FACULDADE SENAI PARAÍBA

Programação Orientada a Objetos

Prof. Daniel Brandão

Prof. Daniel Brandão

- Graduado em **Sistemas para Internet**
- Especialista em **Aplicações Web**

- Desenvolvedor Web desde 2006
- Professor desde 2011 (SENAI)
- Professor Universitário desde 2013

- Atualmente faculdade **SENAI**
- Secretaria de Saúde **(SES PB)**

Prof. Daniel Brandão

Linhas de pesquisa:

- Sistemas Web;
- Banco de dados;
- Engenharia de Sistemas;
- Ciência de dados;

Prof. Daniel Brandão

Contatos:

DanielBrandao.com.br

@Daniel85br

professordanielbrandao@gmail.com

@danielbrandao.com.br

Avisos

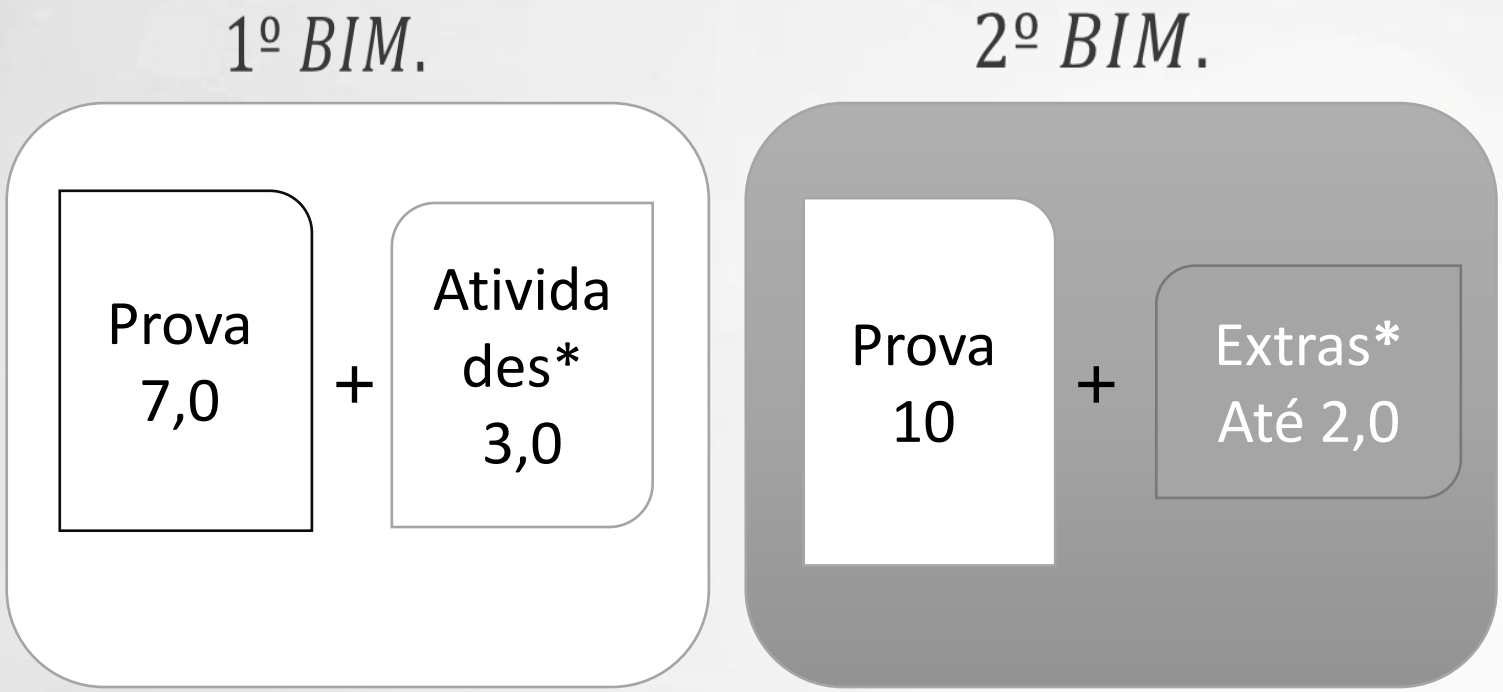
- **Encontros**:
  - Terças (das 18h às 22h).
  - Aulas às terças - intercaladas.
- **Durante a aula:**
  - Coloquem os celulares no silencioso
  - Façam anotações do conteúdo
  - Perguntem e tirem dúvidas
  - Tolerância **MÁXIMA** de 15 minutos

Avisos

- **Trabalhos:**
  - Não é permitido cópia de trabalho;
  - Atenção aos prazos de entrega;
  - Todas as atividades serão pontuadas;

- **Arquivos de aula:**
  - **Github.com/danielbrandao**

Avaliação:



*** Entrega de atividades + assiduidade/pontualidade**

Plano de ensino

OBJETO DE ESTUDO:

- Fundamentos da programação no Paradigma orientado a objetos (POO) para o desenvolvimento de software, com implementação em linguagem de Programação.

Plano de ensino

- **Objetivos** - **Capacitar o aluno a:**

- Utilizar os recursos básicos de um ambiente de programação, com linguagem **orientada a objetos**;
- Implementar **algoritmos** computacionais básicos, em linguagem de programação orientada a objetos;
- Resolver problemas computacionais básicos, aplicados à área de automação industrial, com a utilização de linguagem orientada a objetos.

Plano de ensino

Ementa da disciplina:

- Introdução e conceitos da programação orientada a objetos (POO) Conversão de tipos.
- Semelhanças e diferenças entre programação estruturada e POO.
- Classe. Objetos. Instanciação de objetos. Construtores, atributos e métodos de classe e instância. Arrays.
- Encapsulamento: modificadores de acesso.

Plano de ensino

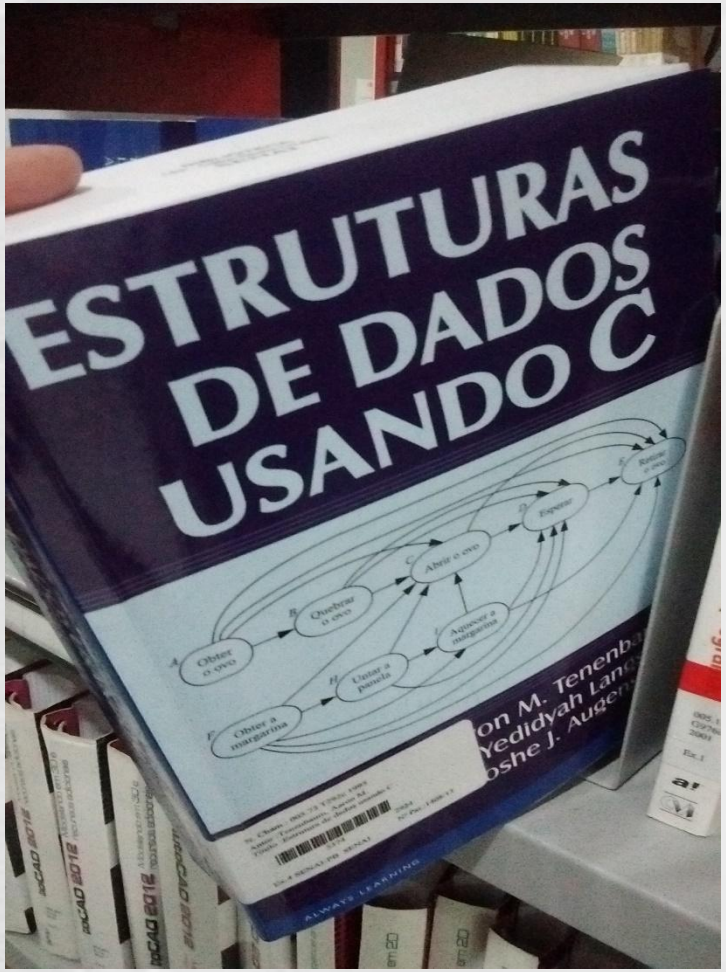
Ementa da disciplina:

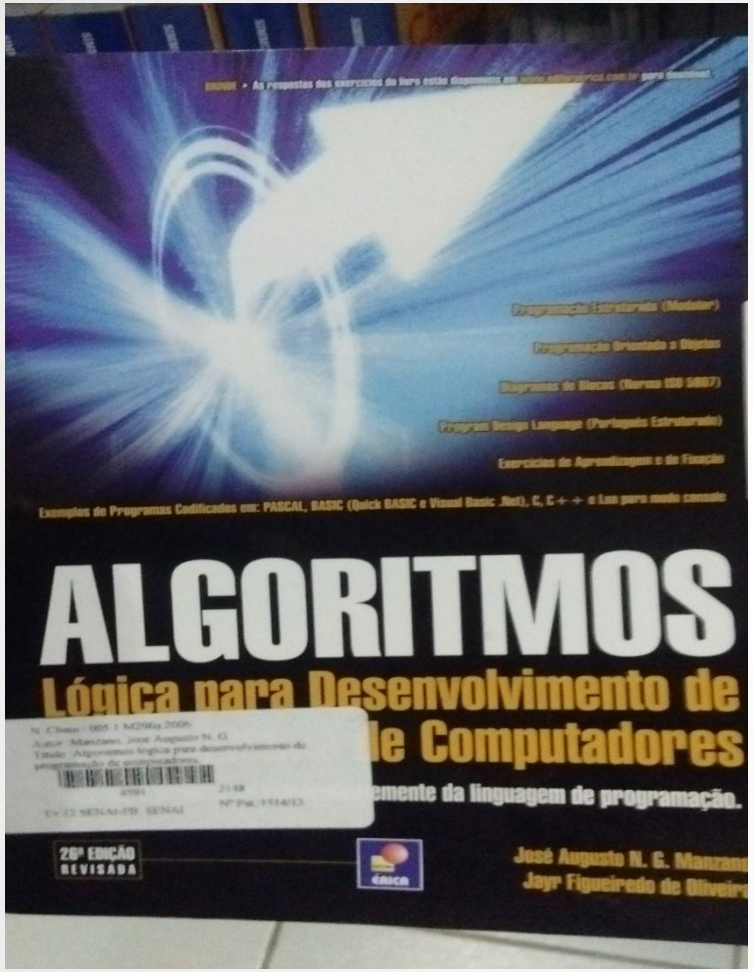
- Herança;
- Polimorfismo;
- Classes abstratas. Interfaces;
- Exceções. Linguagens de POO. Ambientes de POO. Impactos ambientais advindos da utilização de computadores

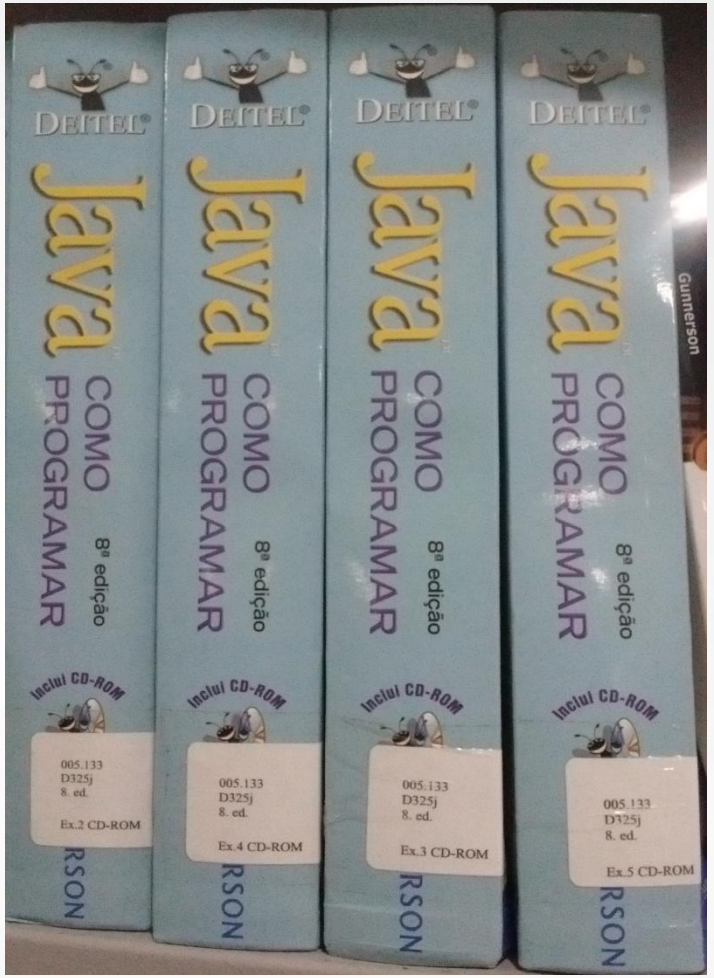
Referências

- DEITEL, H.; DEITEL, P. **Java: como programar. 8ª** ed. São Paulo: Pearson Prentice Hall, 2005.
- SANTOS, Rafael. **Introdução à Programação Orientada a Objetos Usando JAVA.** 1ª Edição. São Paulo: Elsevier, 2003.
- MANZANO, J.A.; OLIVEIRA, J.F. **Algoritmos – Lógica para Desenvolvimento de Programação de Computadores.** 26ª. ed. Revisada. São Paulo: Érica, 2015.
- BORATTI, Isaias Camilo. **Programação Orientada a Objetos em JAVA.** 1ª Edição. São Paulo: Visual Books, 2007
- **CAELUM**. **Java e Orientação a Objetos.** Disponível em https://www.caelum.com.br/apostila-java-orientacao-objetos/

Referências







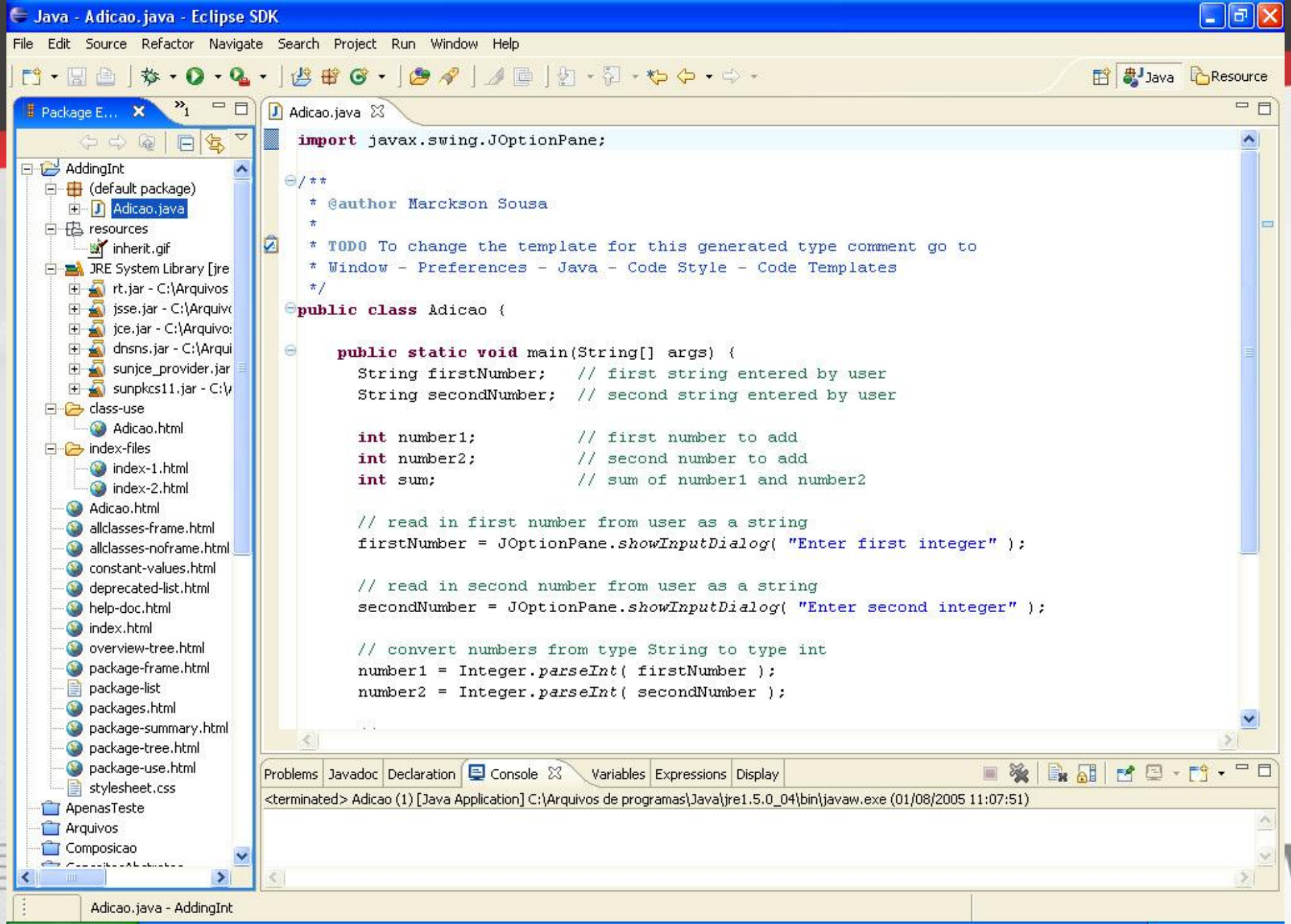
Ambiente de Desenvolvimento

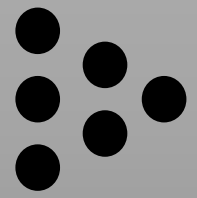
- Apoio de gigantes como IBM, HP, Oracle, Borland, Rational, etc.
  http://www.eclipse.org
- Facilidades de utilização do ambiente

OpenJDK

- Plug-in para acréscimo de funcionalidades (Java JDK)

https://www.oracle.com/


Java - Adicao.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Java
Resource
Package E...
AddingInt
(default package)
Adicao.java
resources
inherit.gif
JRE System Library [jre
rt.jar - C:\Arquivos
jsse.jar - C:\Arquiv
jce.jar - C:\Arquivo
dnsns.jar - C:\Arqui
sunjce_provider.jar
sunpkcs11.jar - C:\
class-use
Adicao.html
index-files
index-1.html
index-2.html
Adicao.html
allclasses-frame.html
allclasses-noframe.html
constant-values.html
deprecated-list.html
help-doc.html
index.html
overview-tree.html
package-frame.html
package-list
packages.html
package-summary.html
package-tree.html
package-use.html
stylesheet.css
ApenasTeste
Arquivos
Composicao
Adicao.java
import javax.swing.JOptionPane;
/**
* @author Marckson Sousa
*
* TODO To change the template for this generated type comment go to
* Window - Preferences - Java - Code Style - Code Templates
*/
public class Adicao {
public static void main(String[] args) {
String firstNumber; // first string entered by user
String secondNumber; // second string entered by user
int number1; // first number to add
int number2; // second number to add
int sum; // sum of number1 and number2
// read in first number from user as a string
firstNumber = JOptionPane.showInputDialog("Enter first integer");
// read in second number from user as a string
secondNumber = JOptionPane.showInputDialog("Enter second integer");
// convert numbers from type String to type int
number1 = Integer.parseInt(firstNumber);
number2 = Integer.parseInt(secondNumber);
Problems Javadoc Declaration Console Variables Expressions Display
<terminated> Adicao (1) [Java Application] C:\Arquivos de programas\Java\jre1.5.0_04\bin\javaw.exe (01/08/2005 11:07:51)
Adicao.java - AddingInt



Introdução a Orientação a Objetos

Introdução POO

- **Orientação a Objetos:**
  - POO é um Paradigma que procura compor modelos de forma mais próxima às interações existentes **no mundo real**, cujas primeiras propostas datam da **década de 60**;





- POO é uma forma de **PROGRAMAÇÃO** de computadores onde se usam **objetos** e **classes,** criados a partir de modelos para representar e processar **dados** usando programas de computadores.

Introdução POO

- Analogia:
  - Problema: Usar um carro.

  - Mas antes disso, o que precisa?
    - Alguém tem que projetar...
    - Desenhos de engenharia elaboram o projeto
    - É decidido sobre as características do veículo como: quantidade de portas, potência do motor, cores, modelo (sedan, hatch, etc)...

Introdução POO

- Analogia:
  - Problema: Usar um carro.

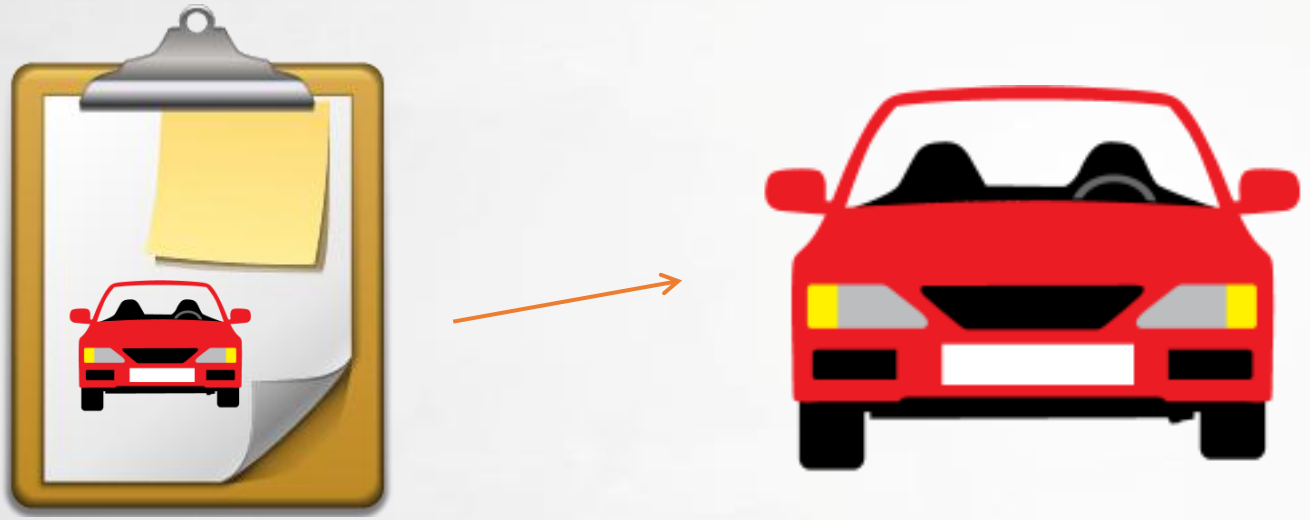
  - No Projeto temos:
    - Pedais de acelerador, de freio...
      - ‘Ocultam’ os complexos mecanismos que realmente fazem o veículo funcionar;

Introdução POO

- **Problema: Usar um carro.**
  - Podemos dirigir um carro que está ainda no projeto?
  - Não... Infelizmente não podemos guiar os desenhos do projeto de um carro! É necessário um produto finalizado.

Introdução POO

- **Problema: Usar um carro.**
  - Antes de guiar, ele deve ser construído a partir dos desenhos de engenharia que o descrevem:



Introdução POO

- **Problema: Dirigir um carro.**
  - Ok, o carro está pronto... Mas ele consegue acelerar sozinho?
  - Não... O motorista tem que pressionar o pedal do acelerador!

Introdução POO

- **Analogia:**
  - Para executar tarefa (rotinas) em um programa é necessário um método;
  - O método descreve os mecanismos que realmente realizam suas tarefas (ROTINAS);
    - Ocultando de seu usuário as tarefas complexas que este realiza, Assim como Os pedais do nosso carro!

Introdução POO

- **Em Java:**
  - Primeiro criamos uma unidade de programa chamada **classe** para abrigar esses tais **métodos**;
  - Você pode fornecer um ou mais métodos que são projetados para realizar as tarefas da classe;
  - Por Exemplo....
    - Uma classe Conta bancária pode abrigar os métodos (tarefas) depositar, debitar, perguntar o saldo atual...

Introdução POO

- **Em Java:**
  - Assim como não podemos dirigir um **projeto** de um carro, não podemos executar **métodos** de uma **classe**;
  - Assim como alguém tem que construir esse carro a partir do seu projeto, você deve construir um **objeto** de uma **classe** antes de fazer um programa realizar as tarefas que a classe descreve como fazer;

Introdução POO

- **Analogia:**
  - Além das capacidades do carro, ele também possui características: Cor, modelo, quantidade de portas...
  - Essas capacidades também são descritas no projeto de engenharia do carro...
  - Cada carro mantém seus próprios:
    - **atributos.**

Introdução POO

- **Em Java:**
  - Um **objeto** tem **atributos** que são portados consigo quando este é utilizado em um programa;
  - Por Exemplo:
    - **Uma classe conta possui:**
      - Número, Saldo, Tipo...

  - Esses atributos são especificados pelas chamadas variáveis de instância.

Conceito de Objetos

- Um exemplo, um celular:
  - **Identificação**
    - Número: (83)9999-9999
  - **Outras propriedades**
    - Largura: 58,6 mm
    - Altura: 115,2 mm
    - Cor: Branco
  - **Comportamento**
    - Ligar
    - Desligar
    - Efetuar chamadas
    - Abrir aplicativos

Fonte: http://store.apple.com/

Conceito de Objetos

Neste caso:

- O CELULAR precisa de uma CLASSE que defina suas características.
- Cada celular diferente será declarado como um OBJETO.
- Por ex: objeto IPHONE X;
  objeto Galaxy J5;
  Etc...

Conceito de Objetos

- O que esses dois smartphones tem em comum?

X

Conceito de Objetos

- O que esses dois smartphones tem em comum?

X

- Ambos são objetos;
- Possuem tela, Câmera, Wifi, 4G;
- Fazem e recebem ligações, mensagens;
- Etc..

Conceito de Objetos

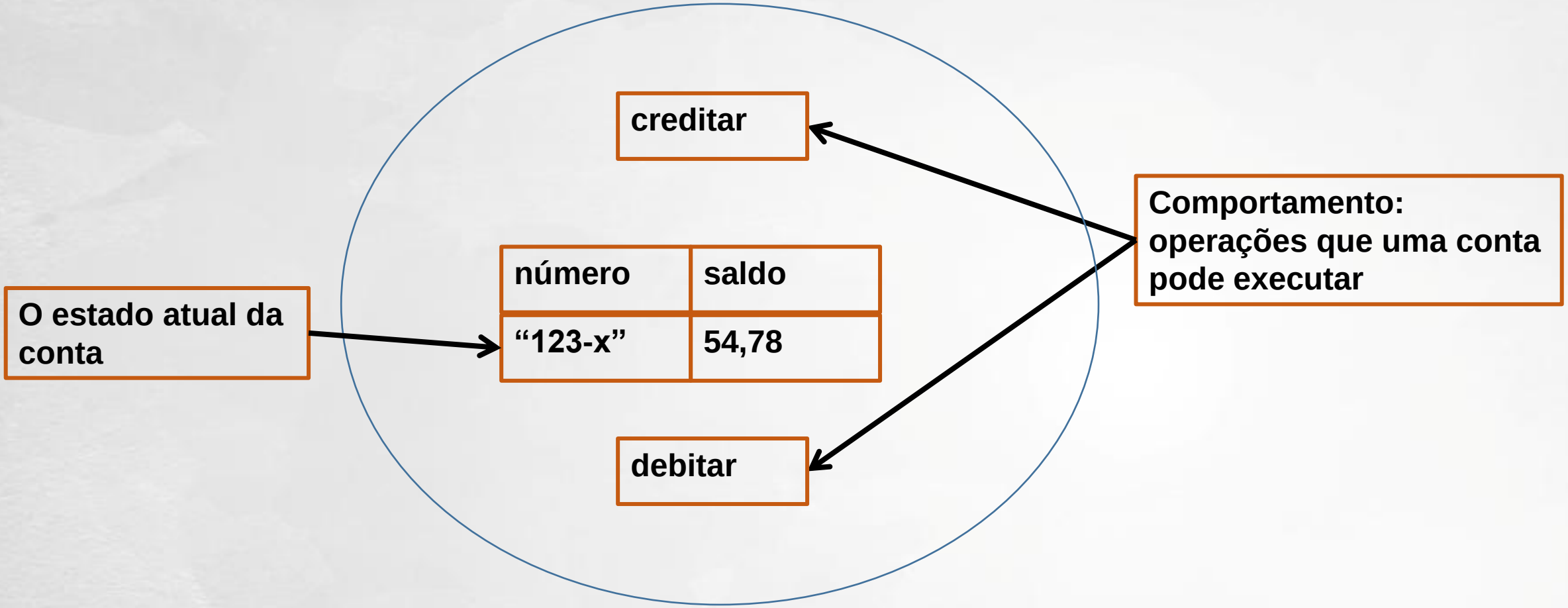
- Um objeto **sozinho** não representa um sistema;
- Um sistema é representado por **vários objetos** distintos ativos, que podem ser chamados de **instância**;
- E a comunicação entre esses objetos ocorre através de **mensagens**, e o envio de cada mensagem significa executar um **método**;

Conceito de Objetos

- Um **MÉTODO** é a função exercida por um **OBJETO**;

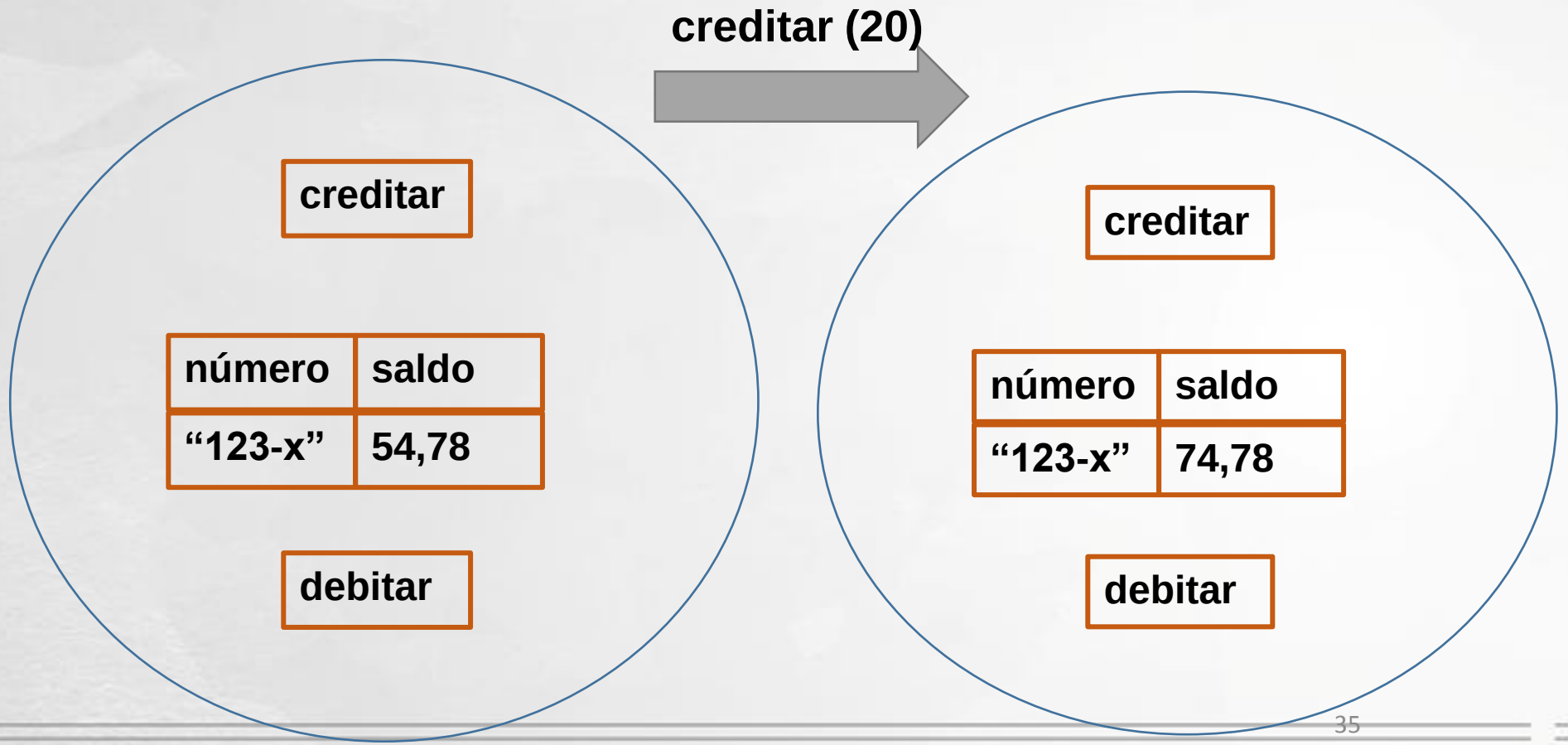
- As funções são variadas, podendo ser elas comuns a vários objetos ou específicas de alguns.

Objeto Conta Bancária



Estado do Objeto Conta

Comportamento mudou o estado do objeto conta bancária



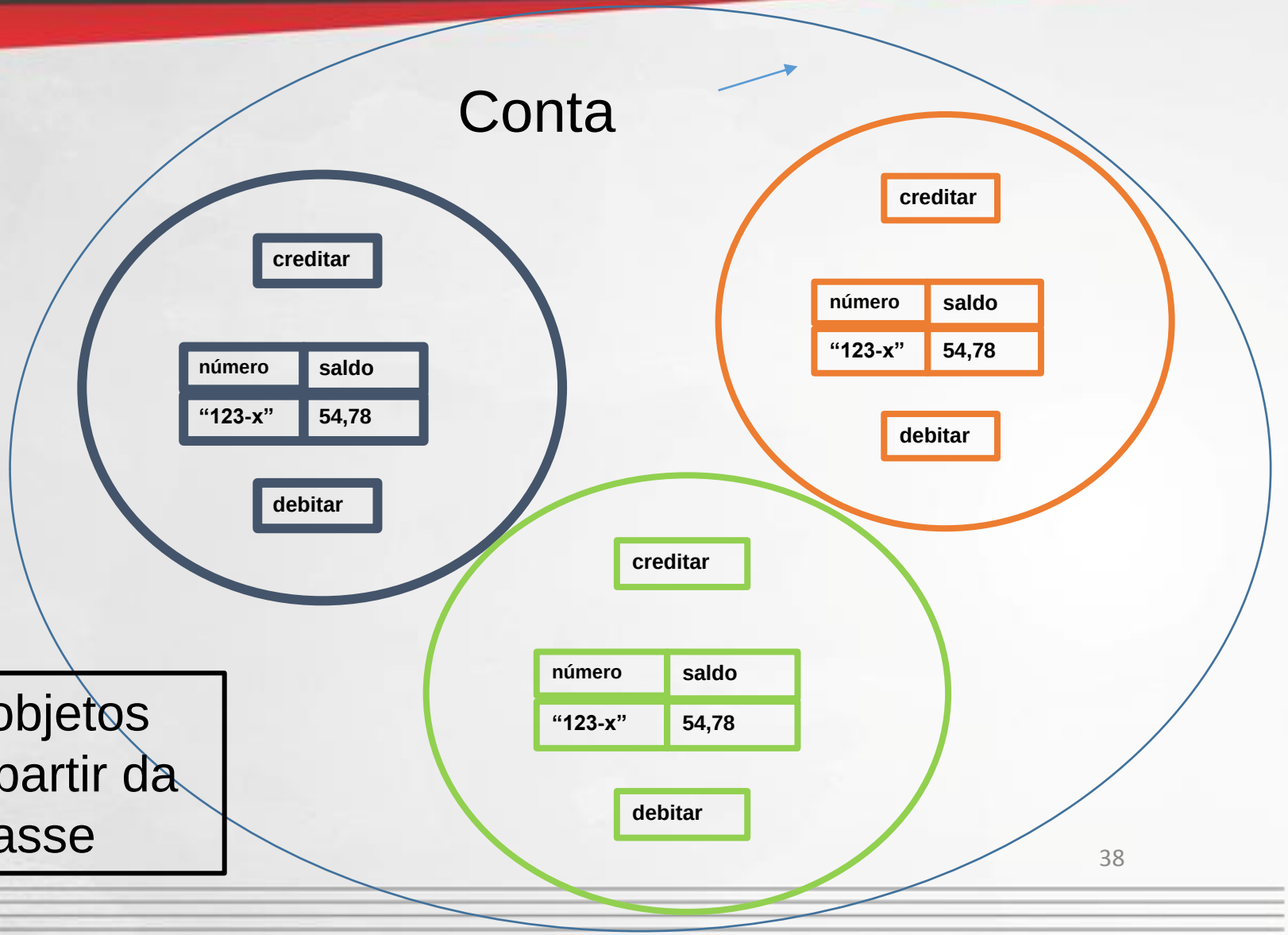
Conceito de Classes

- Voltando para o exemplo do carro, existem vários carros de vários modelos. Um objeto **Pajero** é um instância da classe **Carro**;
- Carros de modelos diferentes possuem características e comportamentos diferentes;
- Logo, pode-se concluir que uma classe é uma espécie de **modelo** (protótipo) que agrupa as **características** e os **comportamentos** em **comum** aos objetos do mesmo tipo;

Conceito de Classes

- É na **classe** que definimos as **variáveis (atributos)** e os **métodos (funções)** que serão utilizados e os objetos são criados (instanciados) a partir das classes;
- Cada **objeto** possui seu espaço de memória individual, assim como suas **variáveis**;
- Tais variáveis são denominadas de variáveis de instâncias (atributos);

Classe x Objeto



Classe

Objetos

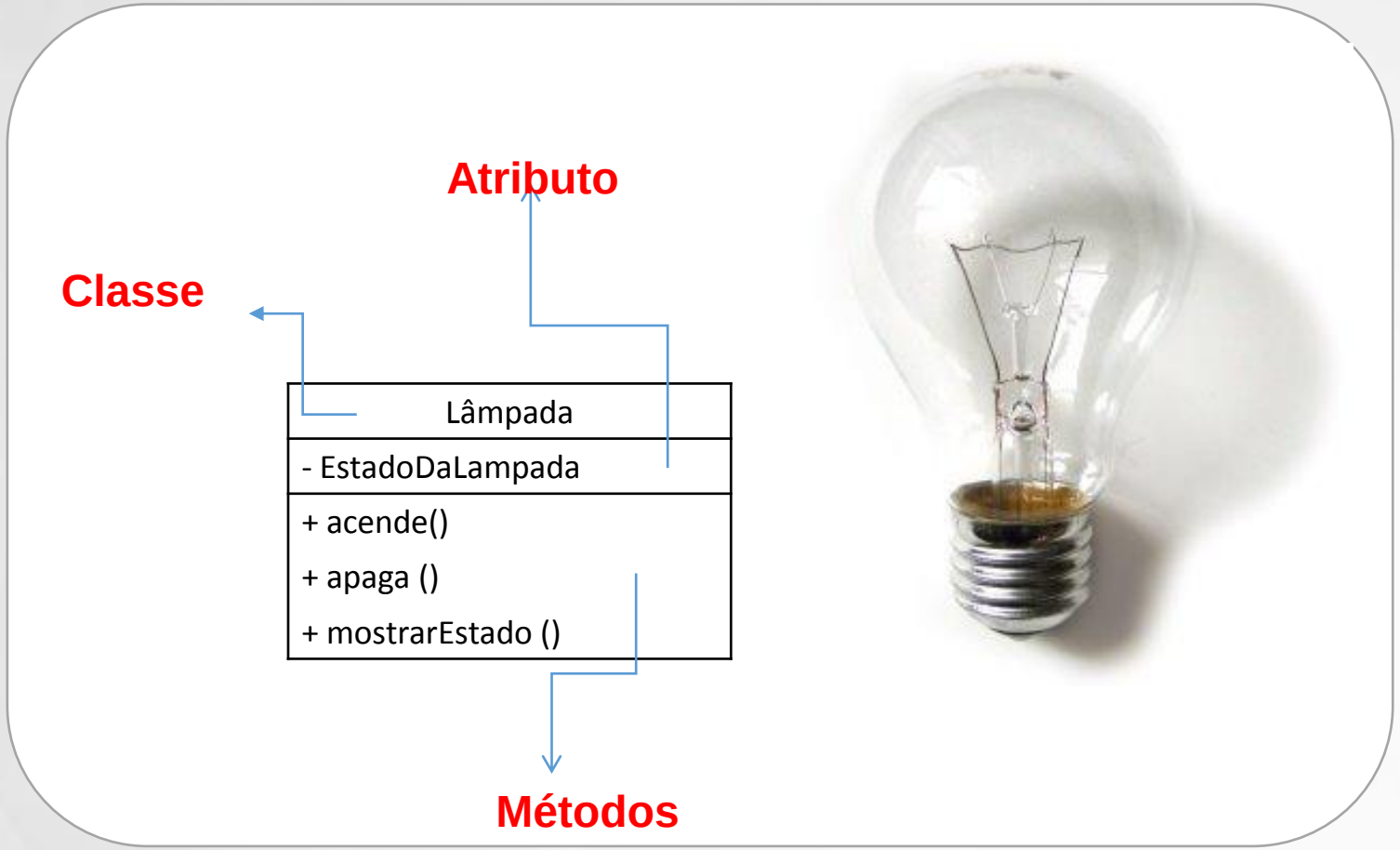
Múltiplos objetos criados a partir da mesma classe

Benefícios da O.O.

- Acelerar o tempo de desenvolvimento;
- Reduzir o tempo de manutenção;
- Mais fácil de entender e adaptar;
- Código de melhor qualidade;

Lembre-se!

Classe ≠ Objeto


Atributo
Classe
Lâmpada
- EstadoDaLampada
+ acende()
+ apaga ()
+ mostrarEstado ()
Métodos

E em java?

- Classe Lâmpada:

Palavra reservada class seguida do nome da classe

Parâmetro, conjunto de caracter

```
1
2  public class Lampada {
3      String estadoDaLampada;
4
5      public void acende(){
6          estadoDaLampada = "aceso";
7      }
8      public void apaga(){
9          estadoDaLampada = "apagado";
10     }
11     public void mostrarEstado(){
12         System.out.print(estadoDaLampada);
13     }
14 }
```

A declaração de método possui respectivamente:
O modificador de acesso
- O tipo de retorno
- Nome do método
- Lista de parâmetros para recebimento (tipo, nome)

E em java?

- Instanciando Objetos

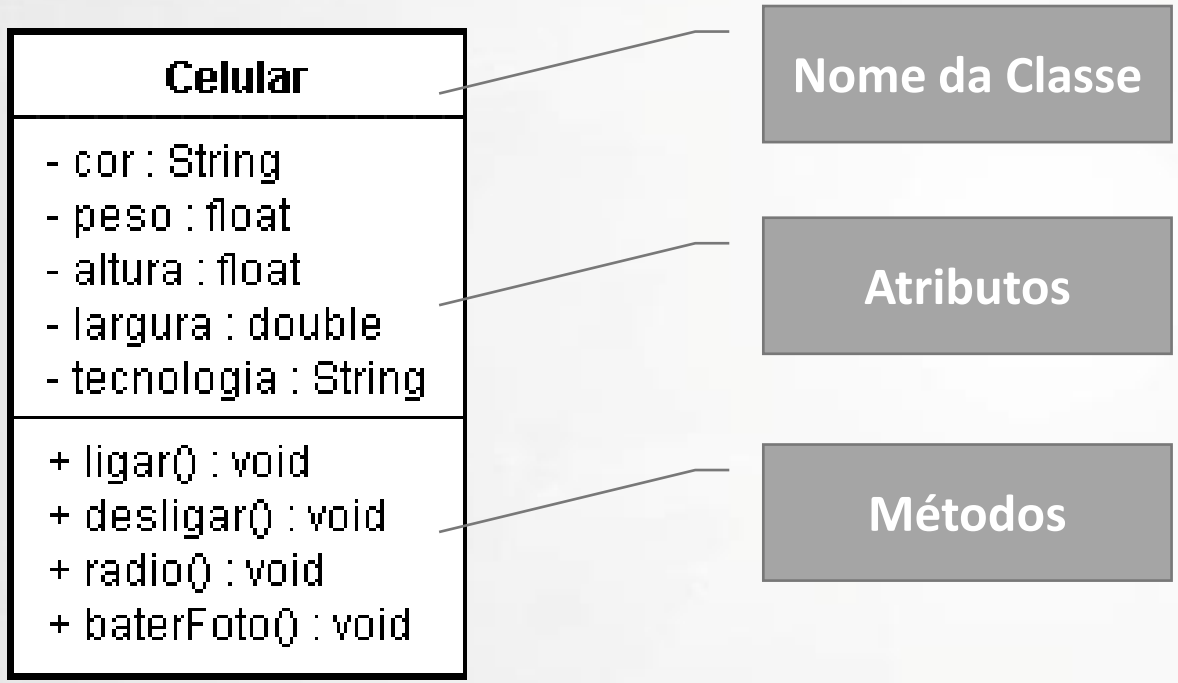
```
1
2  public class usarLampada {
3      public static void main(String[] args) {
4
5          Lampada encandecente = new Lampada();
6          encandecente.acende();
7          encandecente.mostrarEstado();
8
9          Lampada fluorescente = new Lampada();
10         fluorescente.apaga();
11         fluorescente.mostrarEstado();
12
13         encandecente.apaga();
14         encandecente.mostrarEstado();
15     }
16 }
17
```

Para instanciar um objeto, é necessário informar a Classe a que este pertence, designar um nome para ele, e utilizar a palavra reservada new antes da chamada do método construtor – responsável por iniciar os parâmetros (características) do novo objeto.

Para acessar, ou realizar uma chamada, de um método ou parâmetro é utilizado o ponto (.) seguido do nome do método

Introdução POO

- **RESUMINDO:** A **classe** é a entidade responsável por definir os **atributos** (características) e os **métodos** (serviços) que serão oferecidos.



Como executar aplicações?

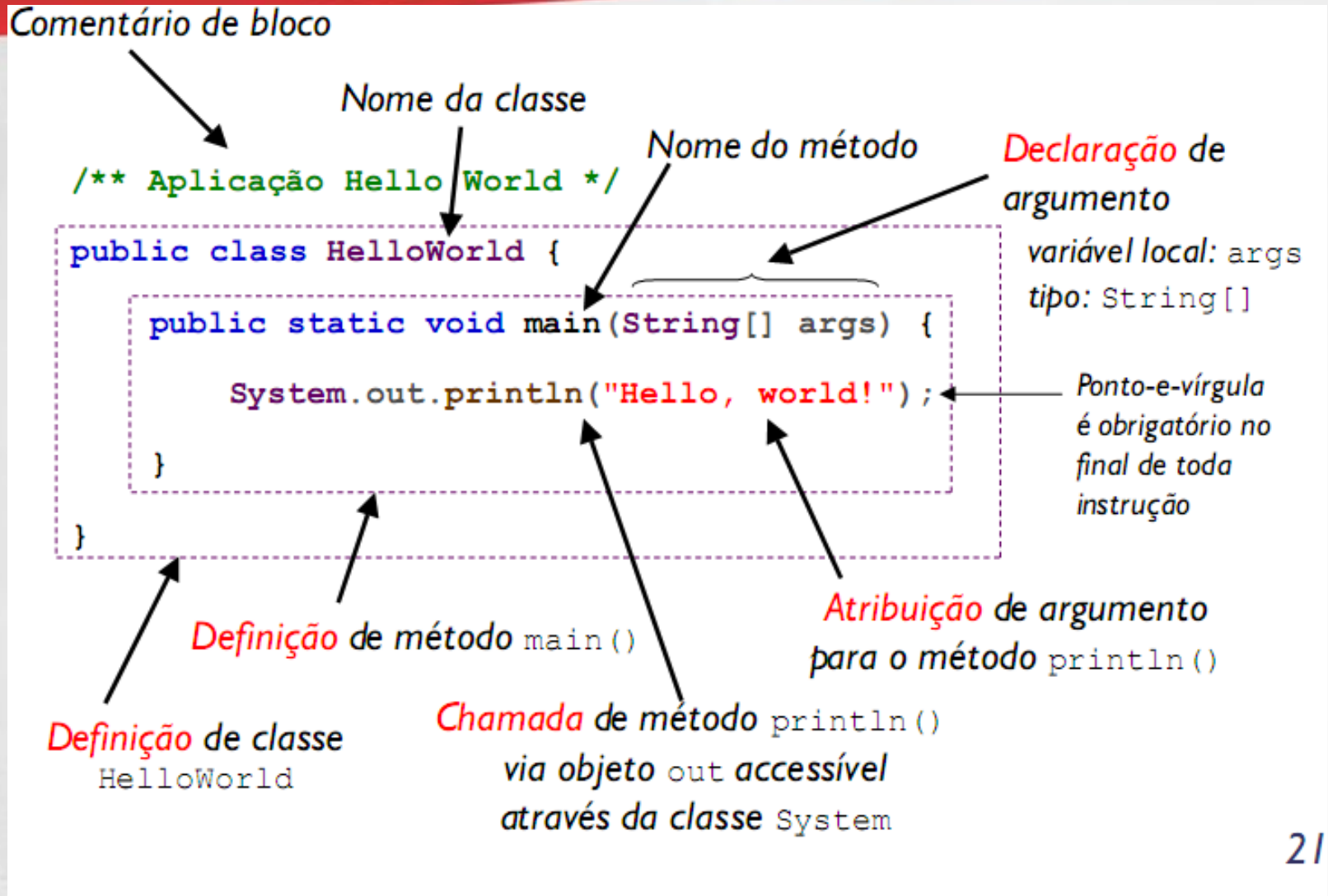
- Uma classe pode definir um método “**main**”;
- É o método responsável pela execução da aplicação;
- Uma aplicação pode conter vários métodos “**main**” (um em cada classe), mas apenas um desses será definido como o método da aplicação;

Assinatura de main

- Ponto de início de toda aplicação Java. Ex:

```
public static void main(String [ ] args){
// Bloco de comando
}
```

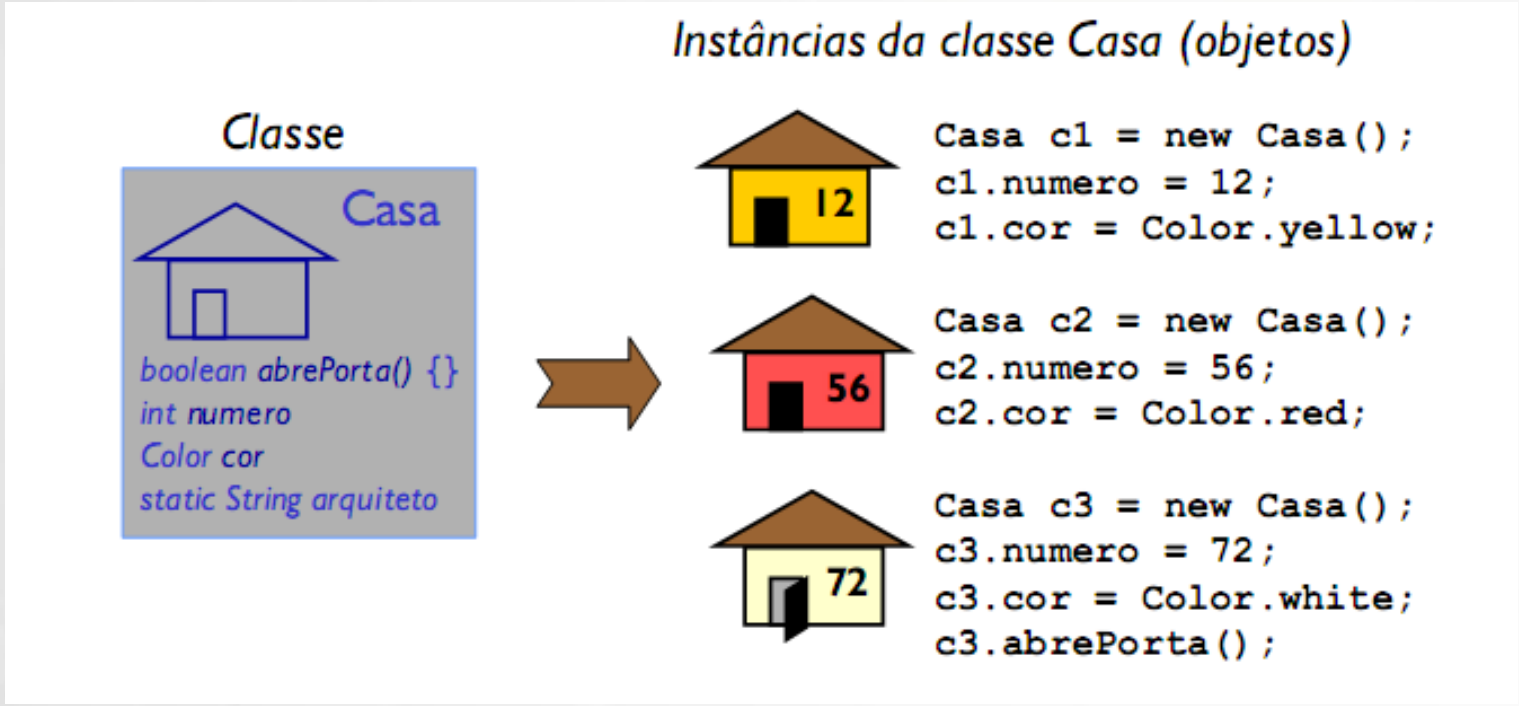
Exemplo do HelloWorld



Como criar e acessar objetos?

- Para criar, use o operador **new**
- Para acessar atributos e métodos, utilize o “.” (ponto)

Exemplo:



Dúvidas?

DICAS JAVA ONLINE:

- **CURSOS ONLINE**: Udemy, Codeacademy, School of Net, Canal prof. Gustavo Guanaraba;
- **Site Devmedia**: http://www.devmedia.com.br/
- Fóruns online;

Referências desta aula

- **Caelum**. **Apostila do curso Java e Orientação a Objetos**. Disponível em: http://www.caelum.com.br/apostilas/
- **SANTOS**, Rafael. **Introdução à Programação Orientada a Objetos Usando Java**. 2º Edição. Elsevier – Rio de Janeiro, 2013. [na biblioteca]
- **SOARES**, Sérgio. **Orientação a Objetos e Java**. Disponível em: <http://www.cin.ufpe.br/~if101/especializacao/Java2.html>
- **SAUVÉ**, Jacques. **Orientação a Objetos**. Disponível em: <http://www.dsc.ufcg.edu.br/~jacques/cursos/p2/html/p2-2.htm>