Linguagem de Programação

CONCEITOS BÁSICOS

Prof. Me. Humberto Zanetti

Conceitos básicos

- Lógica
  - Lógica de programação
- Algoritmos
  - Fluxograma (diagrama de blocos)
  - Português estruturado (pseudocódigo)
- Linguagens de Programação
- Paradigmas
  - Prog. Estruturada
  - Prog. Orientada a Objeto

Java (histórico)

- A Sun Microsystems, em 1991, deu início ao **Green Project** chefiado por James Gosling. Projeto que apostava na convergência dos computadores com outros equipamentos e eletrodomésticos.
- Foi desenvolvido o *7 (StarSeven), um controle remoto com uma interface gráfica *touchscreen* com aplicativos desenvolvidos na linguagem Oak.
- Foi adaptado para desenvolvimento de aplicações para web (conhecidos como *applets*) e foi rebatizada como Java.

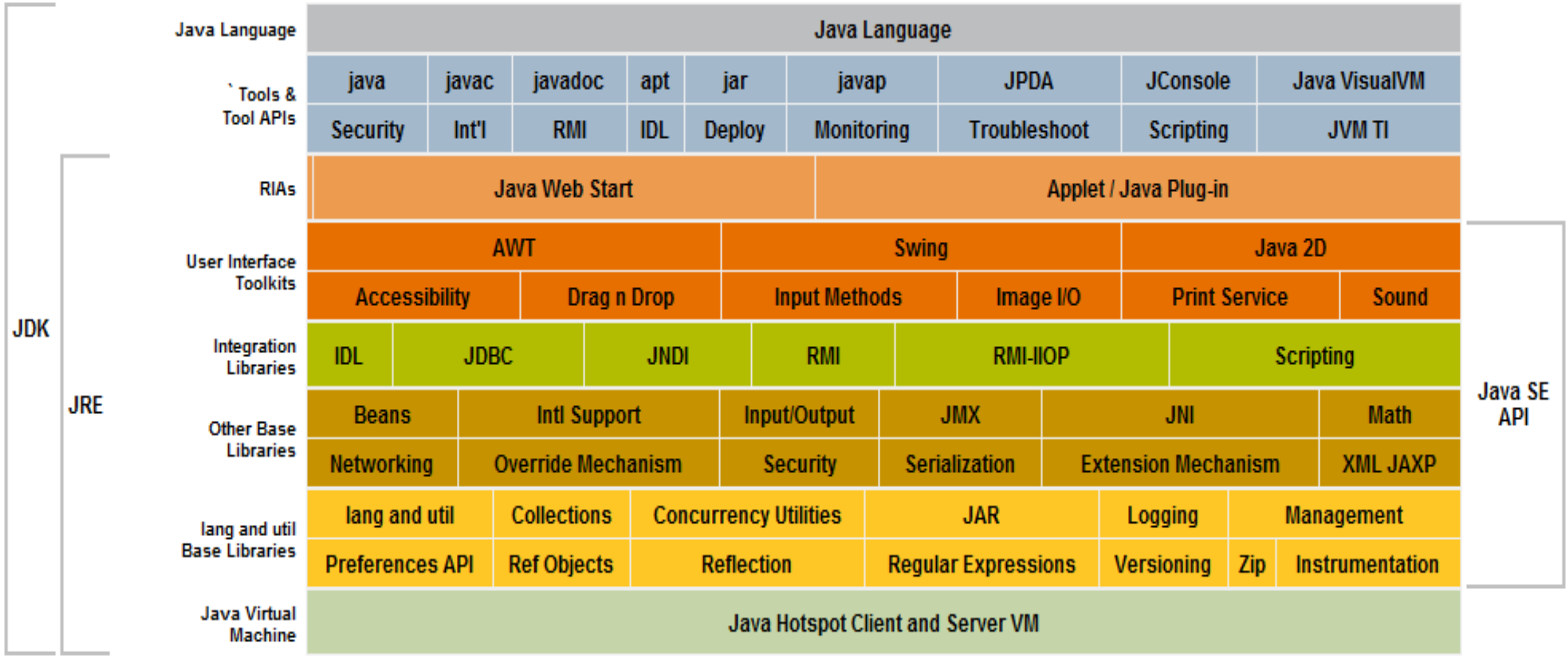
Java

- Em 2009, a Oracle comprou a Sun Microsystems por de 7,4 bilhões de dólares

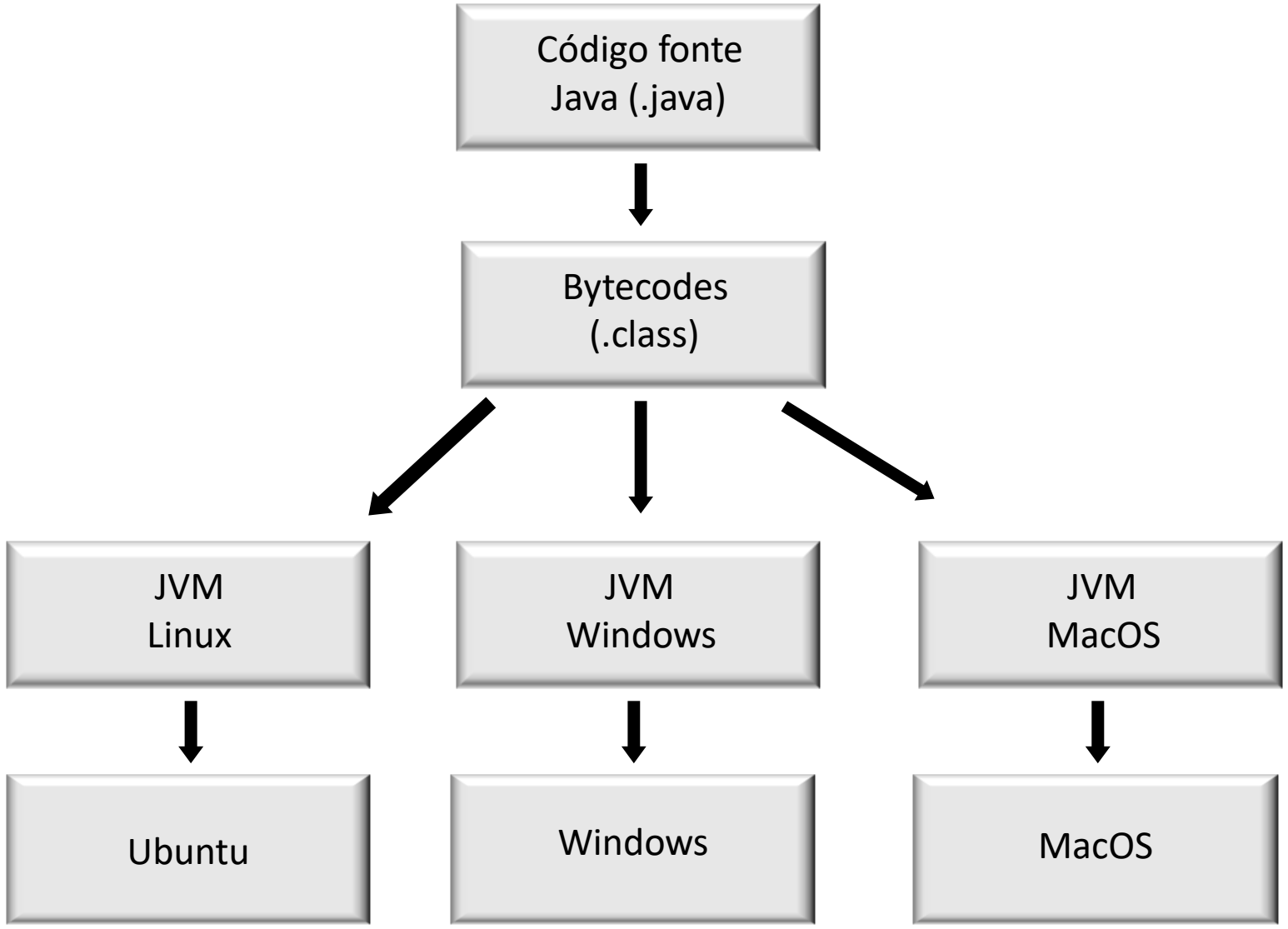
Características

- Totalmente POO
- Portabilidade
- Modularidade
- Reusabilidade
- Facilidade de manutenção e expansão
- Produtividade
- Processamento distribuído
- Multi-threading
- Tratamento de erros
- Armazenamento dinâmico de memória

“Cenário Java”



Java Virtual Machine



Mercado

Feb 2022	Feb 2021	Change	Programming Language	Ratings	Change
1	3	▲	Python	15.33%	+4.47%
2	1	▼	C	14.08%	-2.26%
3	2	▼	Java	12.13%	+0.84%
4	4		C++	8.01%	+1.13%
5	5		C#	5.37%	+0.93%
6	6		Visual Basic	5.23%	+0.90%
7	7		JavaScript	1.83%	-0.45%
8	8		PHP	1.79%	+0.04%
9	10	▲	Assembly language	1.60%	-0.06%
10	9	▼	SQL	1.55%	-0.18%

TIOBE index – Fevereiro de 2022: https://www.tiobe.com/tiobe-index/

Tipos primitivos de dados

- **boolean**: verdadeiro ou falso
- **byte**: um inteiro de 8 bits.
- **char**: um caracter unicode (16-bit unsigned).
- **double**: um número de ponto flutuante de 64 bits.
- **float**: um número de ponto flutuante de 32 bits.
- **int:** um inteiro de 32 bits.
- **long**: um inteiro de 64 bits.
- **short:** um inteiro de 32 bits.

Declaração de variáveis

```
// Declaração de variáveis
int num1 = 0, op = 0;
double valor;
String usuario;

// Declaração de constantes
final double pi = 3.1416;
```

- Tipos primitivos são escritos sempre com letras minúsculas.
- As variáveis podem (e em alguns casos devem) ser inicializadas na declaração
- O Java disponibiliza algumas classes que podem ser utilizadas como tipos (como a String no exemplo acima).

Conversão de tipos

Supondo a variável x	Converter em	y recebe o valor convertido
✓ **Entre tipos numéricos**		
int x = 10	float	float y = (float) x
int x = 10	double	double y = (double) x
float x = 10.5	int	int y = (int) x
✓ **De string para numéricos**		
String x = “10”	int	int y = Integer.parseInt(x)
String x = “20.5”	float	float y = Float.parseFloat(x)
String x = “20.5”	double	double y = Double.parseDouble(x)
✓ **De numéricos para string**		
int x = 10	String	String y = Integer.toString(x) ou String y = String.valueOf(x)
float x = 10.5	String	String y = Float.toString(x) ou String y = String.valueOf(x)
double x = 10.5	String	String y = Double.toString(x) ou String y = String.valueOf(x)

Operadores aritméticos

Função	Sinal
Adição	+
Subtração	-
Multiplicação	*
Divisão	/
Resto da divisão	%
Incremento	++
Decremento	--

Operadores relacionais

Função	Sinal
Igual	==
Diferente	!=
Maior que	>
Maior ou igual a	>=
Menor que	<
Menor ou igual a	<=

Operadores lógicos

Função	Sinal
E	&&
OU	||
Não	!

Estrutura de condicional if-else

```
if (num1>=10) {
    System.out.println("Condição verdadeira!");
}else{
    System.out.println("Condição falsa!");
}
```

Estrutura condicional switch-case

```
switch (op) {
case 1:
    System.out.println("Caso op igual a 1...");
    break;
case 2:
    System.out.println("Caso op igual a 2...");
    break;
case 3:
    System.out.println("Caso op igual a 3...");
    break;
default:
    System.out.println("Caso op não seja 1, 2 ou 3");
    break;
}
```

Laços de repetição

```
// Teste condicional no inicio
while(op != 0){
    // Instruções
}


// Teste condicional no fim
do{
    // Instruções
}while(op != 0);


// Numero pré-definido de interações
for(int i=0; i<=10; i++){
    // Instruções
}
```

Comentários em código

```
// Comentários em uma única linha


/* Comentários em
 * várias linhas
 */


/** Comentários inseridos no formato reconhecido
 *  por um utilitário de documentação chamado javadoc
 *  fornecido pela Sun junto com o JDK
 */
```

Entrada de dados - *Scanner*

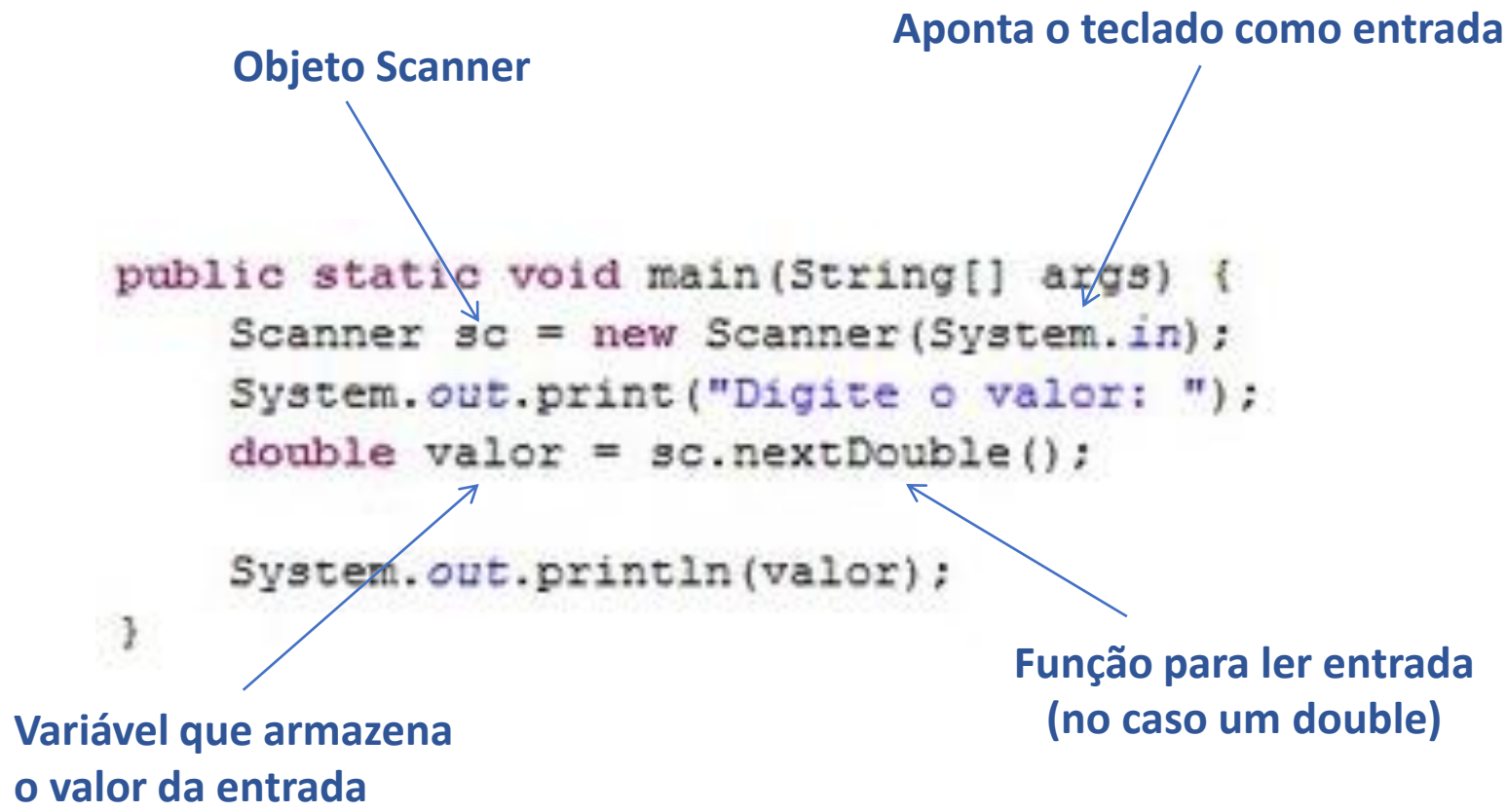
- A classe **Scanner** tem como objetivo separar a entrada dos textos em blocos, gerando os conhecidos *tokens*, que são sequências de caracteres separados por delimitadores que por padrão correspondem aos espaços em branco, tabulações e mudança de linha.
- Com essa classe podem ser convertidos textos para **tipos primitivos e Strings.**

Usando Scanner - Importante

- Se faz necessário importar o conjunto para instanciar um objeto da classe Scanner. Quando invocada a classe Scanner, o compilador pedirá para fazer a seguinte importação:

```
import java.util.Scanner;
```

Usando Scanner – Criando objeto



Métodos de leitura

```
Scanner sc = new Scanner(System.in);

float numF = sc.nextFloat();
int num1 = sc.nextInt();
byte byte1 = sc.nextByte();
long lg1 = sc.nextLong();
boolean b1 = sc.nextBoolean();
double num2 = sc.nextDouble();
String nome = sc.nextLine();
char letra = sc.next.charAt(0);
```

Saída de dados – println

- Um método de saída é utilizado para mostrar alguma coisa ao usuário. No caso de Java, existe um método padrão que utiliza uma saída padrão (monitor de vídeo) para criar a comunicação com o usuário - estamos falando de **println**.
- O método **println** faz parte da classe System que controla a maioria das funcionalidades do computador, e se tratando de uma saída, mais especificamente de ***out***.
- O método **println** imprime na tela uma String (cadeia de caracteres) que é passada ao método como argumento entre parênteses.

```
System.out.println("string")
```

Meios de utilização

Imprimindo apenas uma frase

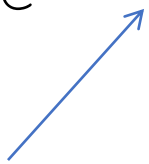
```
System.out.println("Saída Padrão em Java");
```

Imprimindo uma variável

```
int inteiro = 200;
System.out.println (inteiro);
```

Imprimindo uma frase com variável

```
System.out.println ("O número é " + inteiro);
```



Sinal de concatenação
(juntar e formar apenas um String)

Formatação de saída

- Controle de texto
  - **\n** - pula uma linha
  - **\t** - cria uma tabulação (o mesmo que apertar TAB)
  - **\b** - retorna o cursor em um caracter
  - **\r** - retorna o cursor ao início da linha

```
System.out.print("Aula:\t"); // cria uma tabulação

System.out.print("Aula:\n"); //pula uma linha

System.out.print("Aula de Java é demais! #sqn \r");// põe o cursor no início
                                                    da linha
```

Dúvidas!?