando dois objetos `p1` e `p2`. Assuma que o objeto `p1` tem 3.200 na carteira, e o objeto `p2` tem 1.200. Simule operações de gasto e recebimento.

Exemplo V


Pessoa
nome: String
idade: int
profissao: String
dinheiroNaCarteira: double
+gastar(valor: double)
+receber(valor: double)

Exemplo VI

```
1  public class Pessoa
2  {
3      String nome, profissao;
4      int idade;
5      double dinheiroNaCarteira;
6      public void gastar( double valor )
7      {
8          dinheiroNaCarteira = dinheiroNaCarteira - valor;
9      }
10     public void receber( double valor )
11     {
12         dinheiroNaCarteira = dinheiroNaCarteira + valor;
13     }
14 }
```

Exemplo VII

```
1  public class GerenciadorDePessoas
2  {
3      public static void main(String args[])
4      {
5          // Declara um objeto da classe Pessoa
6          Pessoa p1;
7          // Cria um objeto da classe Pessoa
8          p1 = new Pessoa();
9          //Atribuindo valor aos atributos do objeto p1
10         p1.nome = "Vitor Pereira";
11         p1.idade = 25;
12         p1.profissao = "Professor";
13         p1.dinheiroNaCarteira = 3200.00;
14         System.out.println( "Salário de " + p1.nome + " = " +
15         p1.dinheiroNaCarteira );
16         // Vitor recebeu 1000 reais
```

Exemplo VIII

```
17        p1.receber( 1000.00 );
18        System.out.println( p1.nome + "tem " +
19        p1.dinheiroNaCarteira + " reais");
20        // Vitor gastou 200 reais
21        p1.gastar( 200.00 );
22        System.out.println( p1.nome + "tem agora " +
23        p1.dinheiroNaCarteira + " reais");
24
25        // Declara e cria um outro objeto da classe Pessoa
26        Pessoa p2 = new Pessoa();
27        //Atribuindo valor aos atributos do objeto p2
28        p2.nome = "João Silveira";
29        p2.idade = 30;
30        p2.dinheiroNaCarteira = 1200.00;
31        System.out.println( "Salário de " + p2.nome + " = " +
32        p2.dinheiroNaCarteira );
```

Exemplo IX

```
33          // João recebeu 400 reais
34          p2.receber( 400.00 );
35          System.out.println( p2.nome + "tem " +
36          p1.dinheiroNaCarteira + " reais");
37          // João gastou 100 reais
38          p2.gastar( 100.00 );
39          System.out.println( p2.nome + "tem agora " +
40          p1.dinheiroNaCarteira + " reais");
41      }
42  }
```

Exercício I

A partir dos conceitos de classe, objeto, atributos e métodos, crie em Java uma classe Aula e implemente:

- atributos:
  - numAlunos (Inteiro), que indica quantos alunos têm na aula.
  - intervalo (Booleano), que indica se a aula está em intervalo ou não.
- métodos:
  - status (Que mostrará a quantidade de alunos e se estão em intervalo ou não, mostrar em mensagem)
  - exercício (Se possuir mais que um aluno e se não estiverem em intervalo, mostrar a mensagem: Começar a aplicar exercícios na sala...)
  - fimIntervalo (Se não possuir nenhum aluno e estiver em intervalo, mostrar a mensagem: Chamar os alunos de volta à sala)

Exercício II

- começo (Se tiver mais de um aluno, mostrar a mensagem: Começo a dar aula para meus alunos; Se tiver um aluno, mostrar a mensagem: Parece que somente um aluno compareceu, começar a aula...; Se não tiver nenhum aluno presente, mostrar a mensagem: É necessário chamar alunos do intervalo...)
- fim (Se tiver pelo menos um aluno, mostrar a mensagem: Fim da aula! Senão mostrar a mensagem: Erro, não há alunos!)

- Classe principal Instancie um objeto para manipular os atributos e os métodos. Exemplo: atribua uma quantidade de alunos, use os métodos começo, intervalo, mude os atributos e assim por diante...

Java API I

Java API (*Applications Programming Interface*): conjunto de classes pré-definidas do Java;
API está organizadas em pastas (pacotes)

Exemplos:

Pacote	Descrição
`java.lang`	Classes muito comuns (este pacote é importado automaticamente pelo compilador `javac` em todos os programas Java).
`java.io`	Classes que permitem entrada e saída em arquivos.
`java.math`	Classes para executar aritmética de precisão arbitrária
`javax.swing`	Classes de componentes GUI Swing para criação e manipulação de interfaces gráficas do usuário
`java.util`	Utilitários como: manipulações de data e hora, processamento de números aleatórios, armazenamento e processamento de grandes volumes de dados, quebras de *strings* em unidades léxicas etc.

Java API III

A classe `java.lang.Math`, por exemplo, contém valores de constantes, como `Math.PI` e `Math.E`, e várias funções matemáticas. Alguns métodos:

Método	Descrição
`Math.abs(x)`	Valor absoluto de `x`.
`Math.ceil(x)`	Teto de `x` (menor inteiro maior ou igual a `x`).
`Math.cos(x)`	Cosseno de `x` (`x` dado em radianos).
`Math.exp(x)`	Exponencial de `x` (e^x).
`Math.floor(x)`	Piso de `x` (maior inteiro menor ou igual a `x`).

Métodos estáticos

Chamados a partir da classe (e não de um objeto específico)

Java API IV

A classe `String` opera sobre *cadeias de caracteres*.

Método	Descrição
`ob.concat(s)`	Concatena objeto `ob` ao `String s`.
`ob.replace(x,y)`	Substitui, no objeto `ob`, todas as ocorrências do caractere `x` pelo caractere `y`.
`ob.substring(i,j)`	Constrói novo String para `ob`, com caracteres do índice i ao índice $j-1$.

Métodos não são estáticos

Chamados a partir de objetos da classe `String`.

Referências

- HORSTMANN, Cay S.; CORNELL, Gary. *Core Java 2: Vol.1 – Fundamentos*, Alta Books, SUN Mircosystems Press, 7a. Edição, 2005.
- DEITEL, H. M.; DEITEL, P. J. *JAVA – Como Programar*, Pearson Prentice-Hall, 6a. Edição, 2005.
- https://docs.oracle.com/javase/tutorial/java/javaOO/accesscontrol.html