## 2 – A tecnologia Java

Uma ideia base da linguagem JAVA é a de que,

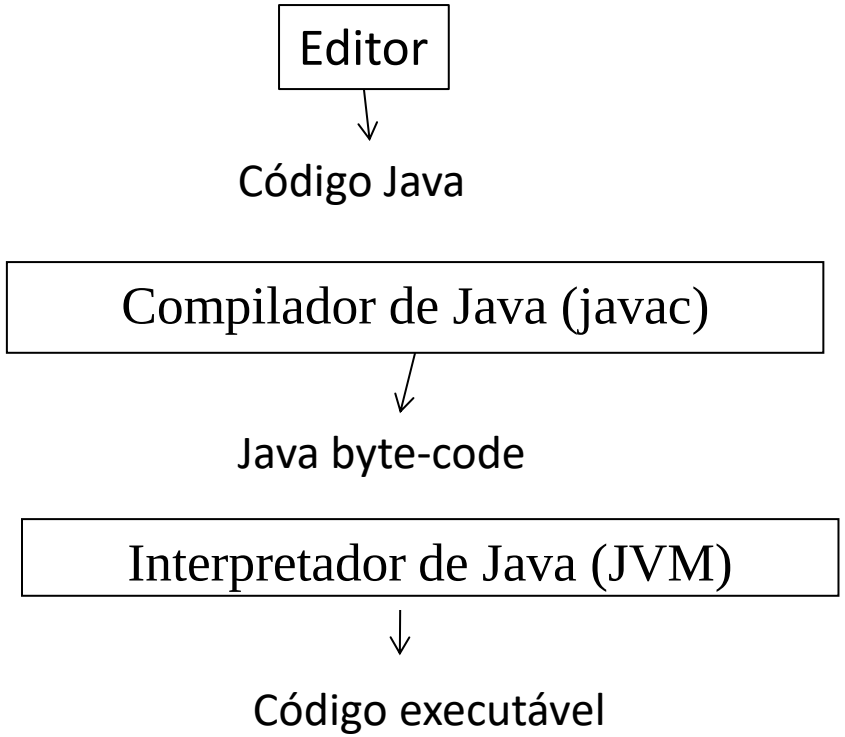
"um programa em JAVA deve poder ser executado em qualquer tipo de computador sem ter que ser alterado ou compilado novamente".

Em Java o código fonte da aplicação é compilado para uma representação intermédia, independente do sistema de execução e da arquitectura da máquina.

Essa representação intermédia é designada por byte-code.

De seguida este código pode ser interpretado sobre o ambiente de cada máquina específica.

Para cada plataforma em que se pretende executar um programa em Java é necessário um "motor de execução" designado por Java Virtual Machine (JVM).
A JVM recebe byte-code e transforma-o em instruções executáveis na máquina onde o ambiente Java é instalado.



O JAVA pode ser usado para criar dois tipos de programas:

Aplicações e Applets.

Aplicações Java são programas que após serem compilados apenas requerem uma JVM para serem interpretados e executados.

Applets são porções de código Java não executável por si próprio. Requerem a existência de um "browser" que incorpore e execute a JVM.

**A linguagem Java**

A linguagem Java foi criada no início dos anos 90 por uma equipa da Sun Microsystems liderada por James Gosling.

Aquela equipa ao pretender desenvolver software para pequenos equipamentos de electrónica deparou-se com dificuldades em encontrar uma linguagem adequada para criar programas que pudessem ser executados em dispositivos com arquitecturas muito diferentes.

A tentativa de criar uma linguagem independente da arquitectura a programar deu origem ao Java (inicialmente designada por Oak).

Java é uma linguagem Orientada a Objectos de uso geral

Java segue a sintaxe e a estrutura geral do C++

A JVM está embutida nos principais “Web-browsers”

Java começou por ser uma marca comercial da Sun Microsystems

Em 2010 a Sun foi comprada pela Oracle Corporation

Java é também um esforço de marketing bem conseguido pela Sun

**O nome “Java”**

Java é o nome de uma ilha da Indonésia
(130.000Km2, 90.000.000 de habitantes, capital Jakarta)

Java é o nome de um tipo de café criado em Java e ilhas vizinhas

Java é uma palavra inventada pela Sun para substituir o nome inicial da linguagem (Oak)

**Classes em Java**

- Em Java os programas são constituídos por diversas classes
- Algumas classes são escritas por nós, outras fazem parte da biblioteca
- As classes são agrupadas em pacotes (packages)
- As classes possuem atributos (variáveis) e métodos (funções)
- As classes são tipos de dados.
- Em Java cada objecto pertence a um determinado tipo. O tipo de um objecto é a sua classe.
- Algumas classes são *Applets* e podem ser executadas num *browser*

**Classes em Java**

- Em Java os programas são constituídos por diversas classes.

- Algumas classes são escritas por nós, outras fazem parte da biblioteca.

- As classes são agrupadas em pacotes (packages).

- As classes possuem atributos (variáveis) e métodos (funções).

**Classes em Java**

•As classes são tipos de dados.

• Em Java cada objecto pertence a um determinado tipo. O tipo de um objecto é a sua classe.

• Algumas classes são *Applets* e podem ser executadas num *browser*

Um **construtor** é uma operação especial da classe

- Serve para criar um objecto da classe
- Todos os construtores de uma classe têm o mesmo nome que a classe
- Os construtores são identificados pela sua lista de parâmetros
- Um construtor é uma função em que não é declarado o tipo do resultado (o resultado é um objeto da classe).

**Tipos referenciados – Objectos e Arrays**

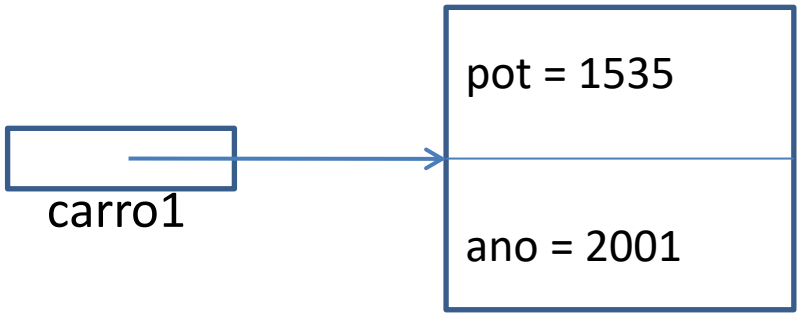
Designam-se por tipos referenciados entidades (objectos ou arrays) que são acedidas através de uma variável que contém o seu endereço.

Note-se que, sendo estes objectos referenciados através de variáveis, atribuições entre estas variáveis assumem uma *semântica de apontadores,* ou seja, o que é atribuído ou manipulado são *referências (isto é, endereços).*

**Objectos**

Veiculo carro1; *// é reservado um espaço de memória, //associado ao identificador carro1, e é-lhe //atribuído o valor null.*

carro1 = **new** Veiculo(1535, 2001); // invocar um construtor;

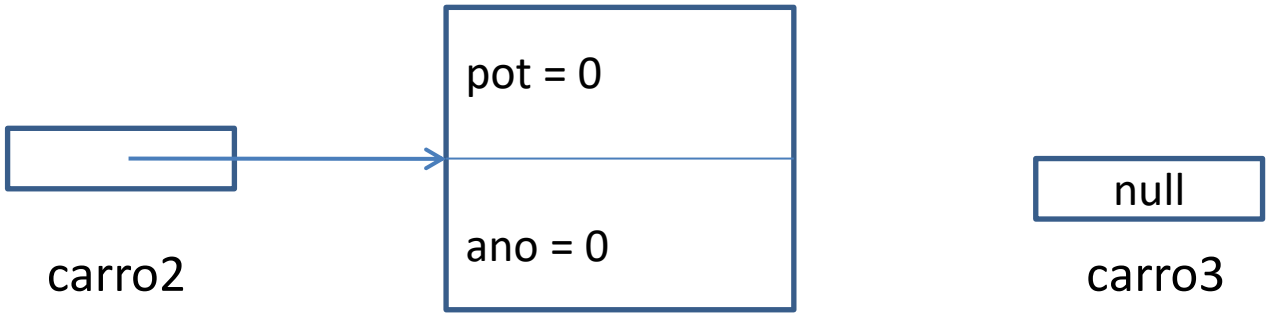



// é criado (instanciado) um objecto do tipo Veiculo e o seu endereço é colocado em carro 1.

Suponhamos:

```
Veiculo carro2 = new Veiculo();

Veiculo carro3;
```



O que significam as seguintes instruções?

carro2 = carro1; **?**

carro1 = carro3; **?**

carro1 = carro2; **?**

carro1 == carro2; **?**

Veiculo carro4 = **new** Veiculo(1535, 2001);

carro1 == carro4; **?**

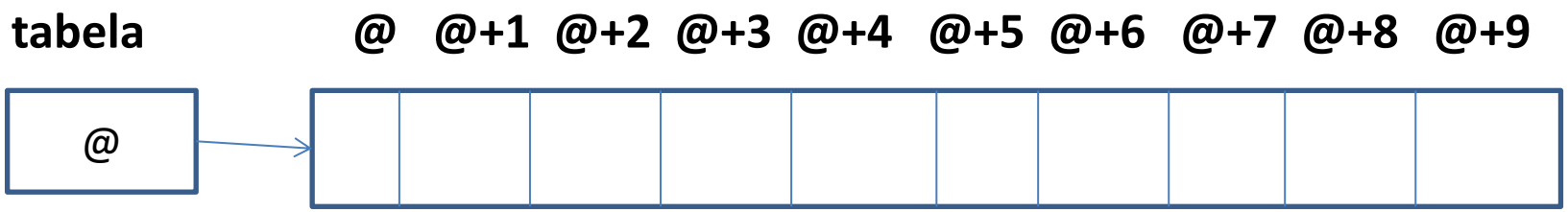
**Arrays**

Os arrays são entidades *referenciadas* mas *não são objectos*

A criação de tabelas (arrays) assemelha-se à criação de objectos

```
int [] tabela;
tabela = new int [10];

        ou
int [] tabela = new int [10];
```



Os arrays são criados dinamicamente (em tempo de execução) e o seu espaço é automaticamente reaproveitado quando deixam de estar referenciados

**Exemplos:**

```
int lista[]; // declaração "à la C"

int[] lista; // declaração equivalente à anterior

int[] lista = { 10, 12, 33, 23, 56, 67, 89, 12 };
/* declaração com inicialização */

int[] lista = new int[20];
// array de inteiros com 20 componentes

byte[] pixels = new byte[600*800];
/* pixels é um array de 480000 bytes */
```

```
String[] texto = new String[200];
/* texto é um array de 200 Cadeias de caracteres */

int[] potencias2 = {2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096};

String [] moedas = { "Dólar", "Yene", "Euro", "Florim" };
// String é um tipo de objecto (classe String) !!!
```

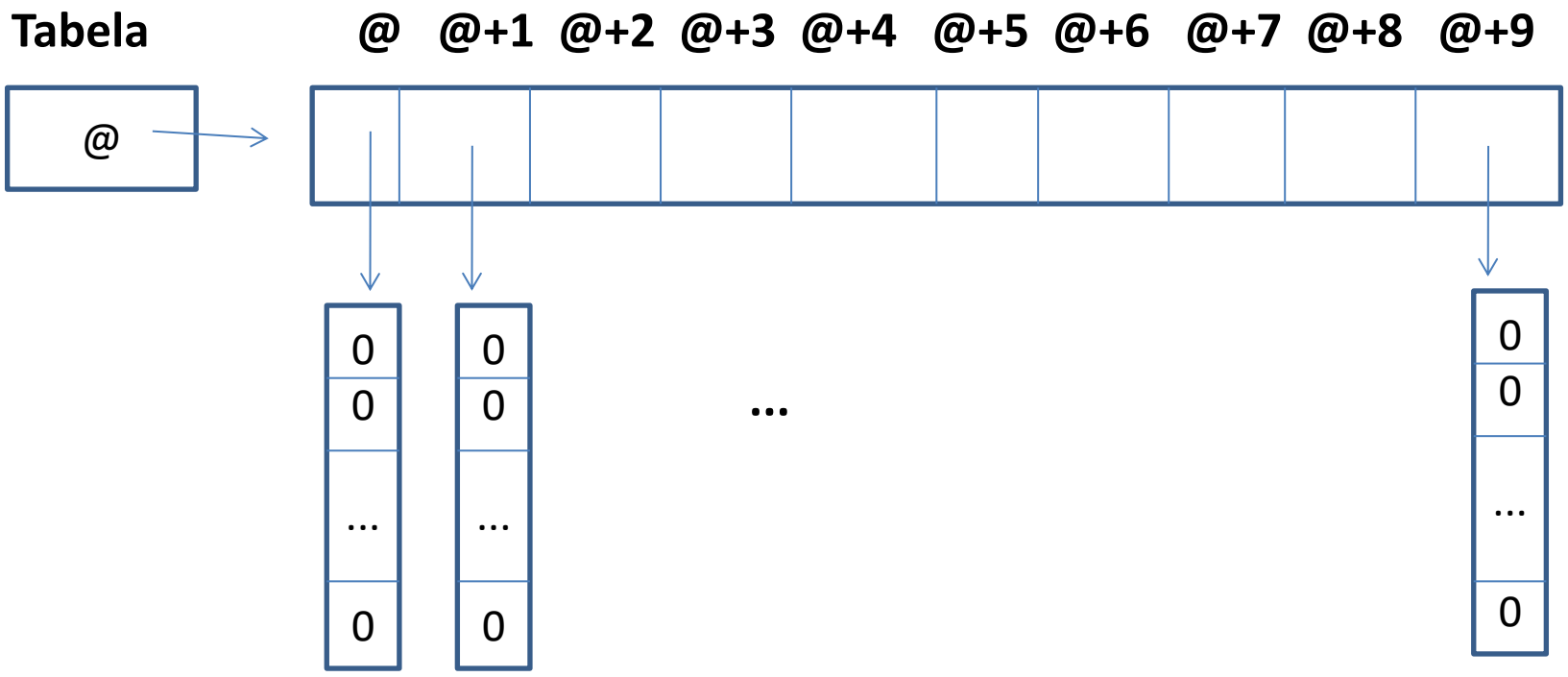
A multi-dimensionalidade é implementada como aninhamento
(array de arrays)

```
int [] [] tabela = new int[10] [20];

/*tabela é um array bidimensional de 10 linhas e 20 colunas*/
```

```
int tabela[] [] = new int[10] [20];

/* declaração equivalente à anterior */
```



int[][] matriz = new int[3][];  ?

Neste caso, seria alocado um array de 3 posições,

com referências de valor null, de momento, para cada um dos elementos a alocar nestas posições,

sabendo-se, que tais elementos devem ser do tipo int[], ou seja, arrays de inteiros.

Os elementos do tipo int[] deste array podem ter dimensões distintas...

```
int[][] matriz = { {1,2}, {10, 20, 30}, {19, 2, 34, 21, 16}};
```

**Acesso aos elementos de um array**

```
int[] a = new int[20] ;

int[][] tabela = new int[20][];
...
tabela[10] = new int[5];
...
int x, y;
x= tabela[10][2] + a[4]*a[7];

x= a[0];

y = a[a.length-1];
a[0 ] = x + y;
```

...com *array.length* evita-se o acesso a elementos inexistentes (**ArrayIndexOutOfBoundsException**)

Cópia de uma tabela ?

**Assim?**

```
int[] b;

b = a;
```

**Ou assim?**

```
int[] b = new int[20] ;

for(int i = 0; i< a.length ; i++ ) b[i] = a[i] ;
```