

Java Orientado a Objeto

Introdução à linguagem Java

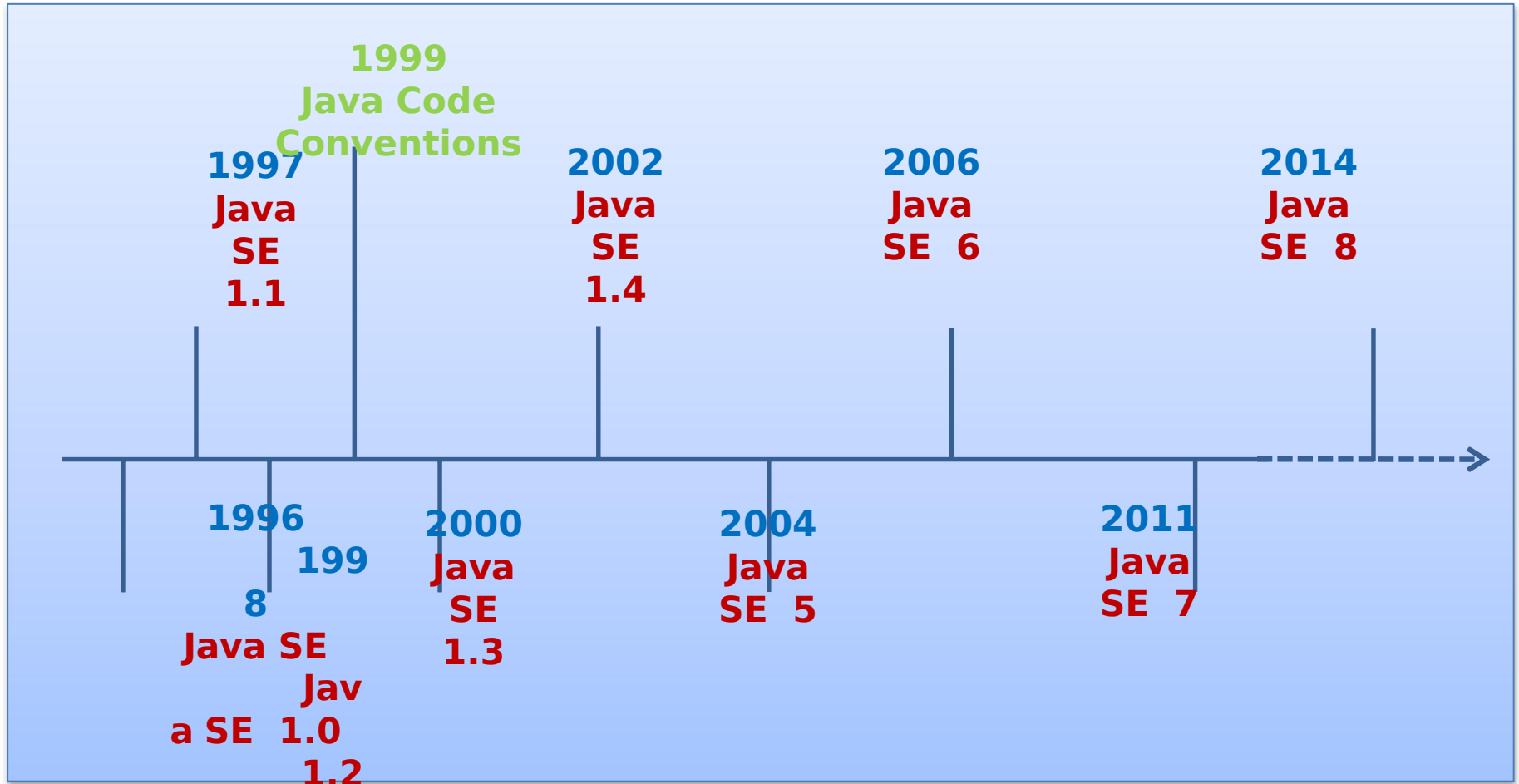




O que é Java?



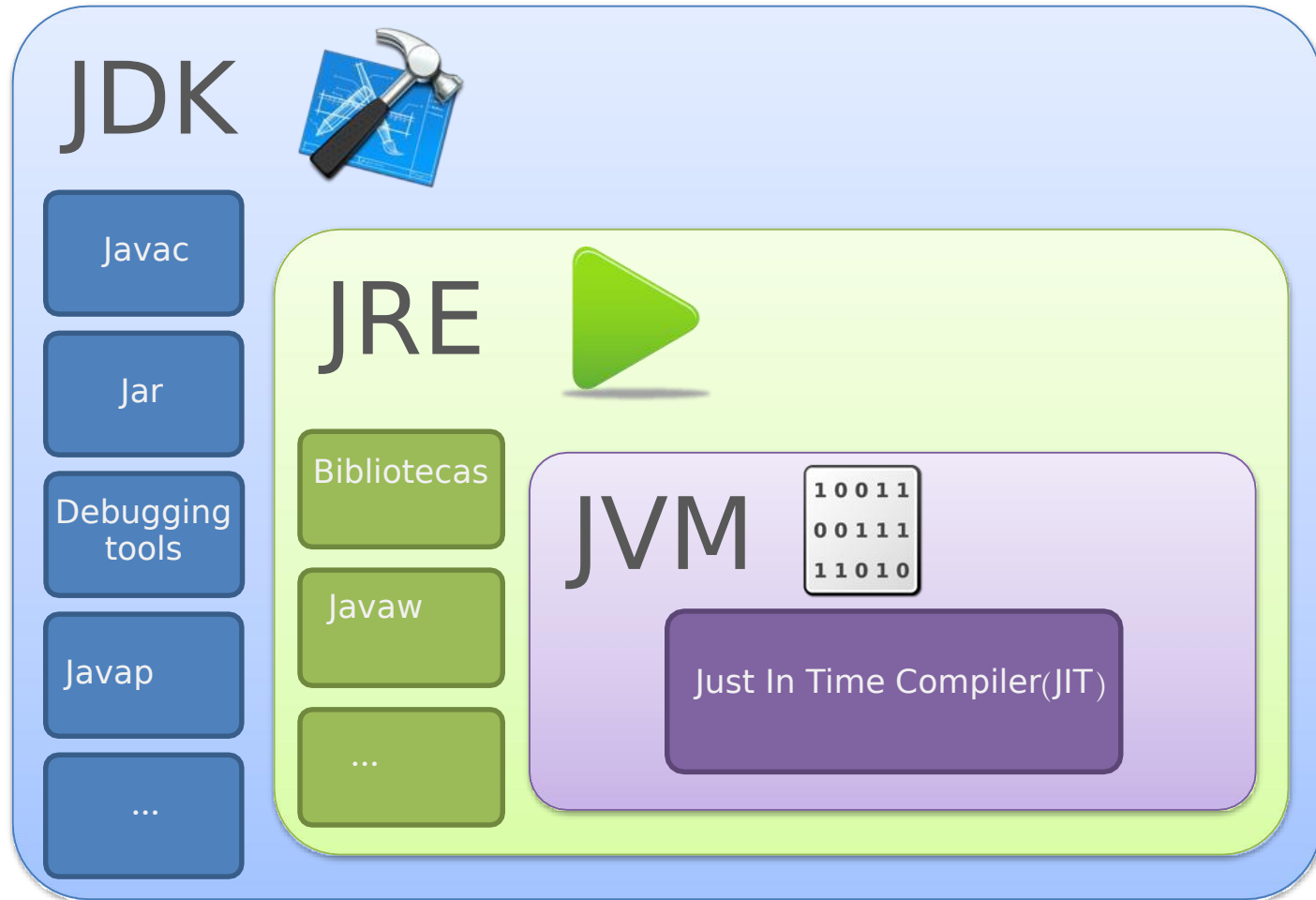
Um pouco da História



Onde encontro Java ?



Plataforma Java



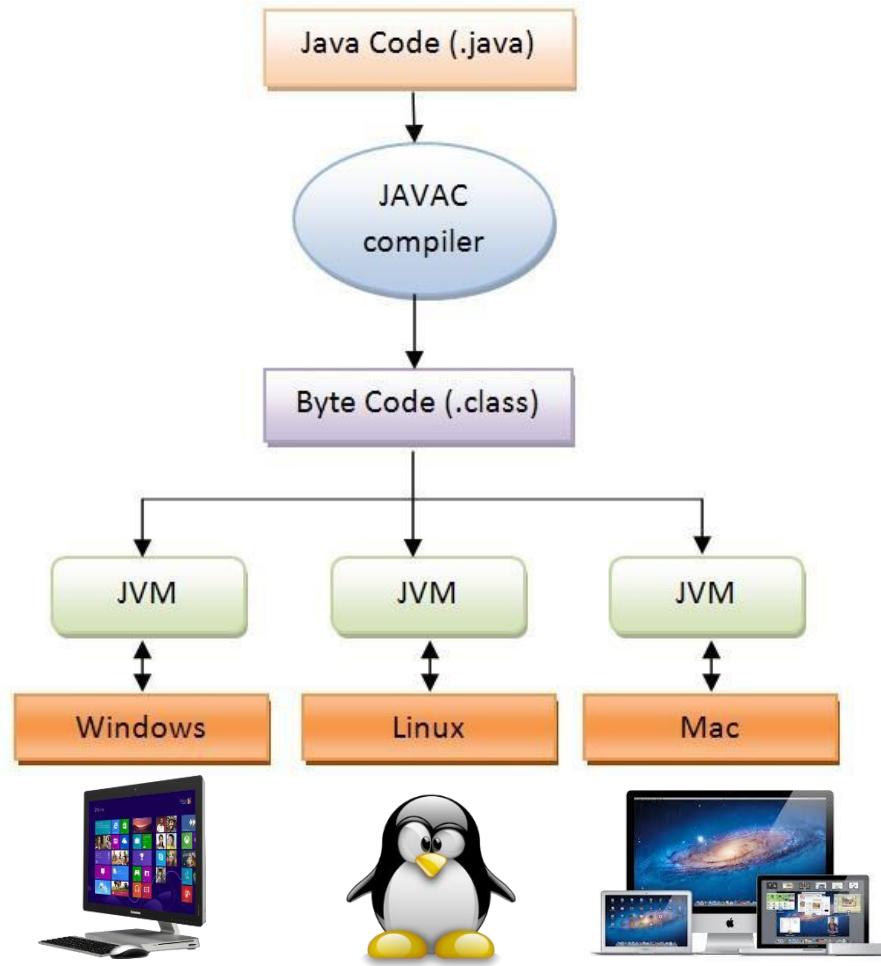
Garbage Collection (Coletor de lixo)



Divisão da Plataforma



Fases do Programa Java



Aplicações mais comuns em Java



→ Applets

→ Servlets

→ JSP

→ JSF

→ EJB

→ JPA

O método main()



Visível a
todos

Não é
instanciado

Nome do
método

Array- Estrutura de
dados que
armazena
os

Nome da
Array

```
public static void main(String[] args) {}
```

Não retorna
nada

Tipo de
argumento

Corpo do
método



Baby Steps



Java Orientado a Objetos

Identificadores, palavras-chave e tipo



A word cloud of Java keywords and identifiers. The words are arranged in a dense, overlapping cluster. The colors of the words are primarily blue, green, and purple. The words include: case, break, byte, abstract, boolean, catch, assert, double, const, float, native, return, static, throw, throws, try, volatile, void, transient, synchronized, throwstrictfp, switch, package, private, public, new, super, int, long, enum, final, finally, instanceof, implements, goto, null, protected, interface, extends, import, char, default, continue, class, else, and instanceof.

JavaDoc

```
/**  
 * Exemplo básico de um comentário em JavaDoc  
 * Com mais de uma linha.  
 */
```

@author

@link

@deprecated

@param

@return

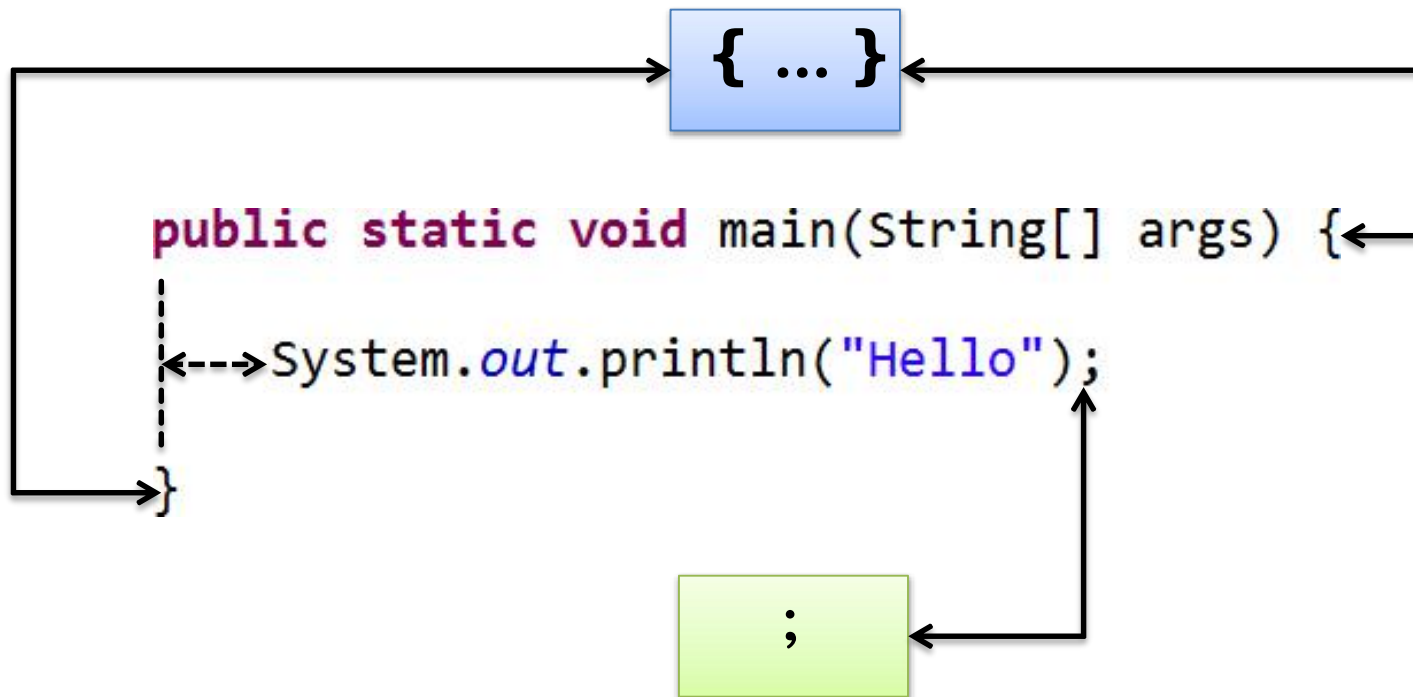
@see

@since

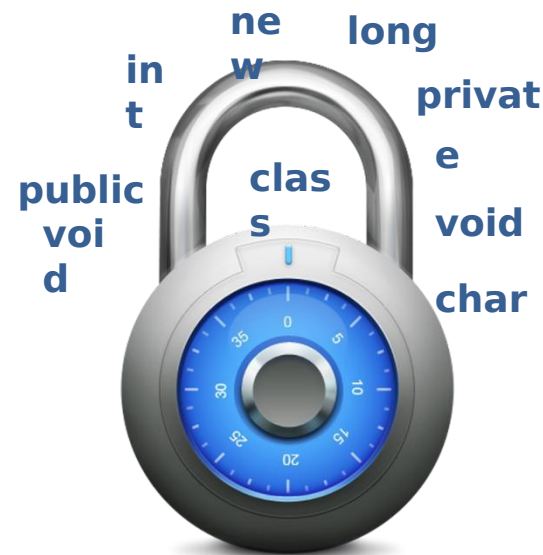
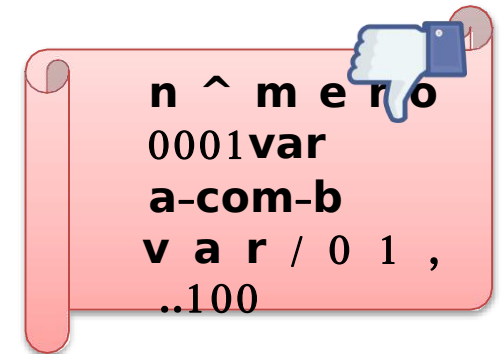
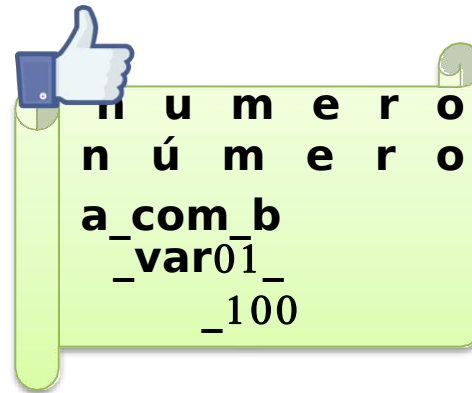
@throws

@version

Ponto-e-Vírgula, Blocos e Espaço



Identificadores e Palavras Reservadas

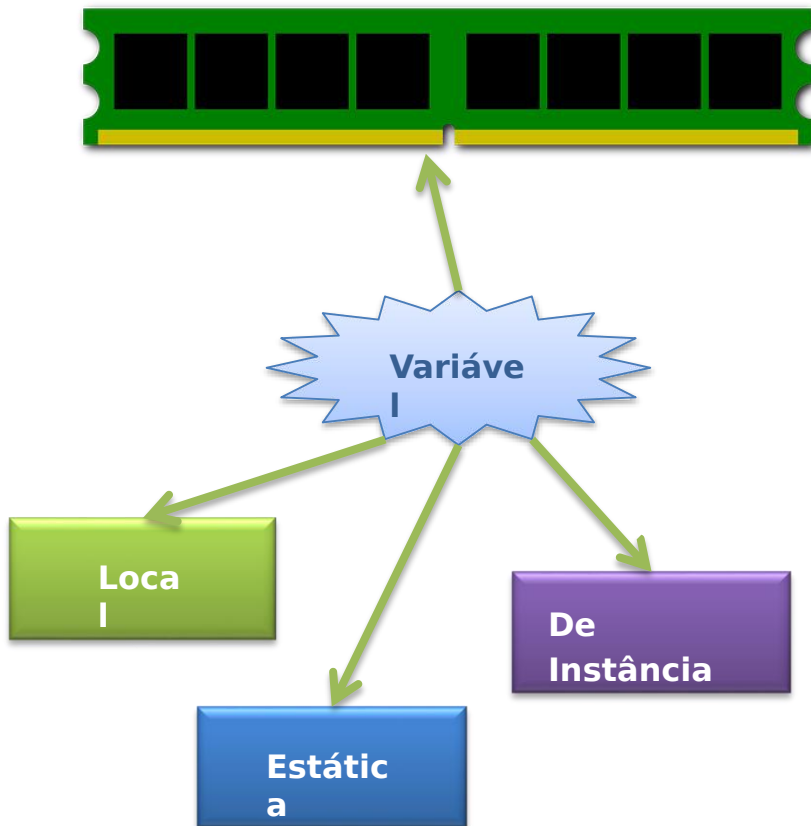


Variáveis, Declaração e Atribuição

Atribuindo Variável

<tipo do dado | **<nome** | = [**valor inicial**] ;

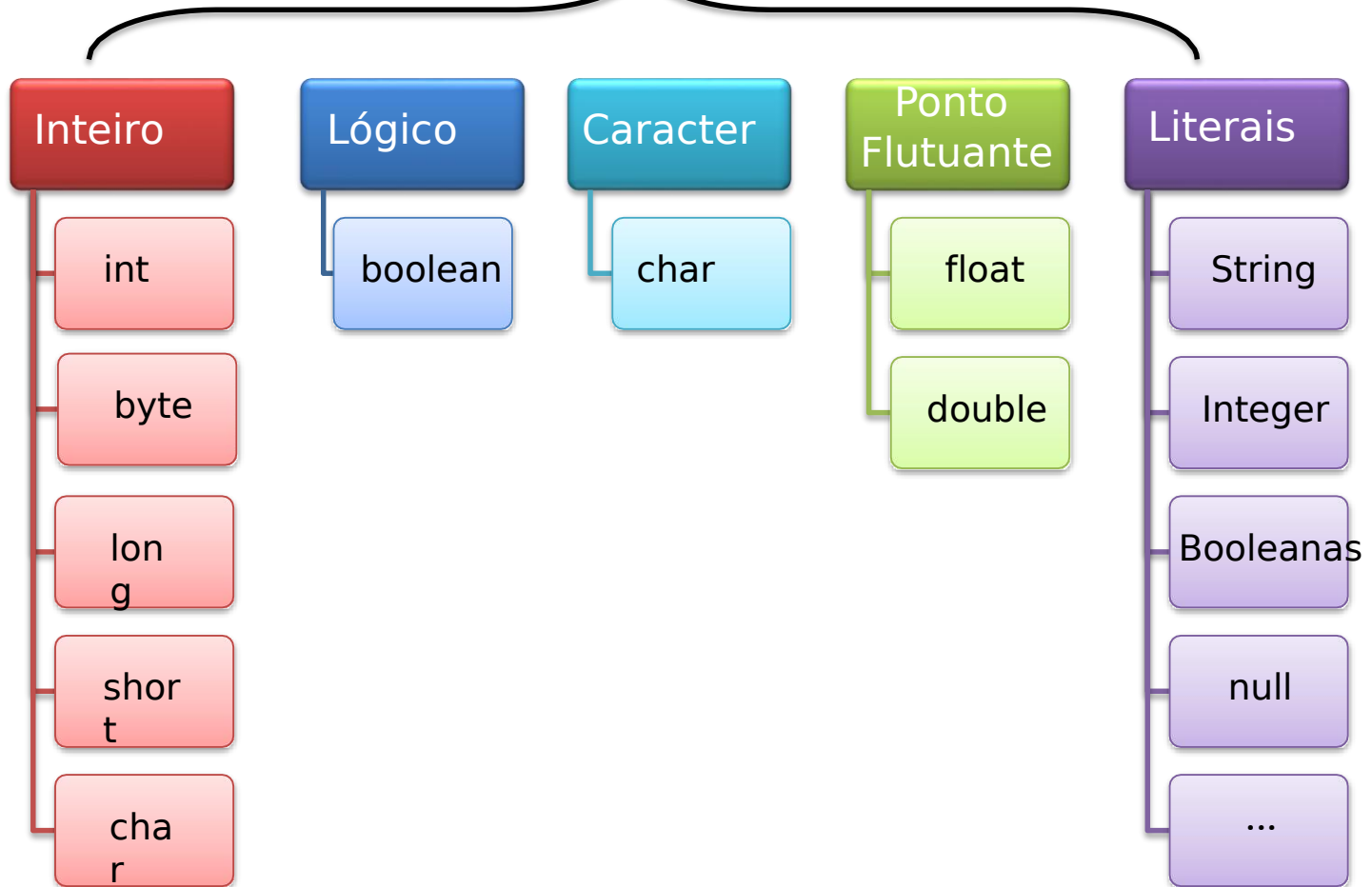
Declarando Variável



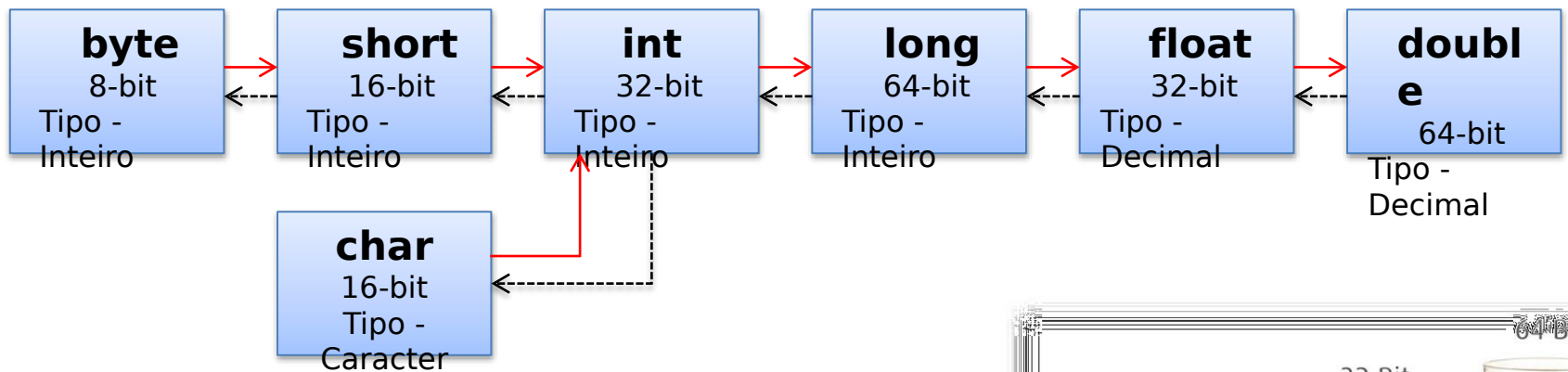
Variáveis devem ser declaradas e inicializadas antes de serem utilizadas.

Tipos de Dados

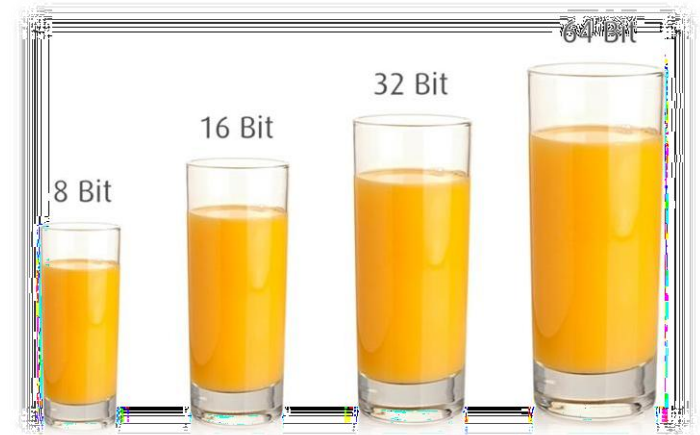
Tipos Primitivos



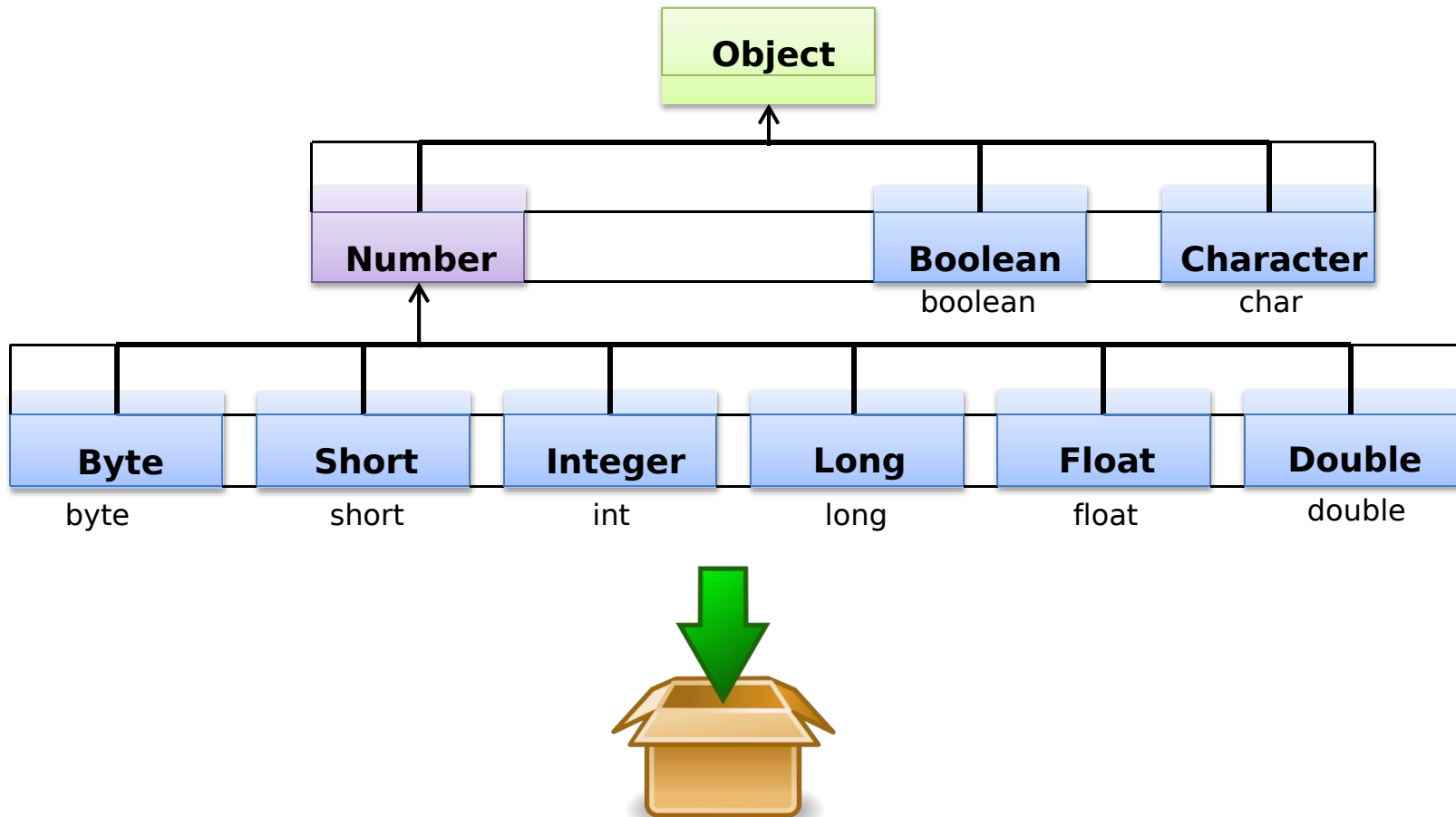
Casting de Tipos Primitivos



→ Casting implícito (Automático)
-----> Casting explícito (Requer a utilização de cast)



Classes Wrapper (Empacotadoras)



Construtores e método valueOf

```
Float variavelFloat =
```

+



Construtores

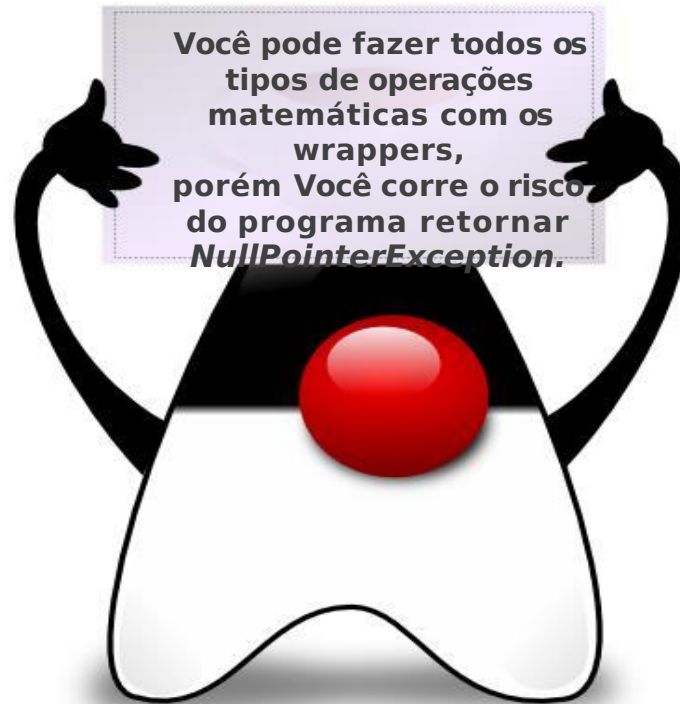
```
new Float(1.1f);  
new Float(1.1);  
new Float("1.1");  
new Float("1.1f");
```

Método valueOf

```
Float.valueOf("1.1f");  
Float.valueOf("1.1");
```

AutoBoxing – Boxing and Unboxing

```
int i = 10;  
Integer iRef = new Integer(i); // Boxing Explícito  
int j = iRef.intValue(); // Unboxing Explícito  
iRef = i; // Boxing Automático  
j = iRef; // Unboxing Automático
```



Java Orientado a Objetos

Operadores



Operadores Aritméticos



| | | |
|---|----------------|---------------------|
| * | var1 * var2 | Multiplicação |
| + | var1 + var2 | Adição |
| - | var1 - var2 | Subtração |
| % | var1 % var2 | Resto da divisão |
| / | var1 / var2 | Divisão |

Operadores Relacionais

!=

|
==
<=
|
|
<

| | | |
|----|--------------|----------------|
| | var1 var2 | Maior que |
| != | var1 != var2 | Diferente |
| >= | var1 >= var2 | Menor ou igual |
| < | var1 < var2 | Menor que |
| > | var1 > var2 | Maior ou igual |
| == | var1 == var2 | Igual |

Operadores Lógicos

&
&

&

&

!

^

||

|

| | | |
|----|-----------------|--------------------------|
| && | var1 &&var2 | 'E' lógico (AND) |
| & | var1 & var2 | 'E' binário |
| | var1 var2 | 'OU' lógico (OR) |
| | var1 var2 | 'OU' binário |
| ^ | var1 ^ var2 | OU' exclusivo binário |
| ! | var1 ! var2 | Negação (NOT) |

&& (e lógico) e & (e binário)

| Condição 1 | Operador | Condição 2 | Resultado |
|---------------|----------|---------------|-----------|
| true | && | true | true |
| false | && | true | false |
| true | && | false | false |
| false | && | false | false |



A diferença básica do operador && para & é que o && suporta uma avaliação de curto-circuito (ou avaliação parcial), enquanto que o & não.