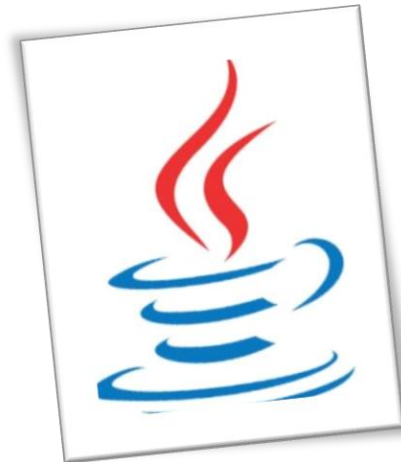


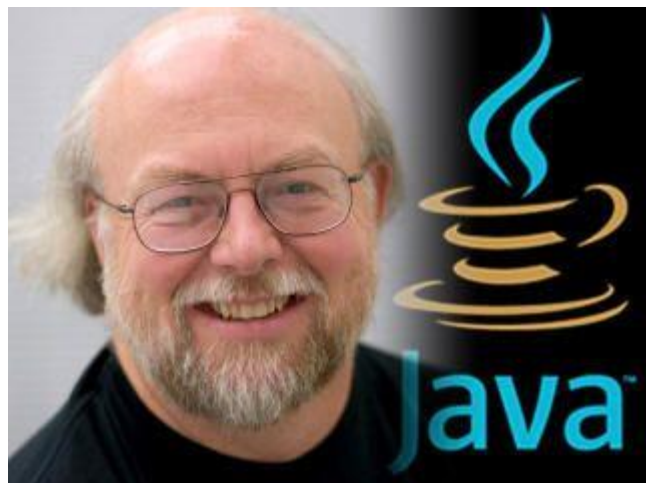
Criando classes em Java

Prof. Dr. Fernando Almeida

Prof. Leonildo Carnevalli Junior



História do Java



Criada por James Gosling da Sun Microsystems e lançada em 1996

Nome inicial era OAK (Carvalho) e depois para Java

Criada inicialmente para execução em televisões, eletro-domésticos, telefones, etc

Utilizou como ponto de partida o mesmo estilo de sintaxe do C/C++

Utiliza o conceito de WORA (Write Once, Run Anywhere)

Mercado

Mercado de trabalho em crescimento

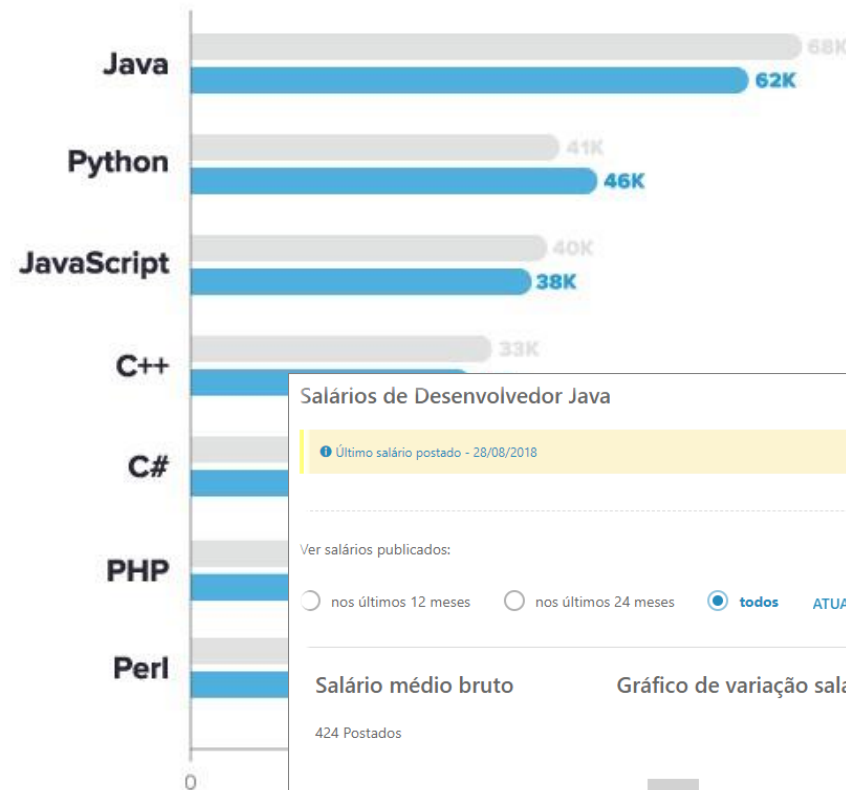
Aug 2018	Aug 2017	Change	Programming Language	Ratings	Change
1	1		Java	16.881%	+3.92%
2	2		C	14.966%	+8.49%
3	3		C++	7.471%	+1.92%
4	5	⬆	Python	6.992%	+3.30%
5	6	⬆	Visual Basic .NET	4.762%	+2.19%
6	4	⬇	C#	3.541%	-0.65%
7	7		PHP	2.925%	+0.63%
8	8		JavaScript	2.411%	+0.31%
9	-	⬆	SQL	2.316%	+2.32%
10	14	⬆	Assembly language	1.409%	-0.40%
11	11		Swift	1.384%	-0.44%
12	12		Delphi/Object Pascal	1.372%	-0.45%
13	17	⬆	MATLAB	1.366%	-0.25%
14	18	⬆	Objective-C	1.358%	-0.15%

Mercado

Número de empregos e média salarial

Job postings containing top languages

Indeed.com - November, 17th 2017



Salários de Desenvolvedor Java

Último salário postado - 28/08/2018

Ver salários publicados:

☐ nos últimos 12 meses ☐ nos últimos 24 meses ☒ todos [ATUALIZAR](#)

Salário médio bruto

424 Postados

R\$ 5.340/mensal

min. R\$ 964 máx. R\$ 61.764

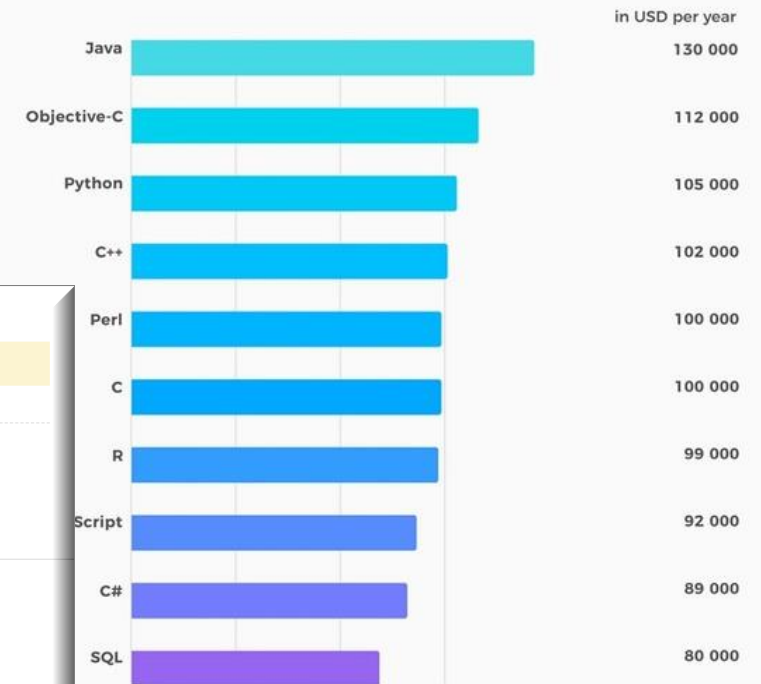
Gráfico de variação salarial



TOP 10

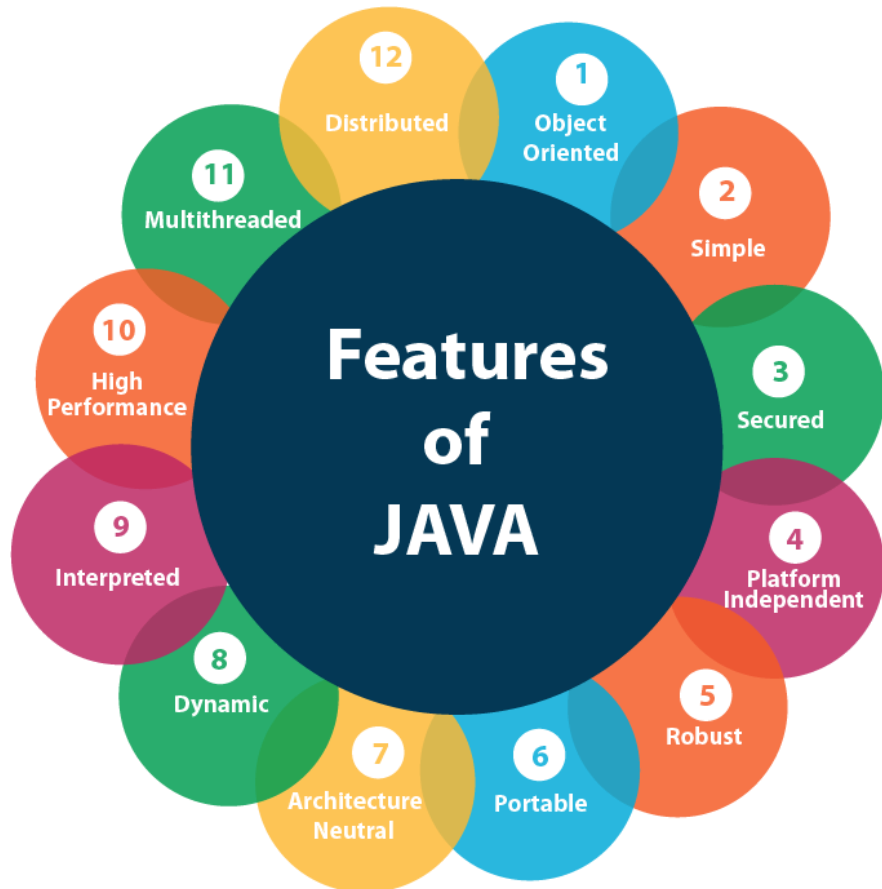
CHALLENGEROCKET.COM RANKING

OF PROJECTED EARNINGS IN 2017 BY A PROGRAMMING LANGUAGE



CHALLENGEROCKET.COM

Características do Java



Maior facilidade na programação

Escreva uma vez, execute em qualquer lugar

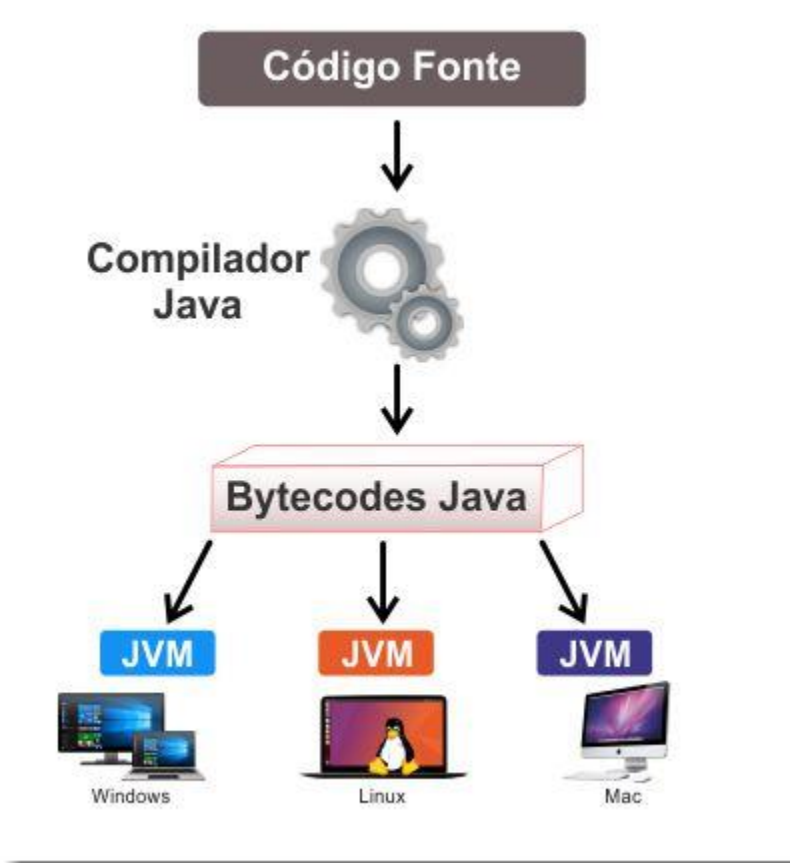
Portabilidade do código

Possibilidade de programas executarem mais de uma tarefa (*multithreading*);

Programação centrada na rede

JVM – Java Virtual Machine

WORA (Write Once, Run Anywhere)



Simula uma máquina real para o bytecode

Interpreta byte codes (que são independentes de plataforma de hardware);

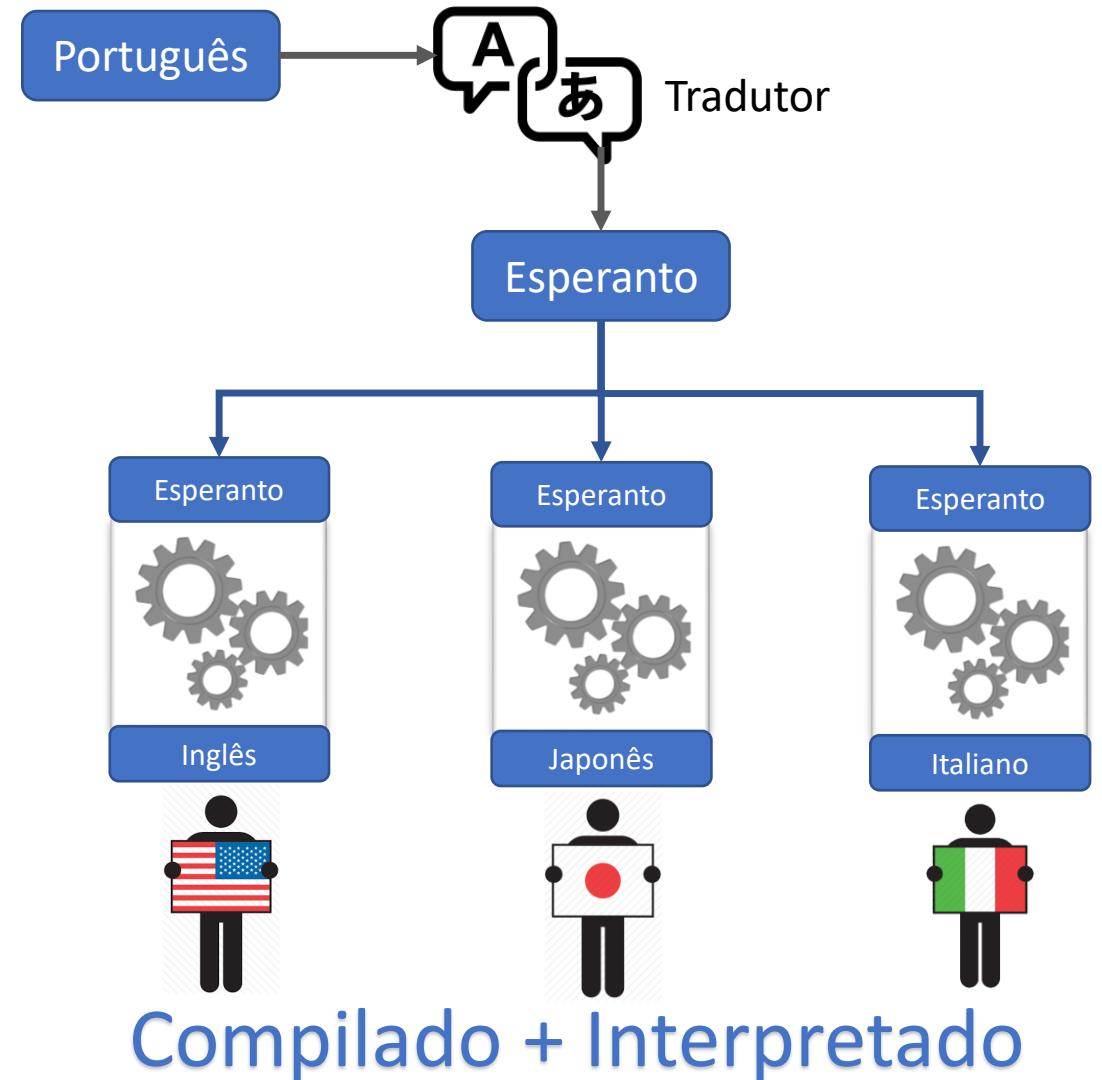
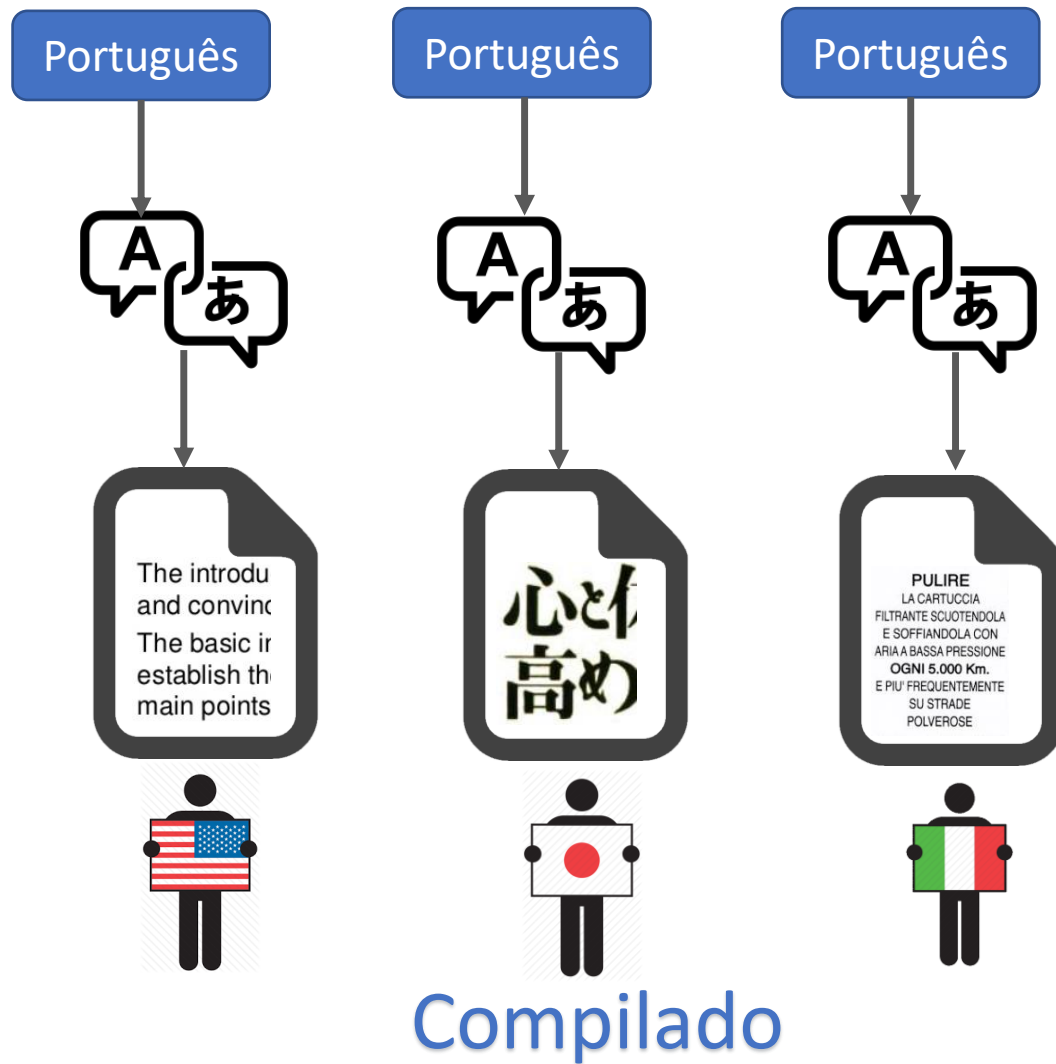
Pode ser implementada tanto na forma de software como de hardware

Possui código compacto

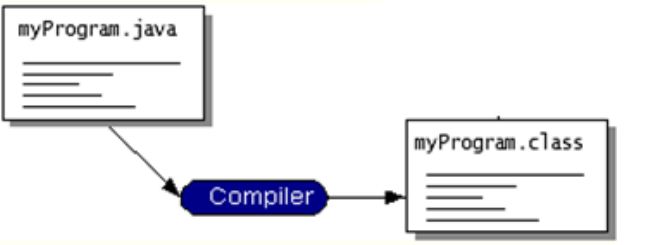
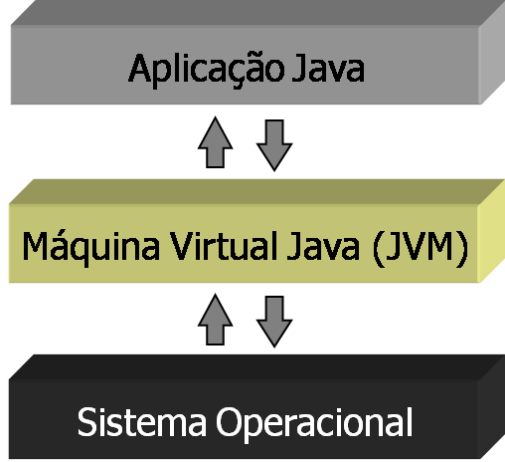
Torna a linguagem JAVA portátil para diversas plataformas.

Qualquer interpretador JAVA (seja para desenvolvimento de tecnologia JAVA ou um browser que rode *applets*) tem sua máquina virtual.

Analogia entre Compilado e Interpretado



Sequencia de execução Java

Criação do programa	<ul style="list-style-type: none">• O programa é criado no editor e armazenado em disco	 <pre>graph LR; A[myProgram.java] --> B[Compiler]; B --> C[myProgram.class];</pre>
	<ul style="list-style-type: none">• O compilador cria bytecodes e os armazena em disco.	
Execução do programa	<ul style="list-style-type: none">• O carregador (loader) de classe coloca bytecodes na memória.	 <pre>graph TD; A[Aplicação Java] <--> B[Máquina Virtual Java (JVM)]; B <--> C[Sistema Operacional];</pre>
	<ul style="list-style-type: none">• O verificador de bytecodes confirma que todos os bytecodes são válidos e não violam restrições de segurança do Java	
	<ul style="list-style-type: none">• O interpretador lê os bytecodes e os traduz para uma linguagem que o computador pode entender, possivelmente armazenando valores dos dados enquanto executa o programa	

Estrutura Programa java

```
package meupacote;
```

Package. Utilizado quando o código do programa deverá fazer parte de um pacote.

```
import java.lang.*;
```

Import. Seção de importação de bibliotecas.

```
/** Primeiro programa Java  
    Conhecendo a estrutura de um  
    programa Java */
```

Comentários. Com sintaxe “// ... para comentários simples ou “/* ... */” e a mais recente “/** .. */” que permite geração de documentação automática (ferramenta javadoc)

```
public class MinhaClassePublica {  
    .....  
    /** Comentário sobre o método */  
    public (private/protected) tipoRet  
    nomeMetodo(<parametros>) {  
        // código do método  
    } // fim da definição do método  
} // fim da classe
```

Classes. Declaração de classes, atributos e métodos do programa Java. A declaração e a definição dos métodos ocorre obrigatoriamente dentro do limite de declaração da classe.

Método main(). Indica que a classe Java é um aplicativo que será interpretado pela máquina virtual.

Programando

Elementos de programação estruturada

```
// Nosso primeiro programa Java
// Conhecendo a estrutura de um programa Java
public class HelloWorld {
    public static void main (String arg[]) {
        System.out.println("Hello, World!");
    } // fim do método main
} // fim da classe MeuPrimeiroPrograma
```

Função Principal. Programas em Linguagem C e C++ e Java buscam seu início pela função principal (main()).

Parâmetros. Parâmetros em funções permitem que essas iniciem com valores recebidos externamente, para variáveis que utilizarão internamente.

Programando

Elementos de programação Orientada a Objeto

```
// Nosso primeiro programa Java
// Conhecendo a estrutura de um programa Java
public class HelloWorld {
    public static void main (String arg[]) {
        System.out.println("Hello, World!");
    } // fim do método main
} // fim da classe MeuPrimeiroPrograma
```

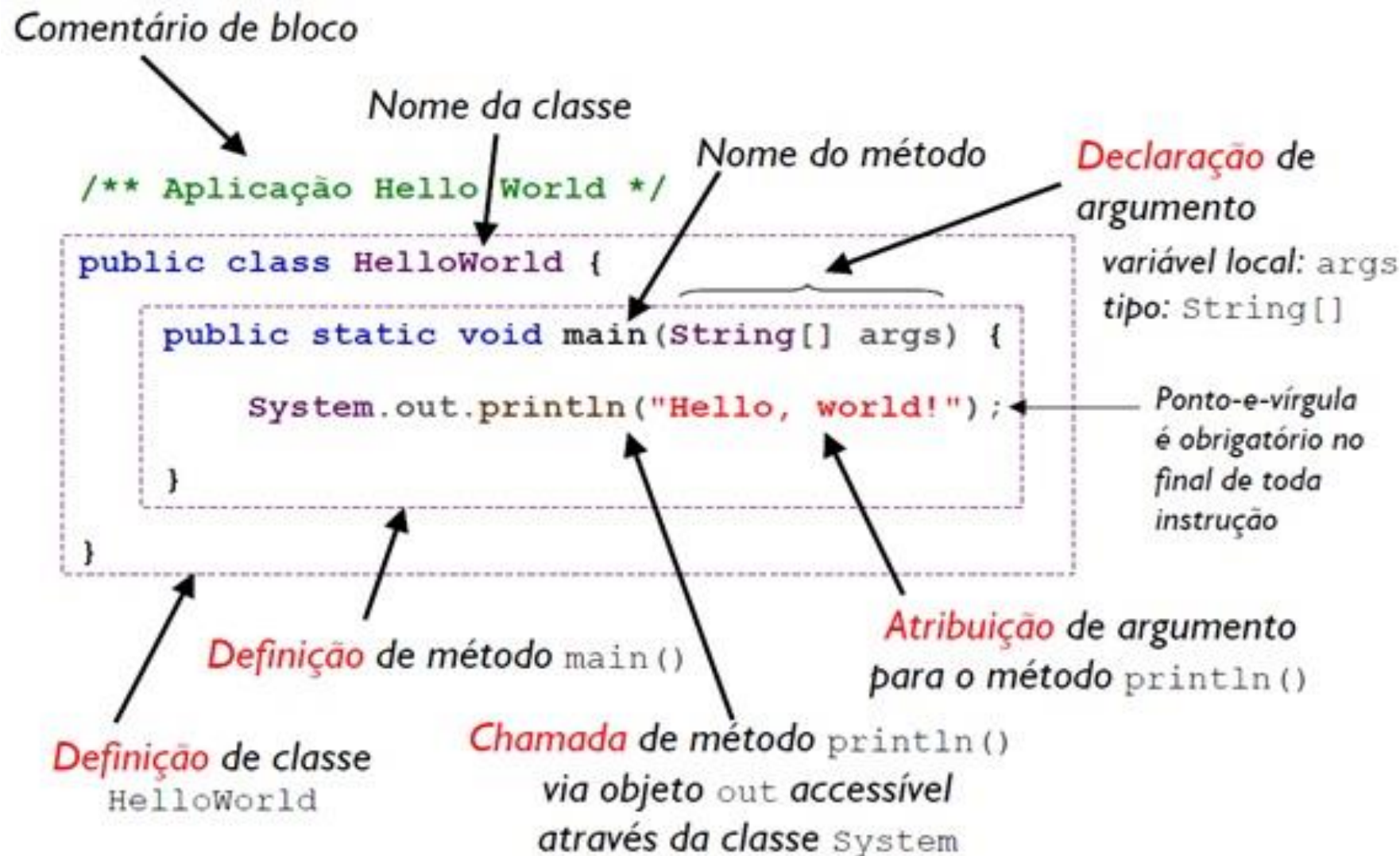
Classe. Como qualquer programa JAVA, ele exige uma classe (palavra reservada “class”). Por ser Publica (palavra “public”), isso garante visibilidade em qualquer contexto de sua utilização

Método. A impressão da mensagem “Hello, World!” se deu pela execução do método “println” da classe “System”.

Objeto. Para imprimir a mensagem de saída do programa é necessário o objeto “out” da classe “System” da biblioteca padrão java.lang

Biblioteca. A organização das classes JAVA se dá na forma de bibliotecas. Nesse programa utilizamos a biblioteca padrão da linguagem JAVA (biblioteca java.lang)

Resumo



Palavras reservadas

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

Tipo de Dados

- Tipos primitivos:

Tipo	Descrição
boolean	Pode assumir os valores true ou false
char	Representa um caractere Unicode de 16 bits para armazenar dados alfanuméricos
byte	Inteiro de 8 bits. Pode assumir valores entre -2^7 e 2^7-1 (de -128 a 127)
short	Inteiro de 16 bits. Pode assumir valores entre -2^{15} e $2^{15}-1$ (de -32.768 a 32.767)
int	Inteiro de 32 bits. Pode assumir valores entre -2^{31} e $2^{31}-1$ (de -2.147.483.648 a 2.147.483.647)
long	Inteiro de 64 bits. Pode assumir valores entre -2^{63} e $2^{63}-1$
float	Número de ponto flutuante de 32 bits. Pode assumir valores entre $\pm 1.40129846432481707e-45$ e $\pm 3.40282346638528860e+38$
double	Número de ponto flutuante de 64 bits. Pode assumir valores entre $\pm 4.94065645841246544e-324$ e $\pm 1.79769313486231570e+308$

- String:

Tipo	Descrição
String	Cadeia de caracteres que usam 2 bytes por caractere. Strings podem ser vazias (zero caractere) e conter qualquer tipo de caractere.

Declaração de Dados

Tipo Variável

`int` `pesoVeiculo;`

Nome Variável

Exemplos:

- `float precoProduto;`
- `byte idadeAluno;`
- `char sexoFuncionario;`
- `String nome_Aluno;`
- `boolean maiorIdade;`

Atribuição de Variaveis

Nome Variável

```
pesoVeiculo = 1500;
```

Valor Variável


Exemplos:

- `float precoProduto = 3.5;`
- `byte idadeAluno = 18;`
- `char sexoFuncionario = 'M';`
- `String nome_Aluno = "João Pedro";`
- `boolean maiorIdade = false;`

Operadores Aritmético

Operador Aritmético

`mediaAluno = totalNotas / 2;`



Expressão

Operação	Operador	Expressão Algébrica	Expressão Java
Adição	+	$X + 1$	<code>X + 1</code>
Subtração	-	$Y - 2$	<code>Y - 2</code>
Multiplicação	*	$K \cdot X$	<code>K * X</code>
Divisão	/	$C / 2$	<code>C / 2</code>
Resto	%	$X \bmod Y$	<code>X % Y</code>

Precedência de Operadores Aritméticos

Operador	Operação	Expressão Algébrica
* / %	Multiplicação, Divisão e Resto	É o primeiro a ser avaliado. A ordem de avaliação é da esquerda para a direita.
- +	Subtração e soma	É avaliado posteriormente. A ordem de avaliação também é da esquerda para a direita.

Operadores Relacionais

Operador Relacional

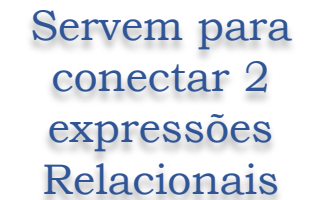
(notaAluno \geq 6)

Permite saber a relação existente entre seus dois operandos

Respostas
TRUE
ou
FALSE

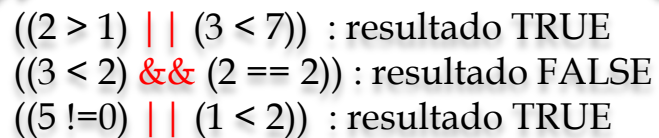
Operação	Operador Matemático	Operador Java	Exemplo	Significado
Igual	=	==	X == Y	X é igual a Y
Diferente	\neq	!=	X != Y	X é diferente de Y
Maior	>	>	X > Y	X é Maior que Y
Menor	<	<	X < Y	X é menor que Y
Maior ou Igual	\geq	>=	X >= Y	X é maior ou igual a Y
Menor ou Igual	\leq	<=	X <= Y	X é menor ou igual a Y

Operadores Lógicos



Servem para
conectar 2
expressões
Relacionais

Operação	Operador Matemático	Operador Java	Exemplo
OU	\vee	<code> </code>	<code>(notaEnem > 6) (notaRedacao == 10)</code>
E	\wedge	<code>&&</code>	<code>(mediaFinal >= 6) && (totalFaltas < 25%)</code>
Negação	\sim	<code>!</code>	<code>!pendenciaDocumento</code>



`((2 > 1) || (3 < 7)) : resultado TRUE`
`((3 < 2) && (2 == 2)) : resultado FALSE`
`((5 != 0) || (1 < 2)) : resultado TRUE`

Conversões de Variáveis

Conversões

IMPLICITAS

Nenhuma sintaxe especial é necessária porque a conversão é de tipo seguro e nenhum dado será perdido. Exemplo:

- byte para int ou long para float → `byte b = 10; int i = b;`

Conversões

EXPLICITAS

As conversões explícitas exigem um operador cast. A conversão é necessária quando as informações podem ser perdidas na conversão ou quando a conversão pode não funcionar por outros motivos. Exemplo:

- Float para long ou int para byte → `int i = 10; byte b = (byte)i;`

Conversões

OUTRAS

O restante dos tipos de conversão disponíveis ou criados no programa. Exemplo:

- String para int → `String s = "123"; int i = Integer.parseInt(s);`

Conversões de Variáveis

PARA:	byte	short	char	int	long	float	double
DE:							
byte	----	<i>Impl.</i>	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
short	(byte)	----	(char)	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
char	(byte)	(short)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
int	(byte)	(short)	(char)	----	<i>Impl.</i>	<i>Impl.</i>	<i>Impl.</i>
long	(byte)	(short)	(char)	(int)	----	<i>Impl.</i>	<i>Impl.</i>
float	(byte)	(short)	(char)	(int)	(long)	----	<i>Impl.</i>
double	(byte)	(short)	(char)	(int)	(long)	(float)	----

Impl. Significa Implícita

Entrada de Dados em Java (captura dados via teclado)

Classe **BufferedReader** para captura através de uma janela de console (entrada pelo teclado)

Classe **Scanner** para captura através de uma janela console (entrada pelo teclado)

Classe **JOptionPane** para captura através de interface gráfica (entrada pelo teclado)

```
package pi;

import java.util.Scanner;

/**
 *
 * @author Fernando
 */
public class GetInputFromScanner {
    public static void main(String arg[]){
        Scanner sc = new Scanner(System.in);
        System.out.println("Please Enter Your Name:");
        String name = sc.next();
        System.out.println("Hello " + name + "!");
    }
}
```

import java.util.Scanner;

Scanner sc = new Scanner(System.in);

String name = sc.next();

Classe Scanner

Engloba diversos métodos para facilitar a leitura de dados pelo teclado

Método	Finalidade
next()	Aguarda uma entrada em formato String
nextInt()	Aguarda uma entrada em formato Inteiro
nextByte()	Aguarda uma entrada em formato Inteiro
nextLong()	Aguarda uma entrada em formato Inteiro Longo
nextFloat()	Aguarda uma entrada em formato Número Fracionário
nextDouble()	Aguarda uma entrada em formato Número Fracionário

Classe Scanner

Métodos da classe **Scanner** para obter dados

Classe Scanner

```
7 package program1;
8
9 import java.util.Scanner;
10
11 /*
12  * Este programa calcula a média de três valores
13  */
14 public class Programa1 {
15
16     public static void main(String[] args) {
17         // Variáveis
18         int valorA;
19         int valorB;
20         int valorC;
21         int media;
22
23         // O Scanner para leitura dos valores
24         Scanner leitor = new Scanner(System.in);
25
26         System.out.println("Digite o primeiro valor:");
27         valorA = leitor.nextInt();
28
29         System.out.println("Digite o segundo valor:");
30         valorB = leitor.nextInt();
31
32         System.out.println("Digite o terceiro valor:");
33         valorC = leitor.nextInt();
34
35         media = (valorA + valorB + valorC) / 3;
36
37         System.out.println("A média é: " + media);
38     }
39 }
```

The diagram illustrates the usage of the `Scanner` class in a Java program. It shows the following code snippets highlighted with red boxes and arrows:

- `import java.util.Scanner;` (Line 9)
- `Scanner leitor = new Scanner(System.in);` (Line 24)
- `valorA = leitor.nextInt();` (Line 27)

Atividades

- Escreva uma classe para calcular a média final dos alunos do 5º semestre. Os alunos realizarão 4 provas: P1, P2, P3 e P4, onde a média é a soma das 4 notas, dividido por 4.
 - Quais são os dados de entrada?
 - Qual será o processamento a ser realizado?
 - Quais são os dados de saída?

Atividades

1. Na empresa onde trabalhamos, há tabelas com o quanto foi gasto em cada mês. Para fechar o balanço do primeiro trimestre, precisamos somar o gasto total. Sabendo que, em Janeiro, foram gastos 15000 reais, em Fevereiro, 23000 reais e em Março, 17000 reais, faça um programa que calcule e imprima o gasto total no trimestre. Siga esses passos:
 - a. Crie uma classe chamada `BalancoTrimestral` com um bloco `main`, como nos exemplos anteriores;
 - b. Dentro do `main` (o miolo do programa), declare uma variável inteira chamada `gastosJaneiro` e inicialize-a com 15000;
 - c. Crie também as variáveis `gastosFevereiro` e `gastosMarco`, inicializando-as com 23000 e 17000, respectivamente, utilize uma linha para cada declaração;
 - d. Crie uma variável chamada `gastosTrimestre` e inicialize-a com a soma das outras 3 variáveis:

```
int gastosTrimestre = gastosJaneiro + gastosFevereiro + gastosMarco;
```
 - e. Imprima a variável `gastosTrimestre`.

Atividades

2. Adicione código (sem alterar as linhas que já existem) na classe anterior para imprimir a média mensal de gasto, criando uma variável `mediaMensal` junto com uma mensagem. Para isso, concatene a `String` com o valor, usando `"Valor da média mensal = " + mediaMensal`.

Atividades