



Java Básico

Estructuras de Datos 1

Copyright

- Copyright (c) 2004
José M. Ordax
- Este documento puede ser distribuido solo bajo los términos y condiciones de la Licencia de Documentación de javaHispano v1.0 o posterior.
- La última versión se encuentra en
<http://www.javahispano.org/licencias/>

Arrays

- Un array es una estructura de datos que permite albergar varios elementos del mismo tipo.
- La longitud de un array se establece durante su creación.
- Una vez establecida la longitud de un array, ya no se puede modificar.
- Un elemento de un array, es el valor de una de sus posiciones, y se identifica mediante un índice.

Arrays

- Un array en Java, es un tipo de clase especial que hereda implícitamente de `java.lang.Object`.
- La declaración de un array se realiza mediante el tipo de datos que va a albergar y los corchetes `[]`.
- Declaración de un array:
 - `modificador_acceso tipo[] nombre [= valor_inicial];`
- Ejemplo:
 - `private int[] numeros;`
 - `private String[] cadenas;`

Arrays

- Los arrays pueden albergar tanto tipos primitivos como tipos complejos.
- Un array sin inicializar, por defecto vale null.
- La creación de un array se realiza mediante la keyword: new, como con cualquier otra clase.
- Creación de un array:
modificador_acceso tipo[] nombre = new tipo[longitud];
- Ejemplo:
private int[] numeros = **new** int[5];
private String[] cadenas = **new** String[4];

Arrays

- Una vez hayamos creado un array, todas sus posiciones son inicializadas al valor por defecto del tipo de variable que albergue.
- Es decir, 0 o 0.0 si se trataba de un número, false si se trataba de un boolean y null si se trataba de un tipo complejo.
- Existe una forma de crear un array inicializando todas sus posiciones a un valor determinado, igualándolo a un listado de elementos separados por comas entre { }.
- El tamaño del array será el número de elementos del listado.

Arrays

Ejemplos:

```
private int[] numeros = {1,2,3,4,5};  
private String[] cadenas = {"hola","adios"};  
private Integer[] ints = {new Integer(12), new Integer(98)};
```

Para el acceso al elemento de un array se utiliza en nombre del array seguido de unos [] con la posición a la que queremos acceder.

La primera posición de un array es la 0.

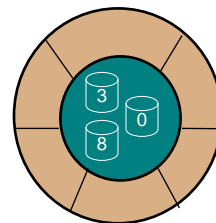
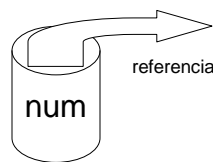
Ejemplo:

```
numeros[2] = 3; int a = numeros[0]; if(numeros[4] == 5).....
```

Arrays

En memoria.....

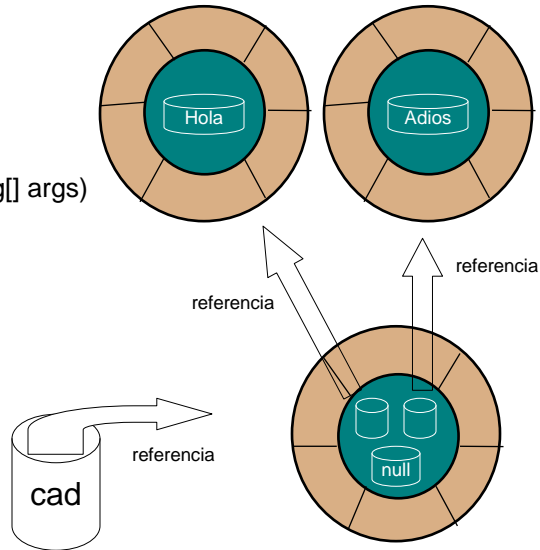
```
public class Colecciones  
{  
    public static void main(String[] args)  
    {  
        int[] num = new int[3];  
        num[0] = 3;  
        num[1] = 8;  
    }  
}
```



Arrays

En memoria.....

```
public class Colecciones
{
    public static void main(String[] args)
    {
        String[] cad = new String[3];
        cad[0] = new String("Hola");
        cad[1] = new String("Adios");
    }
}
```



Arrays

- Para conocer la longitud de un array, podremos acceder a su atributo público: `length`.
- El índice de un array es de tipo `int`.
- Al no ser dinámico, no podemos:
 - Ni eliminar posiciones.
 - Ni insertar posiciones.
- El borrado será algo lógico, como igualar las posiciones a `null`, a `-1`, etc.... dependerá del desarrollador.

Arrays

- Es imposible acceder a una posición fuera del array. Llegado el caso se lanzará una excepción:
 - ArrayIndexOutOfBoundsException
- Ya veremos el tratamiento de excepciones en otro capítulo.

Ejemplo

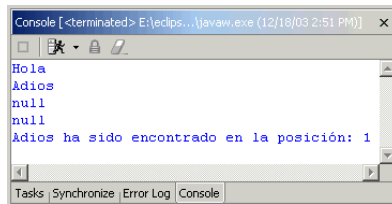
```
public class Colecciones
{
    // Recorrido.
    public static void main(String[] args)
    {
        // Creación e inicialización.
        String[] saludos = new String[4];

        // Inserción.
        saludos[0] = new String("Hola");
        saludos[1] = new String("Adios");
        saludos[2] = new String("Hello");
        saludos[3] = new String("GoodBye");

        // Extracción.
        String tmp = saludos[2];

        // Borrado.
        saludos[2] = null;
        saludos[3] = null;

        // Búsqueda.
        boolean sw = false;
        for(int i=0; i<saludos.length; i++)
        {
            if(saludos[i] != null && saludos[i].equals("Adios"))
            {
                System.out.println("Adios ha sido encontrado en la posición: " + i);
                sw = true;
                break;
            }
        }
    }
}
```



Sentencia for/in

- Esta nueva sentencia del Java SE 5.0 nos facilita la iteración por los elementos de cualquier tipo de colección: arrays, listas, etc...

- ```
for(inicialización: colección) Nota: Se usa ":" en vez de "=".
{
 sentencias;
}
```

- Ejemplo:

- ```
public void listar(int[] param)  
{  
    for(int i: param)  
        System.out.println(i);  
}
```

Sentencia for/in

- Básicamente, se trata de una simplificación a la hora de codificar.

- Es decir, al final, el compilador convierte el código en una sentencia *for* convencional:

- ```
public void listar(int[] param)
{
 for(int i=0; i<param.length; i++)
 System.out.println(param[i]);
}
```

- Veremos como trabajar con este nuevo tipo de sentencia en el capítulo dedicado a Java SE 5.0

## Arrays multidimensionales

- Un array multidimensional es un array de arrays.
- Es decir, una matriz de 4 x 2 en realidad en Java está formada por 5 arrays: 1 array con 4 arrays.
- Y si estuviéramos hablando de tres dimensiones entonces tendríamos un array de arrays de arrays, pero mejor lo vamos a dejar en dos dimensiones.
- Creación de un array bidimensional:  
*modificador\_acceso tipo[][] nombre = new tipo[long][long];*
- Ejemplo:  
**private** int[][] numeros = **new** int[4][2];

## Arrays multidimensionales

- Pero podemos tener arrays bidimensionales no cuadradas. Es decir, que la segunda dimensión tenga longitud diferente dependiendo de la primera dimensión.
- Creación de un array bidimensional:  
*modificador\_acceso tipo[][] nombre = new tipo[long][];*
- Ejemplo:  
**private** int[][] numeros = **new** int[4][];  
numeros[0] = **new** int[2];  
numeros[1] = **new** int[10];  
numeros[3] = **new** int[1];



# Arrays multidimensionales

Al igual que ocurriera en los arrays de una dimensión, también se pueden inicializar en la creación con un listado de valores.

Ejemplos:

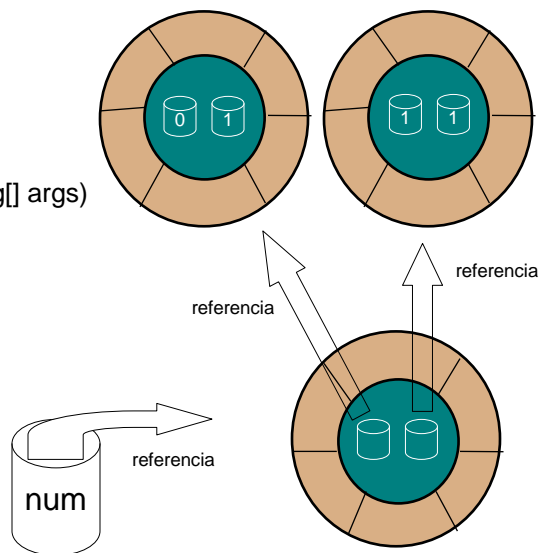
```
private int[][] numeros = { {1,2,3} , {1,2,3} };
```

```
private String[][] dias = { {"Lunes","Martes"} , {"Miércoles"} };
```

## Arrays

En memoria.....

```
public class Colecciones
{
 public static void main(String[] args)
 {
 int[] num = new int[2][2];
 num[0][0] = 0;
 num[0][1] = 1;
 num[1][0] = 1;
 num[1][1] = 1;
 }
}
```

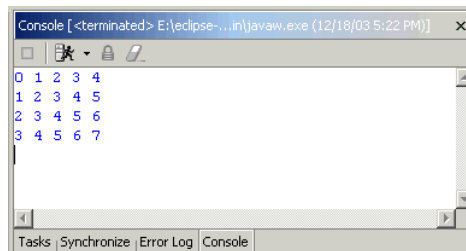


## Ejemplo

```
public class Matriz
{
 public static void main(String[] args)
 {
 int[][] matriz = new int[4][];

 // Rellenar la matriz
 for(int i=0; i<matriz.length; i++)
 {
 matriz[i] = new int[5];
 for(int j=0; j<matriz[i].length; j++)
 matriz[i][j] = i + j;
 }

 // Mostrar la matriz
 for(int i=0; i<matriz.length; i++)
 {
 for(int j=0; j<matriz[i].length; j++)
 System.out.print(matriz[i][j] + " ");
 System.out.println();
 }
 }
}
```



```
Console [<terminated> E:\eclipse-...in\javaw.exe (12/18/03 5:22 PM)]
0 1 2 3 4
1 2 3 4 5
2 3 4 5 6
3 4 5 6 7
```

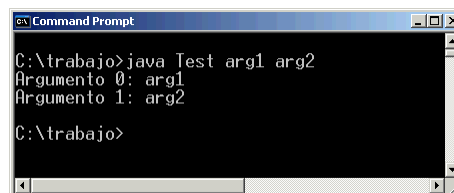
## El método main usa arrays



El método main recibe un array de Strings que contine los argumentos enviados en el arranque.

**public static void** main(String[] args)

```
public class Test
{
 public static void main(String[] args)
 {
 for(int i=0; i<args.length; i++)
 System.out.println("Argumento " + i + ": " + args[i]);
 }
}
```



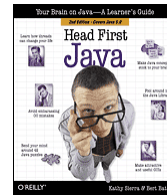
```
Command Prompt
C:\trabajo>java Test arg1 arg2
Argumento 0: arg1
Argumento 1: arg2
C:\trabajo>
```

# Bibliografía



## Head First Java (2<sup>nd</sup> edition)

Kathy Sierra y Bert Bates.  
O'Reilly



## Learning Java (2<sup>nd</sup> edition)

Patrick Niemeyer y Jonathan Knudsen.  
O'Reilly.



## Thinking in Java (4<sup>th</sup> edition)

Bruce Eckel.  
Prentice Hall.



## The Java tutorial

<http://java.sun.com/docs/books/tutorial/>